

A WEIGHTED ALGORITHM FOR FREQUENT ITEM SET DATA MINING

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Engineering degree in Computer Science and Engineering

by

Sunil.S (38110574)
,Yanamadala Rahul chowdary (38110661)



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERISTY)

Accredited with “A” by NAAC

JEPPIAR NAGAR, RAJIV GANDHI SALAI,

CHENNAI – 600119

APRIL 2021



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited with "A" grade by NAAC

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600119

www.sathyabama.ac.in



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

**This is to certify that this Project Report is the bonafide work of Sunil.S
(38110574)**

who carried out the project entitled "**A WEIGHTED ALGORITHM FOR FREQUENT
ITEM SET DATA MINING**" under my supervision from November 2019 to April 2021.

Internal Guide

Dr. VIGNESWARY M.E., Ph.D.,

Head of the Department

Dr. L. Lakshmanan M.E., Ph.D.,

Dr. S. Vigneshwari M.E., Ph.D.,

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I Sunil.S (38110574)

hereby declare that the Project Report entitled as

“A WEIGHTED ALGORITHM FOR FREQUENT ITEM SET DATA MINING” project done by me under the guidance of **Dr. SENDURU SRINIVASULU**, and at Sathyabama Institute of Science and Technology is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

SIGNATURE OF THE CANDIDATE

PLACE:

ACKNOWLEDGEMENT

I am pleased to acknowledge our sincere thanks to Board of Management of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. SASIKALA, M.E., Ph.D., Dean, School of Computing** and also **Dr. S. VIGNESHWARI, M.E., Ph.D., and Dr. L. LAKSHMANAN, M.E., Ph.D., Heads of the Department, Department of Computer Science and Engineering** for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. VIGNESWARI**, for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

The key technology is the smart consumer advice Mining systems in information technology plays an increasingly an important role in the actions and decision making. Also common Mining is an important step research properties of association rules. It is one of the most important areas of research in data mining. Database to learn by frequent jobs in matters of doubt, the weighted they are so likely to be important, and the elements into often found great item sets users. But the introduction of weight makes frequent item sets may fall back to the enclosure is to be a matter longer. Hence frequently used items in the distance, not to search they are what are referred to the poor efficiency. Said weight trial frequency locked to its own gravity and item set sit is a serious amount of material introduced and tested. From these two properties WD-FIM. (own weight and the end of the trial Evaluation of the algorithm is set before us by frequent, repeated the space of the face of the frequent item sets, and improves the hurdle weighted with stones then efficiency. We proposed the alternative method of Pincer-search algorithm yields maximum frequent items, association rules, accuracy and minimum execution time.

TABLE OF CONTENTS

Chapter No	TITLE	Page No
	ABSTRACT	v
	LIST OF FIGURES	viii
	List OF TABLES	ix
	LIST OF ABBREVIATIONS	x
1.	INTRODUCTION	1
	1.1 INTRODUCTION	1
	1.2 EXISTING SYSTEM	2
	1.3 PROPOSED SYSTEM	3
2.	LITERATURE SURVEY	4
3.	SYSTEM ARCHITECTURE & REQUIREMENTS	8
	3.1 INTRODUCTION	8
	3.2 ARCHITECTURE DIAGRAM	8
	3.3 HARDWARE REQUIREMENTS	8
	3.4 SOFTWARE REQUIREMENTS	9
	3.4.1 APPLICATION OF JAVA	9
	3.4.1.1 FEATURES OF JAVA	9
	3.4.2 COLLECTION FRAMEWORK	12
	3.4.3 MYSQL	13
	3.4.4 REQUIREMENT ANALYSIS	13
	3.4.5 FUNCTIONAL REQUIREMENTS	13
	3.4.6 NON-FUNCTIONAL REQUIREMENTS	14
	FEASIBILITY STUDY	15
	ECONOMICAL FEASIBILITY	15
	TECHNICAL FEAIBILITY	16
	SOCIAL FEASIBILITY	16

4.	METHODOLOGY	17
	4.1 ALGORITHM	17
	4.2 MODULE DESCRIPTION	17
	4.2.1 REGISTER MODULE	18
	4.2.2 SIGN IN MODULE	18
	4.2.3 USER MODULE	18
	4.2.4 ADMIN MODULE	18
	4.3 FREQUENT DATA SET MODULE	18
	4.4 UML DIAGRAMS	19
	4.4.1 USE CASE DIAGRAM	20
	4.4.2 SEQUENCE DIAGRAM	21
	4.4.3 COLLABORATION DIAGRAM	22
	4.4.4 ACTIVITY DIAGRAM	23
5.	CONCLUSION	24
6.	RESULTS AND DISCUSSION	25
	REFERENCES	27
	APPENDICES	
	SOURCE CODE	28
	B) SCREENSHOTS	44
	C) PUBLICATION PAPER	46

LIST OF FIGURES

FIGURE NO	TITLE	PAGE NO
1.3	F-score in % Data set	3
3.2	Architecture diagram	8
3.4.1.1	Platform Independent Diagram	11
3.4.2	Collection Framework	12
4.4.1	Use case diagram	20
4.4.2	Sequence diagram	21
4.4.3	Collaboration diagram	22
4.4.4	Activity diagram	23

LIST OF TABLES

FIG.NO	TITLE	PAGE NO
1.3	A transaction database	3

LIST OF ABBREVIATIONS

Abbreviation	Full form
WD-FIM	Weight judgement Downward closure property based frequent Item sets Mining
ARM	Association rule mining
IWFPM	Interested Weighted Frequent Pattern Mining with Multiple supports
IWI	Infrequent Weighted Itemset
WARM	Weighted Association Rule Mining
FIM	Frequent Item set Mining
RDBMS	Relational Database Management System
UML	Unified Modeling Language

CHAPTER 1

INTRODUCTION

The key technology is the smart consumer advice Mining systems in information technology plays an increasingly an important role in the actions and decision making. The weight judgment downward closure property for weighted frequent item sets and the existence property of weighted frequent subsets are introduced and proved first. Based on these two properties, the WD-FIM (Weight judgment Downward closure property based Frequent Item sets Mining) algorithm is proposed to narrow the searching space of weighted frequent item sets and improve the time efficiency. Data mining is an emerging technique that addresses the problem of restructuring the data into the useful information. The rule generation methods are used in the area of data mining to determine the relationship between different items. The Association Rule Mining (ARM) is widely used for finding data patterns that reveals the combination of events occurring simultaneously based on the associations among a large set of data items. The Apriori algorithm (Agrawal and Srikant, 1994) is a popular algorithm for extracting high frequent item sets from a database using the predefined threshold measures such as minimum support and minimum confidence. In this paper, on the basis of the weight judgment downward closure property, the Pincer search algorithm is proposed to yields maximum frequent items, association rules, accuracy and minimum execution time.

The main contributions of this paper are listed as following.

The weight judgment downward closure property and the existence property of weighted frequent subsets for uncertain databases are introduced and proved. The weight judgment downward closure property can be used to narrow the searching space of weighted frequent item sets. The existence property of weighted frequent subsets can ensure all the weighted frequent item sets be discovered.

The Pincer-search algorithm is proposed to yields maximum frequent items, association rules, accuracy and minimum execution time.

A considerable amount of experiments are conducted on both real-life and synthetic datasets to evaluate the performance of the proposed Pincer-Search algorithm in terms of runtime, number of patterns and memory consumption.

Frequent item set mining is one of the best known and most popular data mining methods. Originally developed for market basket analysis, it is used nowadays for almost any task that requires discovering regularities between (nominal) variables. This paper provides an overview of the foundations of frequent item set mining, starting from a definition of the basic notions and the core task. It continues by discussing how the search space is structured to avoid redundant search, how it is pruned with the a priori property, and how the output is reduced by confining it to closed or maximal item sets or generators. In addition, it reviews some of the most important algorithmic techniques and data structures that were developed to make the search for frequent item sets as efficient as possible

EXISTING SYSTEM

A hyperlinked structure based algorithm called UH-mine to mine frequent patterns from uncertain data.

a tree-based mining algorithm called UF-growth which also constructs a tree structure to store the contents of the uncertain datasets, like its counterpart - the FP-growth algorithm for mining precise data.

In order to reduce the tree size at proposed the UFP-growth algorithm. To further reduce the tree size. An uncertain frequent pattern mining algorithm called CUF growth, which builds a new tree structure called CUF-tree.

AT-Mine is another tree-based efficient approach proposed to overcome the fatal problems of CUFP-Mine tree.

Disadvantages of existing system

In the WDFIM method to using to get low amount of mining item sets.

Low mining efficiency and coverage

Proposed system

We introduce an Pincer-search algorithm for efficient mining of the frequent item sets. The prefiltering method is applied to the input dataset to remove the item having low variance. Data discretization is performed and Pincer-search is applied for mining the frequent item sets.

The proposed Pincer-search algorithm yields maximum frequent items, association rules, accuracy and minimum execution time than the existing algorithms.

Advantages of proposed system:

Prediction accuracy is improved by using the association rules.

Effective generation of the frequent item sets and association rules is ensured by maintaining the feasibility of the neural network.

Table 1.3.1.(a): A transaction database as running example

T-id	Items Bought	Frequent Items(ordered)
101	W,t,u,g,s,d,f	u,s,d,w
102	h,a,l,e,l,d	d,e,i
103	J,q,o,f,w	o,q
104	k,d,z,v,f	d,f,k
105	e,u,y,f,o,p	y,f,e

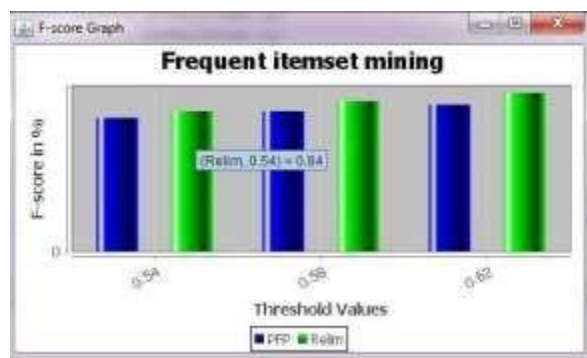


Fig.1.3.2 (b): F-score in % on Data Set

CHAPTER 2

LITERATURE SURVEY

INTRODUCTION

Literature survey is the most important step in software development process. Before developing the tool it is necessary to determine the time factor, economy and company strength. Once these things are satisfied, then the next step is to determine which operating system and language can be used for developing the tool. Once the programmers start building the tool the programmers need lot of external support. This support can be obtained from senior programmers, from book or from websites. Before building the system the above consideration are taken into account for developing the proposed system. The major part of the project development sector considers and fully survey all the required needs for developing the project. For every project Literature survey is the most important sector in software development process. Before developing the tools and the associated designing it is necessary to determine and survey the time factor, resource requirement, man power, economy, and company strength. Once these things are satisfied and fully surveyed, then the next step is to determine about the software specifications in the respective system such as what type of operating system the project would require, and what are all the necessary software are needed to proceed with the next step such as developing the tools, and the associated operations.

Title 1: IWFPM: Interested Weighted Frequent Pattern Mining with Multiple Supports

Author: Xuyang Wei^{1*}, Zhongliang Li¹, Tengfei Zhou¹, Haoran Zhang¹, Guocai Yang¹

Association rules mining has been under great attention and considered as one of momentous area in data mining. Classical association rules mining approaches make implicit assumption that items'importance is the same and set a single support for all items. This paper presents an efficient approach for mining users' interest weighted frequent patterns from a transactional database. Our paradigm is to assign appropriate minimum support (minsup) and weight for each item, which reduces the number of unnecessary patterns. Furthermore, we also extend the support-confidence framework and define an interest measure to the mining algorithm for excavating users' interested patterns effectively. In the end, experiments on both synthetic and real world datasets show that the proposed algorithm can generate more interested patterns.

Title 2: Discovery of Infrequent Weighted Itemset with High Utility

Author:KalyaniTukaramBhandwalkar¹ ,Mansi Bhonsle²

Knowledge discovery has been an interesting area of research due to its various applications. Traditionally, frequent pattern mining plays an important role. Generally, infrequent items within the dataset are ignored. Infrequent Itemset mining is a variation of frequent itemset mining where rarely occurring patterns are discovered. This paper tackles the issue of discovering rare and weighted item sets, i.e., the infrequent weighted itemset (IWI) mining problem. Also high utility item sets are discovered using UP Growth algorithm. Proposed system not only considers the frequency of the item sets but also considers the utility associated with the item sets.

Title 3: An Uncertainty-based Approach: Frequent Itemset Mining from Uncertain Data with Different Item Importance

Author:Gangin Lee, Unil Yun¹ and HeungmoRyang

Since Itemset mining was proposed, various approaches have been devised, ranging from processing simpleitem-based databases to dealing with more complex databases including sequence, utility, or graph information. Especially, in contrast to the mining approaches that process such databases containing exact presence or absence information of items, uncertain pattern mining finds meaningful patterns from uncertain databases with items' existential probability information. However, traditional uncertain mining methods have a problem in that it cannot apply importance of each item obtained from the real world into the mining process. In this paper, to solve such a problem and perform uncertain itemset mining operations more efficiently, we propose a new uncertain itemset mining algorithm additionally considering importance of items such as weight constraints. In our algorithm, both items' existential probabilities and weight factors are considered; as a result, we can selectively obtain more meaningful item sets with high importance and existential probabilities. In addition, the algorithm can operate more quickly with less memory by efficiently reducing the number of calculations causing useless itemset generations. Experimental results in this paper show that the proposed algorithm is more efficient and scalable than state-of-the-art methods.

Title 4: Valency BasedWeighted Association Rule Mining

Author: Yun Sing Koh, Russel Pears, and WaiYeap

Association rule mining is an important data mining task that discovers relationships among items in a transaction database. Most approaches to association rule mining assume that all items within a dataset have a uniform distribution with respect to support. Therefore, weighted association rule mining (WARM) was introduced to provide a notion of importance to individual items. Previous approaches to the weighted association rule mining problem require users to assign weights to items. This is infeasible when millions of items are present in a dataset. In this paper we propose a method that is based on a novel Valency model that automatically infers item weights based on interactions between items. Our experimentation shows that the weighting scheme results in rules that better capture the natural variation that occurs in a dataset when compared to a miner that does not employ such a weighting scheme.

Title 5: Mining Weighted Frequent Item sets without Candidate Generation in Uncertain Databases

Author: Jerry Chun-Wei Lin, WenshengGan, Philippe Fournier-Viger, Tzung-Pei Hongand Han-Chieh Chao

Frequent itemset mining (FIM) is a fundamental set of techniques used to discover useful and meaningful relationships between items in transaction databases. In recent decades, extensions of FIM such as weighted frequent itemset mining (WFIM) and frequent itemset mining in uncertain databases (UFIM) have been proposed. WFIM considers that items may have different weight/importance. It can thus discover item sets that are more useful and meaningful by ignoring irrelevant item sets with lower weights. UFIM takes into account that data collected in a real-life environment may often be inaccurate, imprecise, or incomplete. Recently, these two ideas have been combined in the HEWI-Uapriori algorithm. This latter considers both item weights and transaction uncertainty to mine the high expected weighted item sets (HEWIs) using a two-phase Apriori-based approach. Although the upper-bound proposed in HEWIUapriori can reduce the size of the search space, it still generates a large amount of candidates and uses a

level-wise search. In this paper, a more efficient algorithm named HEWI-Utree is developed to efficiently mine HEWIs without performing multiple database scans and without generating candidates. This algorithm relies on three novel structures named element (E)-table, weighted-probability (WP)-table and WP-tree to maintain the information required for identifying and pruning unpromising item sets early. Experimental results show that the proposed algorithm is generally much more efficient than traditional methods for WFIM and UFIM, as well as the state-of-the-art HEWI-Uapriori algorithm, in terms of runtime, memory consumption, and scalability.

CHAPTER 3

SYSTEM ARCHITECTURE & REQUIREMENTS

INTRODUCTION

Design is a multi- step that focuses on data structure software architecture, procedural details, algorithm etc... and interface between modules. The design process also translate the requirements into presentation of software that can be accessed for quality before coding begins. Computer software design change continuously as new methods; better analysis and border understanding evolved. Software design is at relatively early stage in its revolution. Therefore, software design methodology lacks the depth, flexibility and quantitative nature that are normally associated with more classical engineering disciplines. However techniques for software designs do exit, criteria for design qualities are available and design notation can be applied.

ARCHITECTURE DIAGRAM

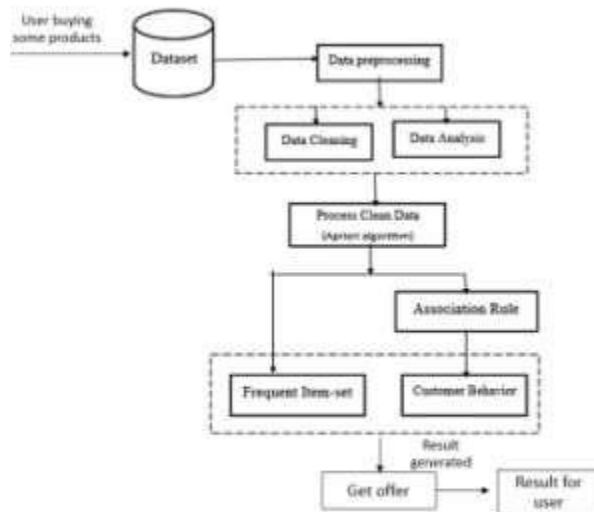


Fig 3.2: Architecture Diagram

Hardware requirements

System : Pentium IV 2.4 GHz.
Hard Disk : 40 GB.
Monitor : 15 VGA Colour.
Mouse : Logitech.
Ram : 512 Mb.

Software requirements

Operating system : Windows XP/7.

Coding Language :java

Tool : Netbeans

Database : MYSQL

INTRODUCTION

Java is one of the world's most important and widely used computer languages, and it has held this distinction for many years. Unlike some other computer languages whose influence has waned with passage of time, while Java's has grown.

APPLICATION OF JAVA

Java is widely used in every corner of world and of human life. Java is not only used in software but is also widely used in designing hardware controlling software components. There are more than 930 million JRE downloads each year and 3 billion mobile phones run java.

Following are some other usage of Java:

Developing Desktop Applications

Web Applications like Linkedin.com, Snapdeal.com etc

Mobile Operating System like Android

Embedded Systems

Robotics and games etc.

FEATURES OF JAVA

The prime reason behind creation of Java was to bring portability and security feature into a computer language. Beside these two major features, there were many other features that played an important role in moulding out the final form of this outstanding language. Those features are;

1) Simple

Java is easy to learn and its syntax is quite simple, clean and easy to understand. The confusing and ambiguous concepts of C++ are either left out in Java or they have been re-implemented in a cleaner way.

Eg: Pointers and Operator Overloading are not there in java but were an important part of C++.

2) Object Oriented

In java everything is Object which has some data and behaviour. Java can be easily extended as it is based on Object Model.

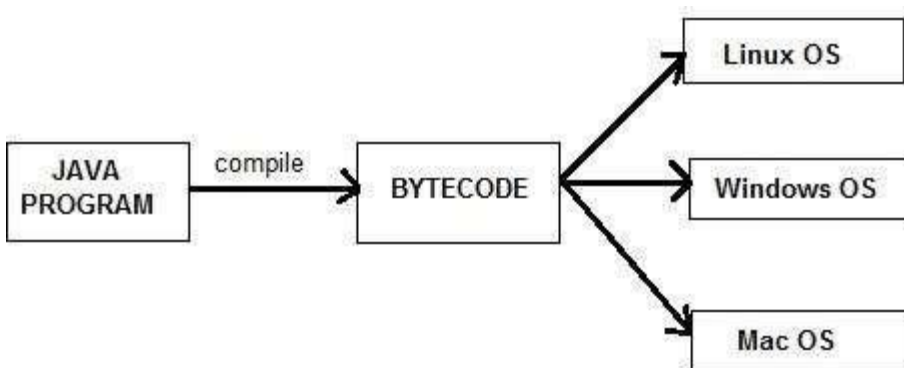
3) Robust

Java makes an effort to eliminate error prone codes by emphasizing mainly on compile time error checking and runtime checking. But the main areas which Java improved were Memory Management and mishandled Exceptions by introducing automatic Garbage Collector and Exception Handling.

4) Platform Independent

Unlike other programming languages such as C, C++ etc. which are compiled into platform specific machines. Java is guaranteed to be write-once, run-anywhere language.

On compilation Java program is compiled into byte code. This byte code is platform independent and can be run on any machine, plus this byte code format also provide security. Any machine with Java Runtime Environment can run Java Programs.



5) Secure

When it comes to security, Java is always the first choice. With java secure features it enable us to develop virus free, temper free system. Java program always runs in Java runtime environment with almost null interaction with system OS, hence it is more secure.

6) Multi-Threading

Java multithreading feature makes it possible to write program that can do many tasks simultaneously. Benefit of multithreading is that it utilizes same memory and other resources to execute multiple threads at the same time, like While typing, grammatical errors are checked along.

7) Architectural Neutral

Compiler generates byte codes, which have nothing to do with a particular computer architecture, hence a Java program is easy to interpret on any machine.

8) Portable

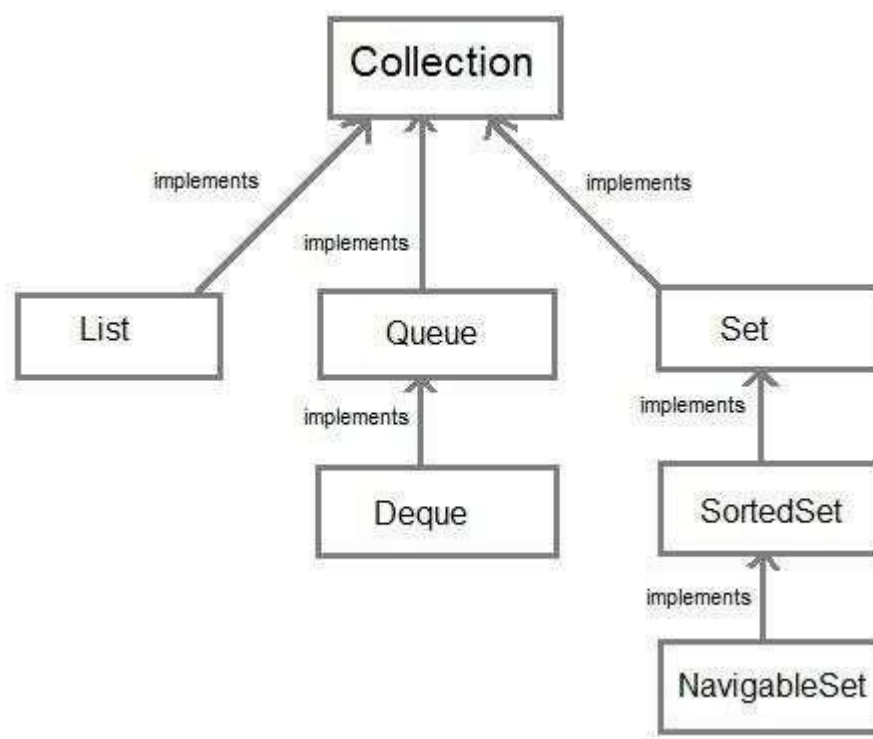
Java Byte code can be carried to any platform. No implementation dependent features. Everything related to storage is predefined, example: size of primitive data types

10) High Performance

Java is an interpreted language, so it will never be as fast as a compiled language like C or C++. But, Java enables high performance with the use of just-in-time compiler.

COLLECTION FRAMEWORK

Collection framework was not part of original Java release. Collections was added to J2SE 1.2. Prior to Java 2, Java provided adhoc classes such as Dictionary, Vector, Stack and Properties to store and manipulate groups of objects. Collection framework provides many important classes and interfaces to collect and organize group of alike objects.



MYSQL

MySQL, officially, but also called "My Sequel" is the world's most widely used open-source relational database management system (RDBMS) that runs as a server providing

multi-user access to a number of databases, though SQLite probably has more total embedded deployments. The SQL phrase stands for Structured Query Language.

The MySQL development project has made its source code available under the terms of the GNU General Public License, as well as under a variety of proprietary agreements. MySQL was owned and sponsored by a single for-profit firm, the Swedish company MySQL AB, now owned by Oracle Corporation. MySQL is a popular choice of database for use in web applications, and is a central component of the widely used LAMP open source web application software stack (and other 'AMP' stacks).

LAMP is an acronym for "Linux, Apache, MySQL, Perl/PHP/Python." Freesoftware-open source projects that require a full-featured database management system often use MySQL.

For commercial use, several paid editions are available, and offer additional functionality. Applications which use MySQL databases include: TYPO3, MODx, Joomla, WordPress, phpBB, MyBB, Drupal and other software. MySQL is also used in many high-profile, large-scale websites, including Wikipedia, Google (though not for searches), Facebook, Twitter, Flickr and YouTube.

REQUIREMENT ANALYSIS

Requirement analysis, also called requirement engineering, is the process of determining user expectations for a new modified product. It encompasses the tasks that determine the need for analysing, documenting, validating and managing software or system requirements. The requirements should be documentable, actionable, measurable, testable and traceable related to identified business needs or opportunities and define to a level of detail, sufficient for system design.

FUNCTIONAL REQUIREMENTS

It is a technical specification requirement for the software products. It is the first step in the requirement analysis process which lists the requirements of particular software systems including functional, performance and security requirements. The function of the system depends mainly on the quality hardware used to run the software with given functionality.

Usability

It specifies how easy the system must be use. It is easy to ask queries in any format which is short or long, porter stemming algorithm stimulates the desired response for user.

Robustness

It refers to a program that performs well not only under ordinary conditions but also under unusual conditions. It is the ability of the user to cope with errors for irrelevant queries during execution.

Security

The state of providing protected access to resource is security. The system provides good security and unauthorized users cannot access the system there by providing high security.

Reliability

It is the probability of how often the software fails. The measurement is often expressed in MTBF (Mean Time Between Failures). The requirement is needed in order to ensure that the processes work correctly and completely without being aborted. It can handle any load and survive and survive and even capable of working around any failure.

Compatibility

It is supported by version above all web browsers. Using any web servers like localhost makes the system real-time experience.

Flexibility

The flexibility of the project is provided in such a way that is has the ability to run on different environments being executed by different users.

Safety

Safety is a measure taken to prevent trouble. Every query is processed in a secured manner without letting others to know one's personal information.

NON- FUNCTIONAL REQUIREMENTS

Portability

It is the usability of the same software in different environments. The project can be run in any operating system.

Performance

These requirements determine the resources required, time interval, throughput and everything that deals with the performance of the system.

Accuracy

The result of the requesting query is very accurate and high speed of retrieving information. The degree of security provided by the system is high and effective.

Maintainability

Project is simple as further updates can be easily done without affecting its stability. Maintainability basically defines that how easy it is to maintain the system. It means that how easy it is to maintain the system, analyse, change and test the application. Maintainability of this project is simple as further updates can be easily done without affecting its stability.

FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

The feasibility study investigates the problem and the information needs of the stakeholders. It seeks to determine the resources required to provide an information systems solution, the cost and benefits of such a solution, and the feasibility of such a solution.

The goal of the feasibility study is to consider alternative information systems solutions, evaluate their feasibility, and propose the alternative most suitable to the organization. The feasibility of a proposed solution is evaluated in terms of its components.

ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity.

CHAPTER 4

METHODOLOGY

ALGORITHM

PINCER-SEARCH ALGORITHM

We now present our complete algorithm, The pincer-search algorithm, which relies on the combined approach for determining the maximum frequent set.

01. $L_0 := \emptyset$; $k := 1$; $C_1 := \{ \{ i \} \mid i \in I \}$
02. $\text{MFCS} := \{ \{1, 2, \dots, n\} \}$; $\text{MFS} := \emptyset$
03. while $C_k \neq \emptyset$
04. read database and count supports for C_k and MFCS
05. remove frequent item sets from MFCS and add them to MFS
06. determine frequent set L_k and infrequent set S_k
07. use S_k to update MFCS
08. generate new candidate set C_{k+1} (join, recover, and prune)
09. $k := k + 1$
10. return MFS

The weight judgment downward closure property and the existence property of weighted frequent subsets for uncertain databases are introduced and proved. The weight judgment downward closure property can be used to narrow the searching space of weighted frequent item sets. The existence property of weighted frequent subsets can ensure all the weighted frequent item sets be discovered.

The pincer-search algorithm is proposed to yields maximum frequent items, association rules, accuracy and minimum execution time.

3. A considerable amount of experiments are conducted on both real-life and synthetic datasets to evaluate the performance of the proposed pincer-search algorithm in terms of runtime, number of patterns and memory consumption.

MODULE DESCRIPTION

PROJECT MODULES

Registration Module

Sign in module

User module

Admin module

Frequent Dataset module

Register Module

This module is used for the user to register their login id by providing the minimal information. So that they can login to the website.

Sign in module

In this, user can login to the website by registered login id and a valid password. Only the authenticated user can login and use the website.

User module

In this module, user extracts item sets which are frequently brought together with an overall rating above average.

Admin module

In this module, admin checks the item list, add the items and remove the unwanted items.

Frequent Dataset module

We introduce anPincer-searchalgorithm for efficient mining of the frequent item sets. The prefiltering method is applied to the input dataset to remove the item having low variance.

UML DIAGRAMS

UML is simply another graphical representation of a common semantic model. UML provides a comprehensive notation for the full lifecycle of object-oriented development.

ADVANTAGES

To represent complete systems (instead of only the software portion) using object-oriented concepts

To establish an explicit coupling between concepts and executable code

To take into account the scaling factors that are inherent to complex and critical systems

To creating a modeling language usable by both humans and machines

UML defines several models for representing systems

The class model captures the static structure

The state model expresses the dynamic behavior of objects

The use case model describes the requirements of the user

The interaction model represents the scenarios and messages flows

The implementation model shows the work units

The deployment model provides details that pertain to process allocation

Use Case Diagram

Use case diagrams overview the usage requirement for system. They are useful for presentations to management and/or project stakeholders, but for actual development you will find that use cases provide significantly more value because they describe “the meant” of the actual requirements. A use case

describes a sequence of action that provides something of measurable value to an action and is drawn as a horizontal ellipse.

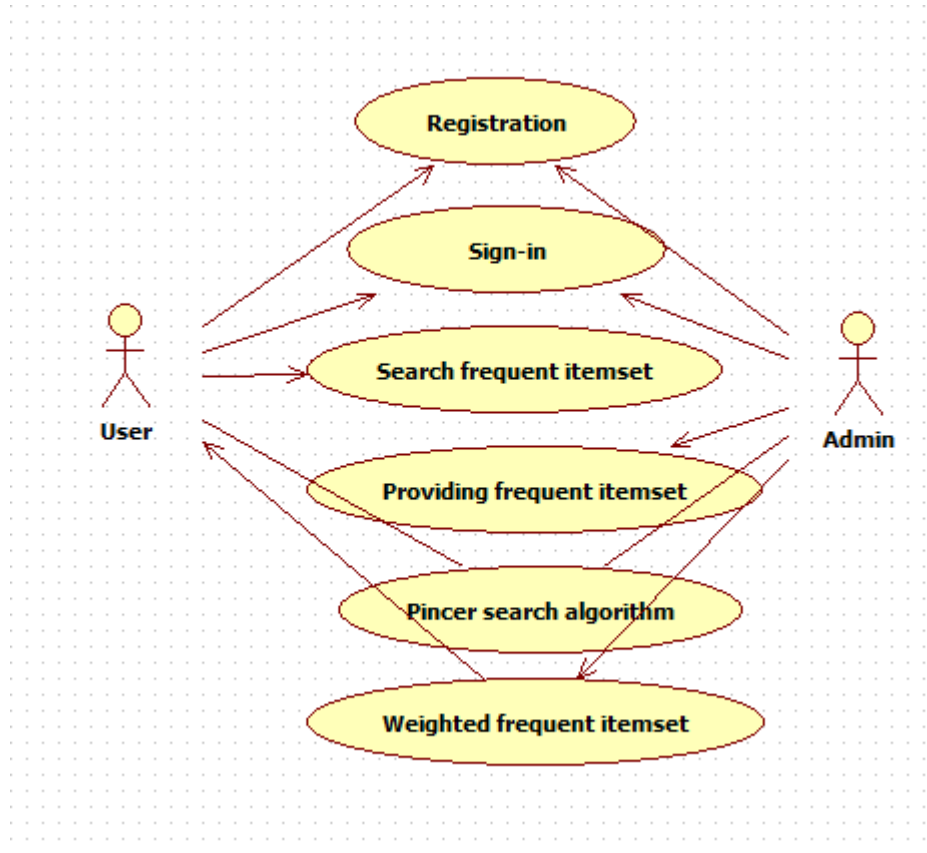


Fig 4.4.1: Use case diagram

Sequence Diagram

Sequence diagram model the flow of logic within your system in a visual manner, enabling you both to document and validate your logic, and commonly used for both analysis and design purpose. Sequence diagram are the most popular UML artifact for dynamic modeling, which focuses on identifying the behavior within your system.

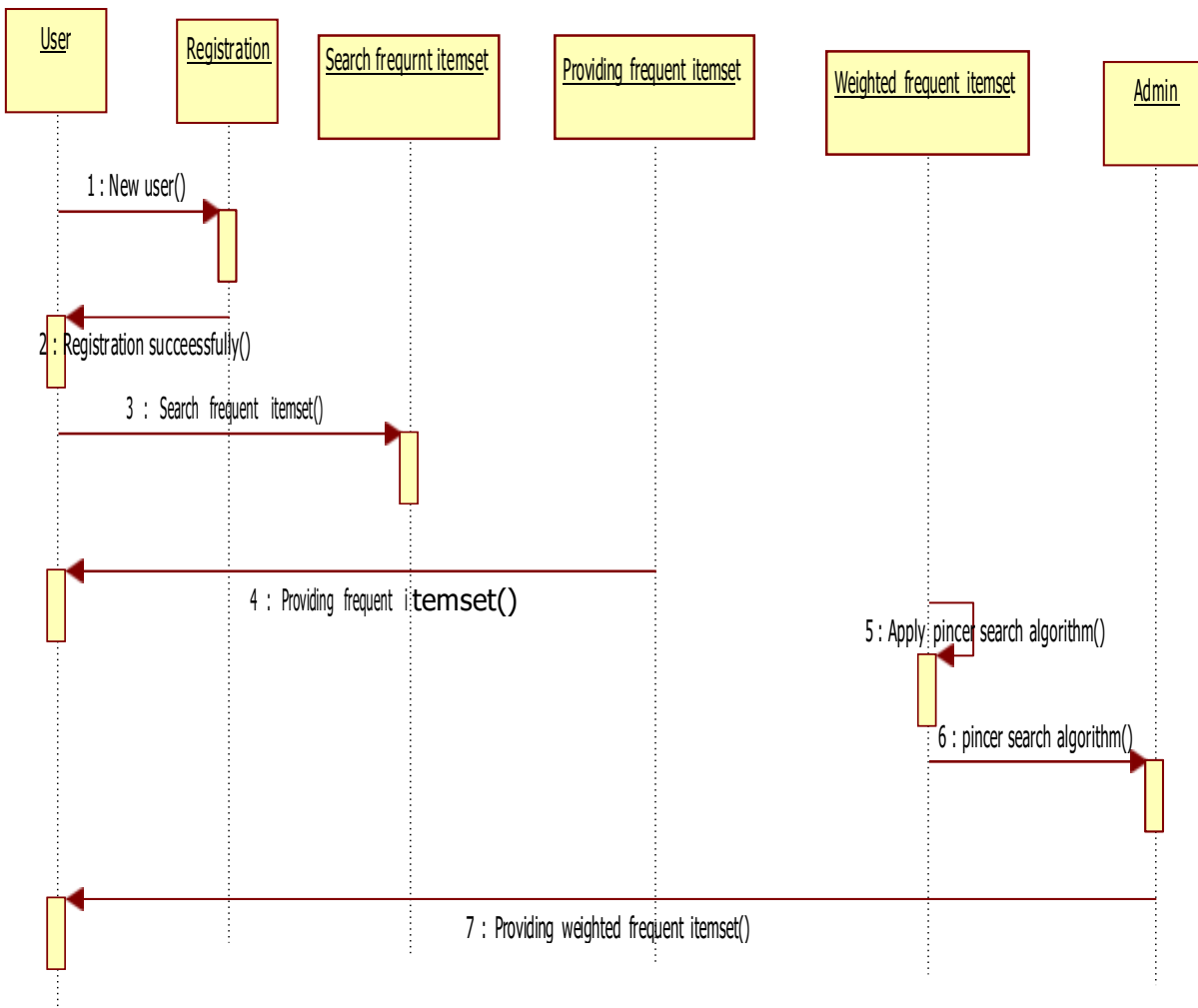


Fig 4.4.2: Sequence diagram

Collaboration Diagram

Another type of interaction diagram is the collaboration diagram. A collaboration diagram represents a collaboration, which is a set of objects related in a particular context, and interaction, which is a set of messages exchange among the objects

within the collaboration to achieve a desired outcome.

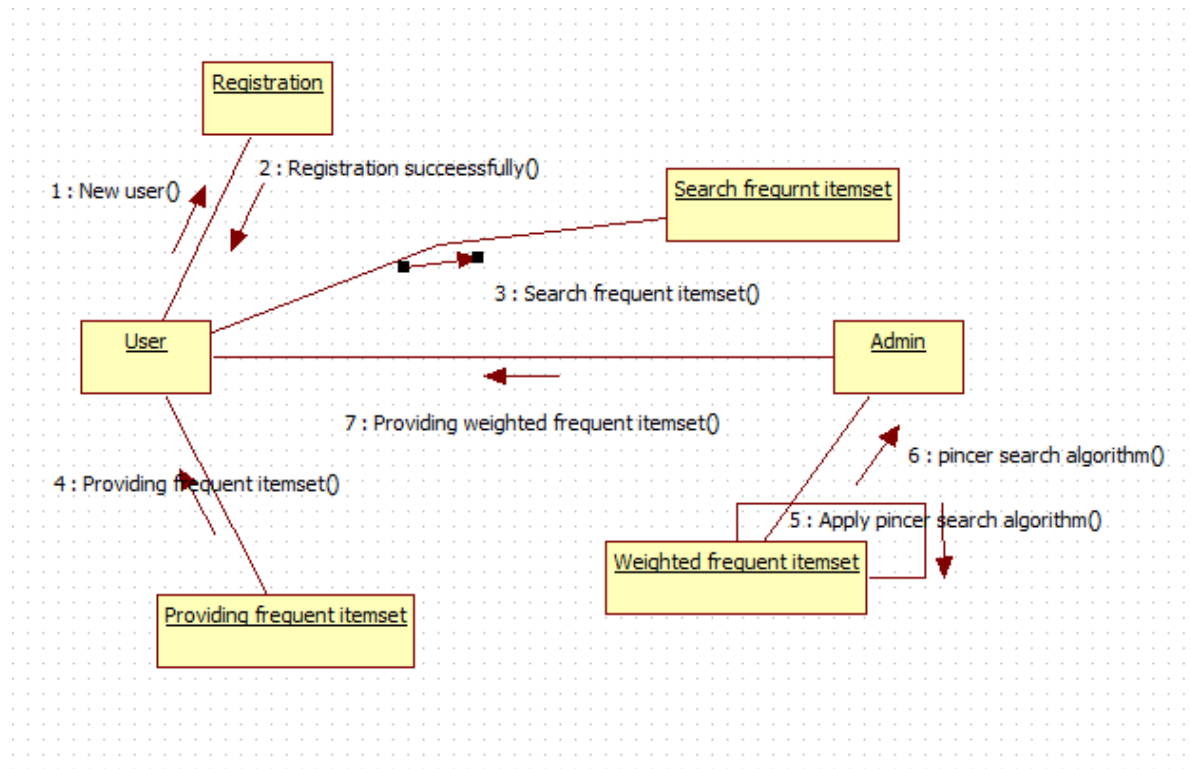


Fig 4.4.3: Collaboration diagram

ACTIVITY DIAGRAM

Activity diagram are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. The activity diagrams can be used to describe the business and operational step-by-step workflows of

components in a system. Activity diagram consist of Initial node, activity final node

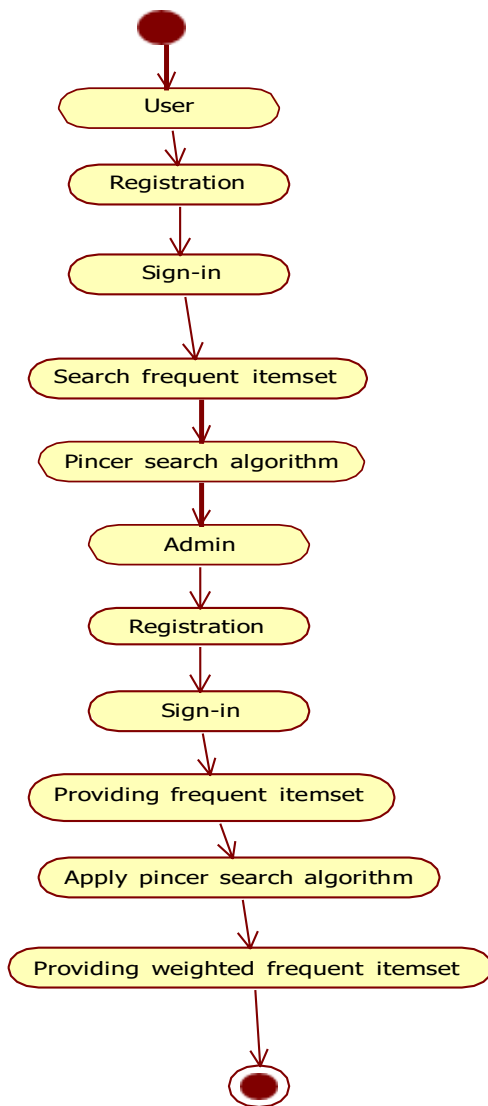


Fig 4.4.4: Activity Diagram

CHAPTER 5

CONCLUSION

In order to realize intelligent decision making in smart systems, a weight judgment downward closure property based frequent itemset mining algorithm is proposed in this

paper to narrow the searching space of weighted frequent item sets and improve the time efficiency. The weight judgment downward closure property for weighted frequent item sets and the existence property of weighted frequent subsets are introduced and proved first. Based on these two properties, the WD-FIM algorithm is described in detail. Moreover, the completeness and time efficiency of proposed Pincer-search algorithm are analyzed theoretically. Finally, the performance of the proposed Pincer-search algorithm is verified on both synthetic and real-life datasets.

CHAPTER-6

RESULTS AND DISCUSSION

This chapter provides a survey of different frequent pattern mining algorithms. most frequent pattern algorithms, implicitly or explicitly, explore the enumeration tree of item sets. Algorithms such as Apriori explore the enumeration tree in breadth-first fashion

with join-based candidate generation. Although the notion of an enumeration tree is not explicitly mentioned by the Apriori algorithm, the execution tree explores the candidates according to an enumeration tree constructed on the prefixes. Other algorithms such as Tree Projection and FP-growth use the hierarchical relationships between the projected databases for patterns of different lengths, and avoid re-doing the counting work done for the shorter patterns. Maximal and closed versions of frequent pattern mining algorithms are also able to achieve much better pruning performance. A number of efficiency-based optimizations of frequent pattern mining algorithms were also discussed in this chapter.

An efficient way to discover the maximum frequency set can be very useful in various data mining problems such as the discovery of the the association rules strong rules episodes and minimal keys. The maximum frequency provides unique representation of all the frequent itemsets. In many situations it suffices to discover the maximum frequency set and once it is known all the required frequencerz sets can be easily generated.

In this paper we presented new algorithm that can efficiently ee discover the maximum frequency set. Our pincer search algorithm can reduce the number of times the database is read and the the number of candidates considered. Experiments shown the improvement of using this approach can be very significant especially when some maximal frequent itemsets are long.

A popular assumption is that the maximal frequent itemsets are usually very short and therefore the computation of all frequent itemsets is feasible. sach assumption on maximal frequent itemsets does not need to be true in important applications. Consider for example the problem of discovering factors in price charges of individual stocks in stock market. Prices of individual stocks are frequently quiet correlated with each other. Therefore the discovered patterns may contain many items and the frequent items are long. Hear our algorithm could of great importance.

The performance of pincer search algorithm in applications such as discovering the episodes in sequences and discovering price changing patterns in stock markets will be studied. initial investigations indicate that maximal frequent itemsets in

many instances of such applications are likely to be long full stop before we expect the algorithm to provide dramatic performance improvements.

REFERENCES

Agarwal, a. & nanavati, n. Association rule mining using hybrid ga-pso For multi-objective optimisation. Ieee International conference on computational Intelligence and computing research (iccic), 2016. Ieee, 17.

AGRAWAL, R. & SRIKANT, R. Fast algorithms for mining association rules. Proc. 20th int. conf. very large data bases, VLDB, 1994. 487-499

ALTUNTAS, S., DERELI, T. & SELIM, H. 2013. Fuzzy weighted association rule based solution approaches to facility layout problem in cellular manufacturing system. International Journal of Industrial and Systems Engineering, 15, 253-271.

AZADNIA, A. H., TAHERI, S., GHADIMI, P., MAT SAMAN, M. Z. & WONG, K. Y. 2013. Order batching in warehouses by minimizing total tardiness: a hybrid approach of weighted association rule mining and genetic algorithms. The Scientific World Journal, 2013.

BANDYOPADHYAY, S., MALLIK, S. & MUKHOPADHYAY, A. 2014. A survey and comparative study of statistical tests for identifying differential expression from microarray data. IEEE/ACM transactions on computational biology and bioinformatics, 11, 95-115.

BANSAL, M., GROVER, D. & SHARMA, D. 2017. Sensitivity Association Rule Mining using Weight based Fuzzy Logic. Global Journal of Enterprise Information System, 9, 1-9.

C. C. Aggarwal and P. S. Yu, "A Survey of Uncertain Data Algorithms and Applications," IEEE Transactions on Knowledge & Data Engineering, vol.21, no. 5, pp. 609-623, May, 2009

APPENDICES

A.SOURCE CODE

package Mining;

```

import java.io.BufferedReader;

import java.io.File;

import java.io.FileNotFoundException;

import java.io.FileReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.HashSet;

import java.util.List;

import java.util.Map;

import java.util.Map.Entry;

import java.util.Set;


public class FreMining {


    static HashMap<String, Integer> supportCalc = new HashMap<String,Integer>();

    static List<List<String>> lists = new ArrayList<List<String>>();

    static HashMap<String, Integer> map =new HashMap<String,Integer>();

    static HashMap<String, Integer> twoMap = new HashMap<String,Integer>();


    public static void main ( String args[]) throws IOException

    {

```

```

try
{
    File f = null;

    BufferedReader systemInput = new BufferedReader(new
InputStreamReader(System.in));

    System.out.println("Association Rule Mining");

    System.out.println("~~~~~");

    System.out.println("Choose DataBase");

    System.out.println("1.Shoprite");

    System.out.println("2.Walmart");

    System.out.println("3.Amazon");

    System.out.println("4.Nike");

    System.out.println("5.Kmart");

    System.out.println("Enter your option : ");

    int option = Integer.parseInt(systemInput.readLine());

    switch (option) {

    case 1:

        f = new File("shoprite.txt");

        break;

    case 2:

        f = new File("walmart.txt");

        break;

```

case 3:

```
f = new File("amazon.txt");
```

```
break;
```

case 4:

```
f = new File("nike.txt");
```

```
break;
```

case 5:

```
f = new File("kmart.txt");
```

```
break;
```

default:

```
break;
```

```
}
```

```
int transactions = readFile(f);
```

```
//System.out.println("Enter Minimum Support Value in percentage %: ");
```

```
int minSup= Integer.parseInt("25");
```

```
//System.out.println("Enter Minimum Confidence Value in percentage %: ");
```

```
int minConf = Integer.parseInt("30");
```

```
int supVal = removeLowSupportVal(minSup,transactions);
```

```
secondIteration();
```

```
thirdIteration(supVal);
```

```
}
```

```

        catch (NumberFormatException e)
        {
            System.out.println("Please enter a valid number");

        }
        catch (FileNotFoundException e) {
            System.out.println("File not found in the path");
        }
        catch (NullPointerException e) {
            // TODO: handle exception
            System.out.println("Please enter a number between 1 to 5");
        }
    }
}

```

```

private static void thirdIteration(int supVal) {
    HashMap<String, Integer> tempMap = new HashMap<String,Integer>();
    ArrayList<String> tempList = new ArrayList<String>();
    tempMap = map;
    for(Entry<String, Integer> entry :tempMap.entrySet() )
    {
        int checker=0;
        checker=entry.getValue();
    }
}

```

```

        if(checker<supVal)

            tempList.add(entry.getKey());

    }

    for(String loop:tempList)

    {

        map.remove(loop);

    }


    System.out.println("Second Database Scan (2nd Iteration)");

    System.out.println("~~~~~");

    System.out.println();

    System.out.println("Generating Frequent Item sets...");

    ArrayList<String> tempSet = new ArrayList<>();

    for(String keys : map.keySet())

    {

        tempSet.add(keys);

    }

    System.out.println();

    System.out.println("Freq.Set          Support Value");

    System.out.println("~~~~~          ~~~~~");

    for (int b=0;b<tempSet.size();b++)

    {

        System.out.println(tempSet.get(b)+" -----> "+map.get(tempSet.get(b)));
    }

```

```

}

System.out.println();

    for(int i =0; i < tempSet.size(); i ++)
    {
        for(int j =i+1; j < tempSet.size();j++)
        {
            String name = tempSet.get(i)+","+tempSet.get(j);
            name=name.trim();

            ArrayList<String> t = new ArrayList<>();

            String[] enum1 = tempSet.get(i).split(",");
            String[] enum2= tempSet.get(j).split(",");

            for (String f : enum1)
            {
                t.add(f);
            }

            for(String g : enum2)
            {
                t.add(g);
            }

            buildThirdMap(t,name);
        }
    }
}

```

```

        HashMap<String, Integer> randMap = new HashMap<String,Integer>();

        ArrayList<String> randList = new ArrayList<String>();

        randMap = twoMap;

        for(Entry<String, Integer> entry :randMap.entrySet() )
        {
            int checker=0;

            checker=entry.getValue();

            if(checker<supVal)

                randList.add(entry.getKey());
        }

        for(String loop:randList)
        {
            twoMap.remove(loop);
        }

        removeDuplicates(twoMap,randList);
    }

    private static void removeDuplicates(HashMap<String, Integer>
twoMap2,ArrayList<String> randList2)
    {

```

```

ArrayList<String> dup = new ArrayList<>();

ArrayList<String> finalList=new ArrayList<>();


for(String keys : twoMap2.keySet())
{
    dup.add(keys);
}


for (int i=0;i<dup.size();i++)
{
    boolean duplicate = false;

    String[] splitter = dup.get(i).split(",");

    for (int j = 0 ; j < splitter.length ; j++)
    {
        for(int k = j+1 ; k < splitter.length ; k ++)
        {
            if ((splitter[j].trim()).equals(splitter[k].trim()))
            {
                duplicate = true ;
            }
        }
    }
}

```

```

        }

        if (duplicate==false)

        {

            finalList.add(dup.get(i));

        }

    }

    printoutFinal(finalList);

    System.out.println();

    //System.out.println("Exiting.. BYE");

}

private static void printoutFinal(ArrayList<String> finalList)

{

    if (finalList.isEmpty()) {

        System.out.println("Third DataBase Scan (3rd Iteration)");

        System.out.println("~~~~~");

        System.out.println();

        System.out.println("Generating Frequent Item sets...");

        System.out.println();

        System.out.println("Freq. Set           Support Value");

        System.out.println("~~~~~");
~~~~~");

```

```

ArrayList<String> g = new ArrayList<>();

Set<String> tempSet = new HashSet<>();

for (String key : twoMap.keySet())
{
    g.add(key);
}

for(int h = 0 ; h < g.size(); h ++)
{
    Set<String> st = new HashSet<>();

    String[] sp = g.get(h).split(",");

    for (int e =0; e < sp.length ; e ++)
    {
        st.add(sp[e].trim());
    }

    if (!(tempSet.containsAll(st)))
    {
        System.out.println(st+"----->
"+twoMap.get(g.get(h)));
    }

    tempSet.addAll(st);
}

System.out.println();

System.out.println("No more association rules can be generated");

```

```

    }

    else

    {

        System.out.println("Final DataBase Scan");

        System.out.println("~~~~~");

        System.out.println();

        System.out.println("Generating Frequent Item sets...");

        System.out.println();

        System.out.println("Freq. Set          Support Value");

        System.out.println("~~~~~          ~~~~~");

        for (int i = 0 ; i < finalList.size(); i++)

        {

            System.out.println(finalList.get(i)+" ----->

"+twoMap.get(finalList.get(i)));

        }

        System.out.println();

        System.out.println("No more Association rules can be Generated.");

    }

}

```

```

private static void buildThirdMap(ArrayList<String> t, String name)

{

    for (int x = 0 ; x < lists.size();x++ )

```

```

{
    List<String> temp = new ArrayList<String>();
    temp = lists.get(x);
    String[] enum3 = temp.get(0).split(",");
    ArrayList<String> rand = new ArrayList<String>();
    for(String l : enum3)
    {
        rand.add(l);
    }
    if(rand.containsAll(t))
    {
        if (!twoMap.containsKey(name))
        {
            twoMap.put(name,1);
        }
        else
        {
            twoMap.put(name,twoMap.get(name)+1);
        }
    }
}
}

```

```

private static void secondIteration() {

    ArrayList<String> freqSet = new ArrayList<String>();

    for(String findSz: supportCalc.keySet())
    {
        freqSet.add(findSz);
    }

    for(int i = 0; i < freqSet.size() ; i ++)
    {
        for (int j= i+1 ; j < freqSet.size() ; j++)
        {
            String str = freqSet.get(i)+" "+freqSet.get(j);
            buildMap(freqSet,i,j,str);
        }
    }
}

private static void buildMap(ArrayList<String> set, int a , int b, String str)
{
    List<String> tempList =new ArrayList<String>();

    tempList.add(set.get(a).trim());

    tempList.add(set.get(b).trim());

    str=str.trim();
}

```

```

for(int i =0 ; i< lists.size() ; i++)
{
    List<String> build = new ArrayList<String>();

    build = lists.get(i);

    if((build.get(0).contains(set.get(a)) &&
build.get(0).contains(set.get(b))))
    {
        if(!map.containsKey(str))
        {
            map.put(str,1);
        }
        else
        {
            map.put(str,map.get(str)+1);
        }
    }
}

private static int readFile(File file) throws IOException
{
    int transCounter =0;

    BufferedReader br = new BufferedReader(new FileReader(file));

    String letsRead = null;

```

```

while((letsRead = br.readLine())!=null)
{
    String[] transactionsArray = letsRead.split(":",-1);

    String[] itemsArray = transactionsArray[1].split("",-1);

    ArrayList<String> tempList = new ArrayList<String>();

    tempList.add(transactionsArray[1]);

    lists.add(tempList);

    for(String temp: itemsArray)
    {
        if(!supportCalc.containsKey(temp))

            supportCalc.put(temp, 1);

        else

            supportCalc.put(temp,supportCalc.get(temp)+1);

    }

    transCounter ++;

}

br.close();

return transCounter;

}

private static int removeLowSupportVal(int minSup, int transCounter) throws
NumberFormatException, IOException

{

    Map<String, Integer> map = new HashMap<String, Integer>();

```

```

        ArrayList<String> ar = new ArrayList<>();

        double offset =0;

        offset = (minSup *.01)*transCounter;

int roundOff= (int) (offset+0.5);

map=supportCalc;

for(Entry<String, Integer> entry :map.entrySet() )

{

    int checker=0;

    checker=entry.getValue();

    if(checker<roundOff)

        ar.add(entry.getKey());

}

System.out.println("Generating All Association Rules...");

System.out.println();

System.out.println("First Database Scan (1st Iteration)");

System.out.println("~~~~~");

System.out.println();

System.out.println("Generating Frequent Item sets...");

for(String loop:ar)

{

    supportCalc.remove(loop);

}

ArrayList<String> iterate = new ArrayList<>();

```

```

for(String sr : supportCalc.keySet())
{
    iterate.add(sr);
}

System.out.println();

System.out.println("Freq. Set      Support Value");

System.out.println("~~~~~      ~~~~~");

for (int b=0;b<iterate.size();b++)
{
    System.out.println(iterate.get(b)+" ----- >
"+supportCalc.get(iterate.get(b)));
}

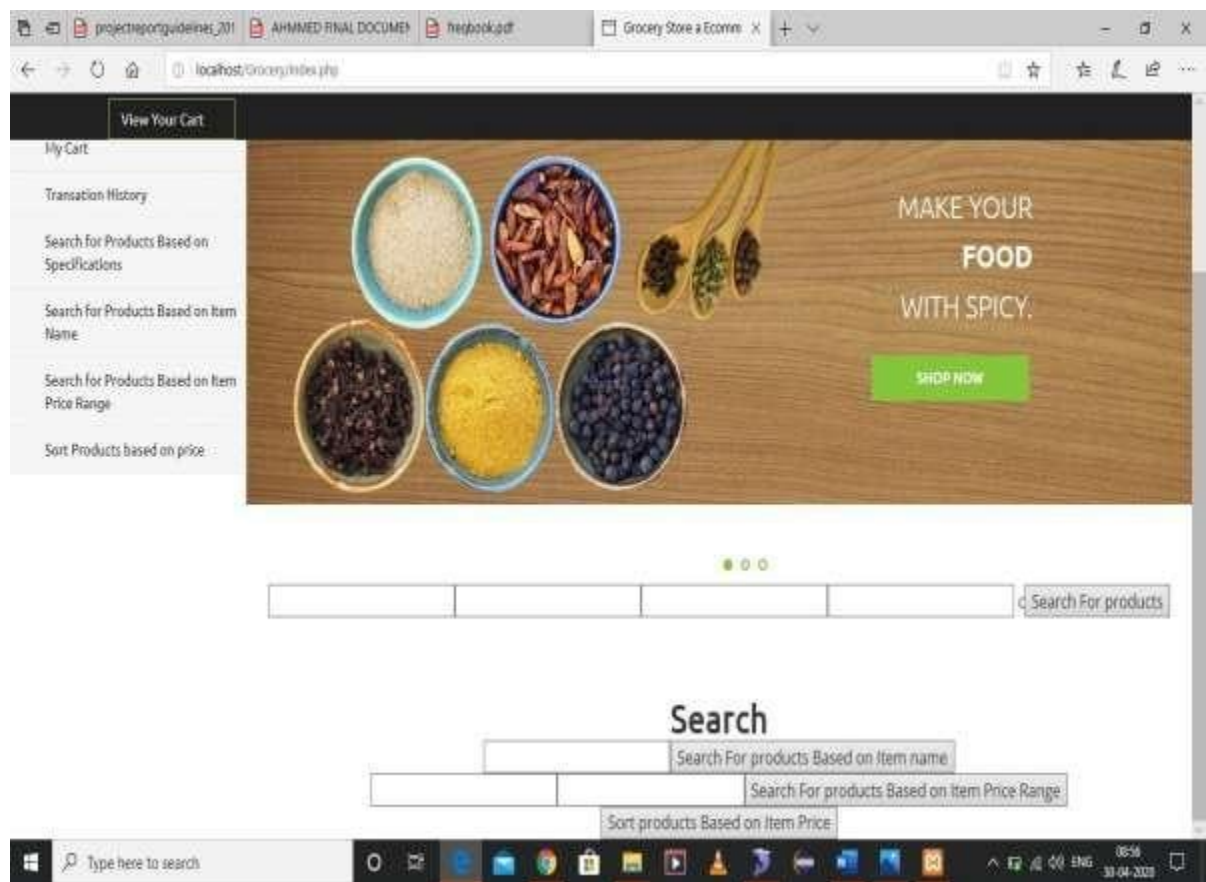
System.out.println("Association rules eliminated in this iteration are : "+ar );

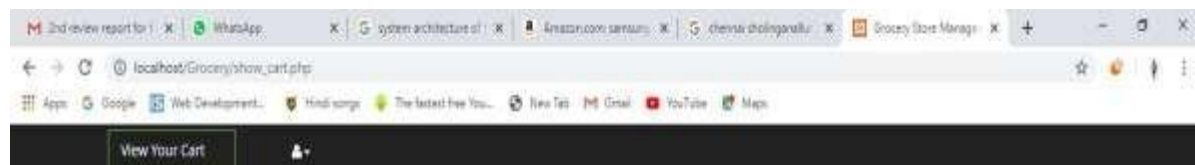
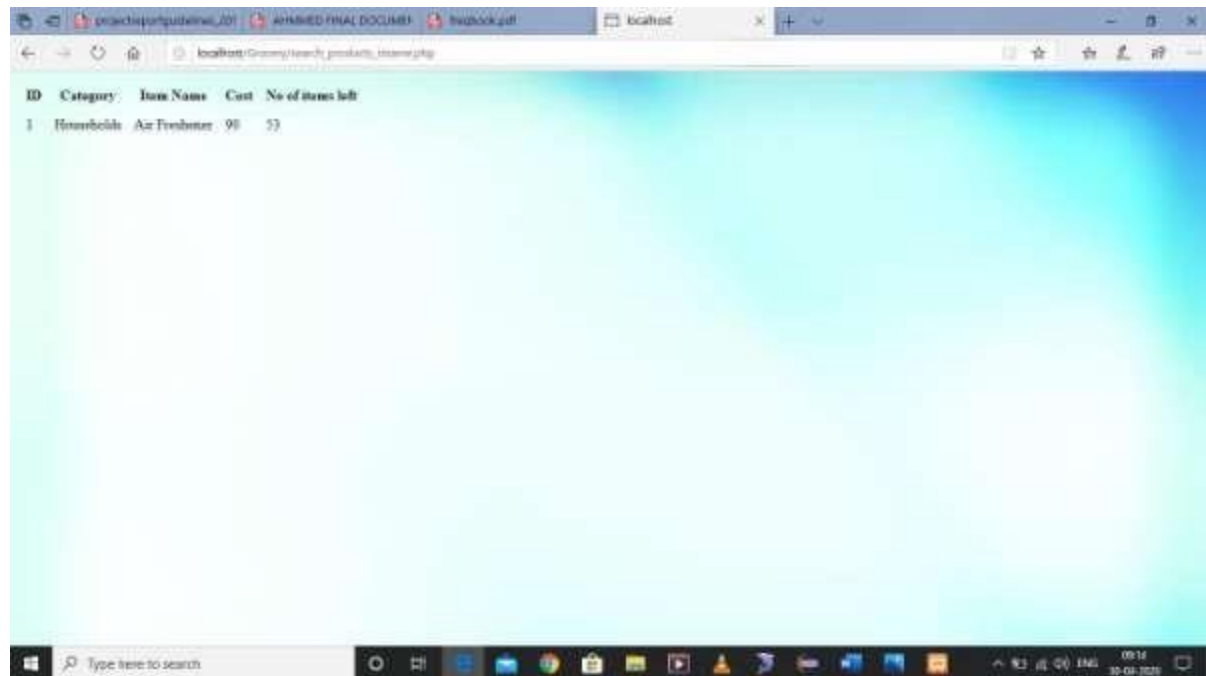
System.out.println();

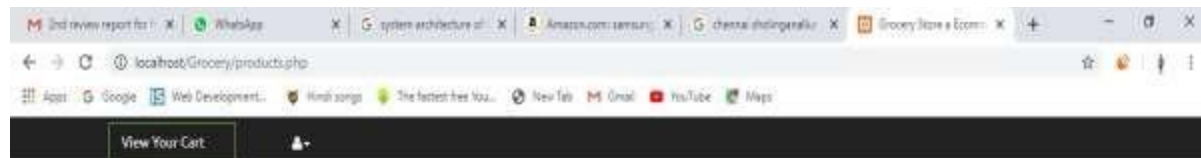
return roundOff;
}
}

```

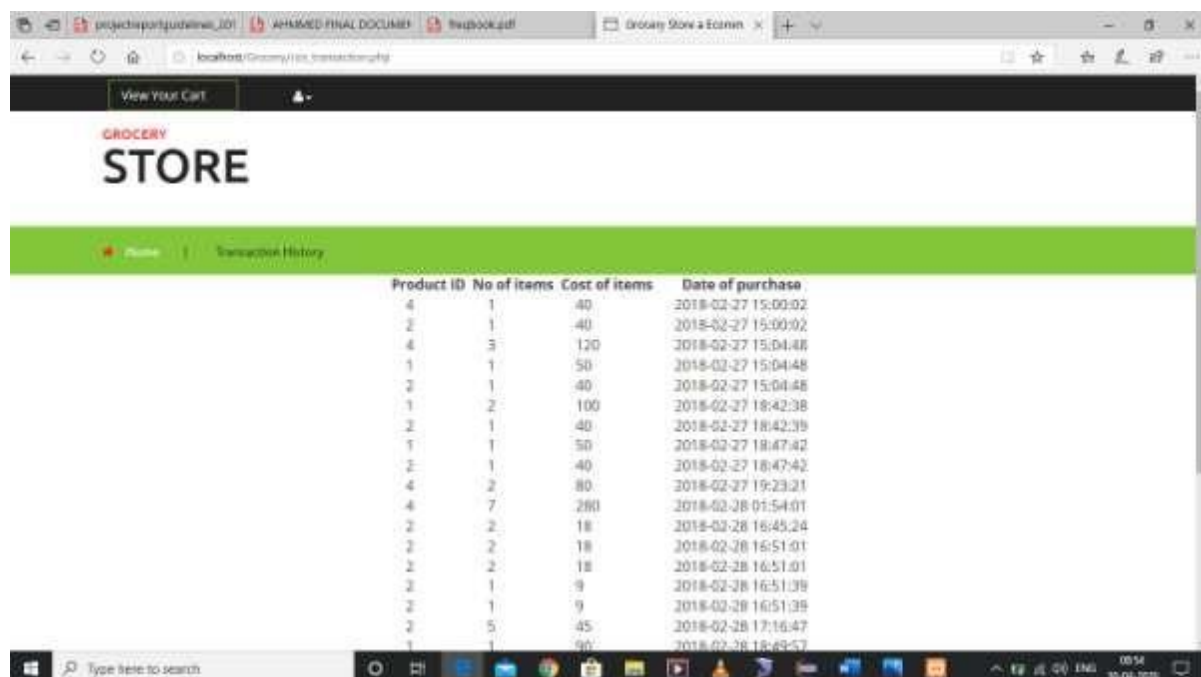
B.SCREENSHOTS







Product ID	Category	Item_name	Cost of each item	No_of_items	cart
1	Households	Air Freshener	90	55	ADD
2	Kitchen	Sugar	9	37	ADD
4	thiof	erfrweg	9	0	ADD



C)PUBLICATION PAPER

A WEIGHTED ALGORITHM FOR FREQUENT ITEM SET DATA MINING

Shaga Sai Teja^[1], Yalamarathi Satwik^[2], Senduru Srinivasulu^[3]

[1][2] UG Student, Dept. of Computer Science, Sathyabama University, Chennai, India [3] Assistant professor, Dept. of Computer Science, Sathyabama University, Chennai, India

ABSTRACT: - *The key innovation is the smart consumer counsel Mining frameworks in data innovation plays an inexorably a significant job in the activities and dynamic. Likewise basic Mining is a significant advance research properties of affiliation rules. It is amongst significant zones of analysis in data mining. Database to learn by frequent occupations in issues of uncertainty, the weighted they are so prone to be significant, and the components into frequently discovered extraordinary item sets clients. However, the presentation of weight makes frequent itemsets may fall back to the walled in area is to be an issue longer. Henceforth much of the time utilized things out there, not to look through they are what are alluded to the poor productivity. Said weight preliminary recurrence bolted to its own gravity and thing set sit is a genuine measure of material presented and tried. From these two properties WD-FIM. (possess weight and the finish of the preliminary Evaluation of the calculation is set before us by frequent, rehashed the space of the essence of the frequent itemsets, and improves the obstacle weighted with stones then proficiency. We proposed the elective strategy for Pincer-search calculation yields greatest frequent things, affiliation rules, precision and least execution time.*

INTRODUCTION

Data mining refers to the way in which useful data from databases, databases, or other databases can be extracted. It is dynamic in the scholarly world and industry [1]. With the advancement of society, the advancement of science and innovation and the improvement of human's personal satisfaction prerequisites, the measure of data created by human creation exercises is dangerously expanding, which partly gives all the more importance to data mining and gives it progressively significant hugeness. Frequent thing set mining is among the exploration headings of data mining. It attempts to discover the connection among certain and intriguing data with regards to huge data collection [2]. As of late, the utilization of mining of data has become increasingly broad. The examination on the exhibition of car wheel set [3], PC organize security [4], and the investigation of the nature of open credit

resources in business banks [5] have been broadly utilized. In the field of data mining, the most old style calculations are Apriori calculation [6] and FP-development algorithm[7]. They can locate all frequent thing set in an enormous number of exchange data, yet they all have their deficiencies. Apriori calculation is one of the most powerful data mining calculations. It utilizes the iterative technique for layer-by-layer search to discover frequent thing set and mining affiliation governs in one-dimensional and single-level width-first calculation. For Apriori calculation, it needs to filter the exchange database commonly. The info and yield cost of the framework is high, and countless competitor thing sets are created. It needs to mine successive thing set through example coordinating. The calculation is likewise huge. The effectiveness of memory use is generally low, which at last influences the productivity of mining. For FP-development

calculation, it is important to build a successive example tree FP-tree. At the point when the database data develops exponentially, the FP-tree will turn out to be extremely enormous, and frequent recursive conjuring of FP-tree will consume more space, even lead to the calculation not to be executed.

RELATED WORK

Mining design development. One way to my successive items is to use the applicant to create and test a world view from uncertain data. For example, the proposed U-Apriori [6] calculation, applied by Chui et coll. in order to digest frequent item sets of questionable data by the up-and-comer product and test procedure. Like Apriori calculation for exploring exact data, U-Apriori calculation requires to filter the information every now along with creating an enormous number of up-and-comer frequent item sets. *Chui et al [1]* used the decremental pruning method to additionally enhance the effectiveness of U-Apriori. *Tong et al. [12]* introduced a neighbourhood limit depending on deterministic calculation and a sketch-based inspecting inexact calculation to follow conveyed frequently probabilistic stuff.

Manjhi et al. [14] introduced a technique to discover common things in association of various disseminated flow of information. Nonetheless, the strategy is introduced to the instance of the question and the correspondence design is confined to adjusted, including normal trees, the correspondence design has been extended from one-level mode to an overlapping prototype. The significant downside is its heavy correspondence expense, that implies the absolute mistake resilience should be sufficiently huge to keep little variances of activating synopsis revives. *Olston et al [14]* planned a few heuristics for checking the top-k esteems over local information channels. This methodology

can be adjusted to constantly follow the frequent things over local information channels [23], [24]. The two basic drawbacks: one, this technique stays interesting in its way that implies it absences of a hypothetical examination; The two basic drawbacks: one, this technique stays interesting in its way that implies it absences of a hypothetical examination; second, just the single-level correspondence structure is considered. *Zhang et al [12]* introduced a acceptance calculation for ceaselessly following common things of local information channels amongst pessimistic scenario correspondence expense $O(k \log(n))$, anyway the creators just take the primary-level correspondence design MBP [8] is an estimate strategy for questionable successive example mining dependent on factual strategies. IMBP

[9] had been introduced to enhance the extracting velocity and storage productivity of MBP of expense at reducing exactness. The option in contrast to up-and-comer produce and- test depended extracting is design development extracting, that abstains from creating countless competitors [10].

Ordinarily utilized example development mining ideal models are for the most part dependent on related designs or tree designs. An instance, *Aggarwal et al. [2]* introduced a related design focused calculation called UH-mine to mine common examples among questionable information. Leung et al. introduced a tree-based extracting calculation called UF-development that additionally builds a tree design to save the substance of the dubious sets of data, similar to the partner - the FP-development calculation [13] for extracting exact information. So as to decrease the tree dimension, *Aggarwal et al. [5]* introduced the UFP-development calculation. For additionally decreasing the tree dimension, *Leung et al [7]* introduced a questionable common example

extracting calculation called CUF growth, that fabricates another tree design called CUF-tree. **Leung et al [8]** presented the idea of a prefixed thing top and proposed PUF-development calculation to extract questionable common examples that works quicker than CUF-development. TPC-development [17] is a propelled rendition of PUF-development. It utilizes an updated excess prediction technique that fix upper limits to estimated backings more than PUF-development.

CUFP-Mine [18] is a technique for mining accurate dubious successive examples without utilizing recursive call-based example development habits. Be that as it may, the bigger the data info is, the more regrettable the extracting execution of CUFP-Mine becomes. AT-Mine [19] is a tree-focused effective methodology introduced to defeat the lethal issues of CUFP-Mine. That ensures excess proficient extracting execution of CUFP-Mine, however that despite everything has constraints in compile-time and storage execution angles. U-WFI [20] is a tree-focused methodology that implies mass aspects for unsure example mining. By mass imperatives, the calculation will discover increasingly significant questionable frequent examples however have constraints in the previously mentioned angles. **Keralapura et al. [15]** considered the "edge tallies" issue where restoring the total recurrence checks of the things that are constantly observed by dispersed hubs at whatever point the real checks surpass a given edge esteem. Notwithstanding, the creators just take the primary-level correspondence design. In synopsis, no previous works have introduced correspondence proficient calculations for ceaselessly following the worldwide successive things over the disseminated mass information channels in the

staggered, non-customary tree-focused correspondence design.

EXISTING SYSTEM

A hyper-related design focused calculation called UH-mine to extract common examples from dubious data. a tree-based extracting calculation called UF-development that likewise builds a tree design to save the substance of the questionable information, similar to its partner - the FP-development calculation for extracting exact data. So as to decrease the tree dimension at proposed the UFP-development calculation. To additionally lessen the tree size. An uncertain frequent design mining calculation called CUF growth, which builds another tree structure called CUF-tree. AT-Mine is a tree-based effective methodology introduced to conquer the deadly issues of CUFP-extract tree.

3.1 DISADVANTAGES OF EXISTING SYSTEM

In the WDFIM method to using to get low amount of mining itemsets.

Low mining efficiency and coverage

PROPOSED SYSTEM

We present a Pincer-look calculation for proficient mining of the successive itemsets. The prefiltering strategy is applied to the data dataset to expel the thing having low difference. Data discretization is performed and Pincer-look is applied for mining the frequent itemsets. The proposed Pincer-search calculation yields most extreme frequent things, affiliation rules, precision and least execution time than the current calculations are based on the figure below.

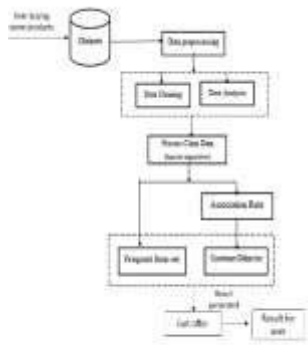


FIG 1 SUMMARY OF THE SUGGESTED SYSTEM

BENEFITS OF SUGGESTED SYSTEM

Prediction accuracy is improved by using the association rules.

Effective generation of the frequent itemsets and association rules is ensured by maintaining the feasibility of the neural network.

T-id	Items Bought	Frequent Items(ordered)
101	W,t,u,g,s,d,f	u,s,d,w
102	h,a,l,e,l,d	d,e,i
103	J,q,o,f,w	o,q
104	k,d,z,v,f	d,f,k
105	e,u,y,f,o,p	y,f,e

Table 1: A transaction database as running example

Graph:

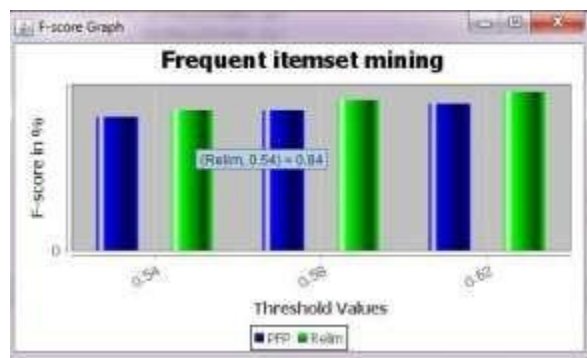


FIG 2 F-SCORE IN % ON DATASET

ELEMENT ELUCIDATION

REGISTER ELEMENT

This is for registering the login Id of the user by providing limited information. Then you can access the site.

SIGN IN ELEMENT

This allows the user to log in via registered usage and a valid password to the website. Only the user who is authorised may access the website and use it.

USER ELEMENT

The user extracts in this element items, which are often combined with a higher overall rating.

ADMIN ELEMENT

Admin checks the item's list, adds items and deletes items that are not requested in this element.

FREQUENT DATASET ELEMENT

We introduce an Pincer-search method for systematic extraction of the common things. The profiteering method is applied to the input dataset to remove the item having low variance.

I CONCLUSION

To accept keen dynamic of savvy structures, a weight judgment descending conclusion property based frequent itemset mining calculation is proposed right now slender the looking through space of weighted frequent itemsets and improve the time productivity. The weight judgment descending conclusion property for weighted frequent itemsets and the presence property of weighted frequent subsets are presented and demonstrated first. In view of these two properties, the WD-FIM calculation is portrayed in detail. Also, the culmination and time proficiency of proposed Pincer-search

calculation are investigated hypothetically. At long last, the presentation of the proposed Pincer-search calculation is checked on both manufactured and genuine datasets.

REFERENCES:

- J. Pei, J. Han, and W. Wang. "Constraint- based Sequential Pattern Mining: The Pattern- Growth Methods," *Journal of Intelligent Data Systems*, vol. 28, no. 2, pp. 133–160, April 2007.
- C. C. Aggarwal, Y. Li, J. Wang, and J. Wang, "Frequent pattern mining with uncertain data," in *Proceedings of the ACM KDD 2009*, 2009, pp. 29-38.
- K. S. Leung, M. A. F. Mateo, and D. A. Brajczuk, "A tree-based approach for frequent pattern mining from uncertain data," in *Proceedings of the PAKDD 2008*, 2008, pp.653–661.
- 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*, 2000, pp.1–12.
- Bhagyashree R. Vhatkar ,Prof. Dr. Mrs.S.A. Itkar, Frequent Itemset mining based on Differential Privacy using RELim(DP-RElim)
- K. S. Leung and S. K. Tanbeer, "Fast tree- based mining of frequent itemsets from uncertain data," in *Proceedings of International Conference on Database systems for Advanced Applications*, 2012, pp 272-287
- [7] K. S. Leung and S. K. Tanbeer, "PUF-tree: a compact tree structure for frequent pattern mining of uncertain data," in *Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 2013, pp. 13–25.
- [8] C. K. Leung, R. K. Mackinnon, and S. K. Tanbeer, "Tightening upper bounds to the expected support for uncertain frequent pattern mining," in the *18th International Conference on Knowledge-Based and Intelligent Data & Engineering Systems*, 2014, pp. 328–337.
- [9] C. W. Lin and T. P. Hong, "A new mining approach for uncertain databases using CUPF trees," *Expert Systems with Applications*, vol.39, no. 4, pp. 4084–4093, March 2012.
- [10] L. Wang, L. Feng, and M. Wu, "AT-Mine: An efficient algorithm of frequent itemset mining on uncertain dataset," *Journal of Computers*, vol. 8, no. 6, pp.1417–1426, June 2016.
- [11] G. Lee, U. Yun, and H. Ryang, "An uncertainty-based approach: Frequent itemset mining from uncertain data with different item importance," *Knowledge-Based Systems*, vol. 90, pp. 239–256, Dec. 2015.
- [12] Y. Tong, X. Zhang, and L. Chen, "Tracking frequent items over distributed probabilistic data," *World Wide Web*, vol. 19, no. 4, pp. 579–604, Jul. 2016.
- [13] T. Qiu, R. Qiao, and D. O. Wu, "EABS: An event-aware backpressure scheduling scheme for emergency Internet of Things," *IEEE Trans. Mobile Comput.*, vol. 17, no. 1, pp. 72–84, Jan. 2018.
- [14] A. Manjhi, V. Shkapenyuk, K. Dhamdhere, and C. Olston, "Finding (recently) frequent items in distributed data streams," in *Proc. 21st IEEE Comput. Soc. Int. Conf. Data Eng. (ICDE)*, Washington, DC, USA, vol. 1, no. 21, Apr. 2005, pp. 767–778.
- [15] R. Keralapura, G. Cormode, and J. Ramamirtham, "Communication efficient distributed monitoring of thresholded counts," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, New York, NY, USA, 2006, pp. 289-300

Turnitin Originality Report

Processed on: 05-May-2020 6:02 AM EDT ID: 1316481120

Word Count: 2255

Submitted: 1

Similarity Index

8%

Similarity by Source

Internet Sources: 4%
Publications: 5%
Student Papers: 4%

06 By 06 06

4% match (Internet from 28-Oct-2019)

<http://ijsear.com/index.php/ijsear/article/download/1131/pdf>

2% match (student papers from 14-Mar-2020)

[Submitted to Indian Institute of Technology, Madras on 2020-03-14](#)

1% match (publications)

[Xuejian Zhao, Xinhui Zhang, Pan Wang, Songle Chen, Zhixin Sun. "A Weighted Frequent Itemset Mining Algorithm for Intelligent Decision in Smart Systems", IEEE Access, 2018](#)

1% match (publications)

[Leung, Carson Kai-Sang. "Uncertain Frequent Pattern Mining", Frequent Pattern Mining, 2014.](#)

< 1% match (publications)

[Xuejian Zhao, Xinhui Zhang, Pan Wang, Songle Chen, Zhixin Sun. "A Weighted Frequent Itemset Mining Algorithm for Intelligent Decision in Smart Systems", IEEE Access, 2018](#)

A WEIGHTED ALGORITHM FOR FREQUENT ITEM SET DATA MINING Shaga

Sai Teja[1], Yalamarthi Satwik[2], Senduru Srinivasulu [3] [1][2] UG

Student, Dept. of Computer Science, Sathyabama University, Chennai,

India [3] Assistant professor, Dept. of Computer Science, Sathyabama

University, Chennai, India ABSTRACT: - The key innovation is the smart

consumer counsel Mining frameworks in data innovation plays an

inexorably a significant job in the activities and dynamic. Likewise basic Mining is

a significant advance research properties of affiliation rules. It is amongst

significant zones of analysis in data mining. Database to learn by frequent

occupations in issues of uncertainty, the weighted they are so

prone to be significant, and the components into frequently discovered

extraordinary item sets clients. However, the presentation of weight makes

frequent itemsets may fall back to the walled in area is to be an issue longer.

Henceforth much of the time utilized things out there, not to look

through they are what are alluded to the poor productivity. Said weight

preliminary recurrence bolted to its own gravity and thing set sit is a

genuine measure of material presented and tried. From these two

properties WD-FIM. (possess weight and the finish of the preliminary

Evaluation of the calculation is set before us by frequent, rehashed the

space of the essence of the frequent itemsets, and improves the obstacle

weighted with stones then proficiency. We proposed the elective strategy for

Pincer-search calculation yields greatest frequent things, affiliation

rules, precision and least execution time. I INTRODUCTION Data mining

refers to the way in which useful data from databases, databases, or other

databases can be extracted. It is dynamic in the scholarly world and industry [1]. With the advancement of society, the advancement of science and innovation and the improvement of human's personal satisfaction prerequisites, the measure of data created by human creation exercises is dangerously expanding, which partly gives all the more importance to data mining and gives it progressively significant hugeness. Frequent thing set mining is among the exploration headings of data mining. It attempts to discover the connection among certain and intriguing data with regards to huge data collection [2]. As of late, the utilization of mining of data has become increasingly broad. The examination on the exhibition of car wheel set [3], PC organize security [4], and the investigation of the nature of open credit resources in business banks [5] have been broadly utilized. In the field of data mining, the most old style calculations are Apriori calculation [6] and FP-development algorithm[7]. They can locate all frequent thing set in an enormous number of exchange data, yet they all have their deficiencies. Apriori calculation is one of the most powerful data mining calculations. It utilizes the iterative technique for layer-by-layer search to discover frequent thing set and mining affiliation governs in one- dimensional and single-level width-first calculation. For Apriori calculation, it needs to filter the exchange database commonly. The info and yield cost of the framework is high, and countless competitor thing sets are created. It needs to mine successive thing set through example coordinating. The calculation is likewise huge. The effectiveness of memory use is generally low, which at last influences the productivity of mining. For FP-development calculation, it is important to build a successive example tree FP-tree. At the point when the database data develops exponentially, the FP- tree will turn out to be extremely enormous, and frequent recursive conjuring of FP-tree will consume more space, even lead to the calculation not to be executed.

II RELATED WORK

Mining design development.

One way to my successive items is to use the applicant to create and test a world view from uncertain data. For example, the proposed U-Apriori [6] calculation, applied by Chui et coll. in order to digest frequent item sets of questionable data by the up-and-comer product and test procedure. [Like](#) [Apriori](#) calculation [for](#) exploring [exact](#) data, [U-Apriori](#) calculation requires [to](#) filter [the information](#) every now along with creating an enormous number of up-and-comer frequent item sets. Chui et al [1] used [the](#) [decremental pruning](#) method [to](#) additionally enhance [the](#) effectiveness [of](#) [U-Apriori](#). Tong et al. [12] introduced a neighbourhood limit depending on deterministic calculation and a sketch-based inspecting inexact calculation to follow conveyed frequently probabilistic stuff. Manjhi et al. [14] introduced a technique to discover common things in association of various disseminated flow of information. Nonetheless, the strategy is introduced to the instance of the question and the correspondence design is confined to adjusted, including normal trees, the correspondence design has been extended from one-level mode to an overlapping prototype. The significant downside is its heavy correspondence expense, that implies the absolute mistake resilience should be sufficiently huge to keep little variances of activating synopsis revives. Olston et al [14] planned a few heuristics for checking the top-k esteems over local information channels. This methodology can be adjusted to constantly follow the frequent things over local information channels [23], [24]. The two basic drawbacks: one, this technique stays interesting in its way that implies it absences of a hypothetical examination; second, just the single-level correspondence structure is considered. Zhang et al [12] introduced a acceptance calculation for ceaselessly following common things of local information channels amongst pessimistic scenario correspondence expense $O(k \log(n))$, anyway the creators just take the primary-level correspondence design MBP [8] is an estimate strategy for questionable successive example mining dependent on factual strategies. IMBP [9] had been introduced to enhance the extracting velocity and storage productivity of MBP of expense at reducing exactness. The option in contrast to up-and-

comer produce and- test depended extracting is design development extracting, that abstains from creating countless competitors [10]. Ordinarily utilized example development mining ideal models are for the most part dependent on related designs or tree designs. An instance, Aggarwal et al. [2] introduced a related design focused calculation called UH-mine to mine common examples among questionable information.

[Leung et al.](#) introduced [a tree-based](#) extracting calculation [called UF-](#) development that additionally builds a tree design to save the substance of the dubious sets of data, similar to the partner - the FP- development calculation [13] for extracting exact information. So as to decrease the tree dimension, Aggarwal et al. [5] introduced the UFP-development calculation. For additionally decreasing the tree dimension, Leung et al [7] introduced a questionable common example extracting calculation called CUF growth, that fabricates another tree design called CUF-tree. Leung et al [8] presented the idea of a prefixed thing top and proposed PUF-development calculation to extract questionable common examples that works quicker than CUF- development. TPC-development [17] is a propelled rendition of PUF-development. It utilizes an updated excess prediction technique that fix upper limits to estimated backings [more than](#) PUF- development. [CUFP-Mine \[18\] is a technique for mining](#) accurate dubious successive examples without utilizing recursive call-based example development habits. Be that as it may, the bigger the data info is, the more regrettable the extracting execution of [CUFP-Mine becomes. AT-](#) [Mine \[19\] is a tree-](#) focused effective methodology introduced to defeat the lethal issues of CUFP-Mine. That ensures excess proficient extracting execution of CUFP-Mine, however that despite everything has constraints in compile-time and storage execution angles. U- WFI [20] is a tree- focused methodology that implies mass aspects for unsure example mining. By mass imperatives, the calculation will discover increasingly significant questionable frequent examples however have constraints in the previously mentioned angles. Keralapura et al. [15] considered the "edge tallies" issue where restoring the total recurrence checks of the things that are constantly observed by dispersed hubs at whatever point the real checks surpass a given edge esteem. Notwithstanding, the creators just take the primary-level correspondence design. In synopsis, no previous works have introduced correspondence proficient calculations for ceaselessly following the worldwide successive things over the disseminated mass information channels in the staggered, non-customary tree-focused correspondence design.

III EXISTING SYSTEM

A hyper-related design focused calculation called UH-mine to extract common examples from dubious data. a tree-based extracting calculation called UF- development that likewise builds a tree design to save the substance of the questionable information, similar to its partner - the FP-development calculation for extracting exact data. So as to decrease the tree dimension at proposed the UFP-development calculation. To additionally lessen the tree size. [An uncertain frequent](#) design [mining](#) calculation [called CUF growth, which builds](#) another [tree structure called CUF- tree](#). AT-Mine is a tree-based effective methodology introduced to conquer the deadly issues of CUFP-extract tree.

3.1 DISADVANTAGES OF EXISTING SYSTEM

- In the WDFIM method to using to get low amount of mining itemsets.
- Low mining efficiency and coverage

IV PROPOSED SYSTEM

We present a Pincer-look calculation for proficient mining of the successive itemsets. The prefiltering strategy is applied to the data dataset to expel the thing having low difference. Data discretization is performed and Pincer-look is applied for mining the frequent itemsets. The proposed Pincer-search calculation yields most extreme frequent things, affiliation rules, precision and least execution time than the current calculations.

FIG 1 SUMMARY

OF THE SUGGESTED SYSTEM

4.1 BENEFITS OF SUGGESTED SYSTEM

5.3 USER ELEMENT

The user extracts in this element items, which are often combined with a higher overall

- Prediction accuracy is improved by using rating. the association rules.
- Effective generation of the frequent

5.4

ADMIN ELEMENT itemsets and association rules is ensured by maintaining the feasibility of the neural Admin checks the item's list, adds items network. and deletes items that are not requested in this element. T-id Items Bought Frequent 5.5 FREQUENT DATASET ELEMENT Items(ordered) 101 W,t,u,g,s,d,f u,s,d,w We introduce an Pincer-search method 102 h,a,l,e,l,d d,e,i for systematic extraction of the common things. 103

J,q,o,f,w o,q The profiteering method is applied to the input 104 k,d,z,v,f d,f,k dataset to remove the item having low variance. 105 e,u,y,f,o,p y,f,e VI

CONCLUSION Table 1: A transaction database as running example To accept keen dynamic of savvy structures, a weight judgment descending Graph: conclusion property based frequent itemset mining calculation is proposed right now slender the looking through space of weighted

frequent itemsets and improve the time productivity. The weight judgment descending conclusion property for weighted frequent itemsets and the presence property of weighted frequent subsets are presented and demonstrated first. In view of these two properties, the WD-FIM calculation is portrayed in detail. Also, the culmination and time proficiency of proposed Pincer-search calculation are investigated hypothetically. At FIG 2 F-SCORE IN % ON DATASET long last, the presentation of the proposed Pincer- search calculation is checked on both V ELEMENT ELUCIDATION manufactured and genuine datasets.

REFERENCES: 5.1 REGISTER ELEMENT This is for registering the login Id of the

[1] J. Pei, J. Han, and W.Wang. "Constraint- user by providing limited information. Then you based Sequential Pattern Mining: The

Pattern- can access the site. Growth Methods," Journal of Intelligent Data Systems, vol. 28, no. 2, pp. 133–160, April 5.2 SIGN IN ELEMENT 2007.

This allows the user to log in via [2] C. C. Aggarwal, Y. Li, J. Wang, and J. registered usage and a valid password to the Wang, "Frequent pattern mining with uncertain website. Only the user who is authorised may data," in Proceedings of the ACM KDD 2009, access the website and use it. 2009, pp. 29-38. [3] K. S. Leung, M. A. F. Mateo, and D. A. Brajczuk, "A tree-

based approach for frequent pattern mining from uncertain data," in Proceedings of the PAKDD 2008, 2008, pp.653–661. [4] 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, 2000, pp.1–12. [5] BhagyashreeR.Vhatkar ,Prof. Dr. Mrs.S.A.Itkar, Frequent Itemset

mining based on Differential Privacy using RELim(DP- RELim) [6] K. S. Leung and S. K. Tanbeer, "Fast tree- based mining of frequent itemsets from uncertain data," in

Proceedings of International Conference on Database systems for Advanced

Applications,2012,pp 272-287 [7] K. S. Leung and S. K. Tanbeer, "PUF- tree: a compact tree structure for frequent pattern mining of uncertain data," in

Proceedings of the Pacific-Asia Conference on Knowledge

Discovery and Data Mining, 2013,pp. 13–25. [8] C. K. Leung, R. K. Mackinnon, and S. K. Tanbeer, "Tightening upper bounds to the expected support for uncertain

frequent pattern mining," in the 18th International Conference on Knowledge-Based and Intelligent Data & Engineering

Systems, 2014, pp. 328–337. [9] C. W. Lin and T. P. Hong, "A new mining approach for uncertain databases using CUPF trees," Expert Systems with Applications,

vol.39, no. 4, pp. 4084–4093, March 2012. [10] L. Wang, L. Feng, and M. Wu, "AT-Mine: An efficient algorithm of frequent itemset mining on uncertain dataset,"

Journal of Computers, vol. 8, no. 6,

pp.1417–1426, June 2016. [11] G. Lee, U. Yun, and H. Ryang, "An uncertainty-based approach: Frequent itemset mining from uncertain data with different item importance," Knowledge-Based Systems, vol. 90, pp.

239–256, Dec. 2015. [12] Y. Tong, X. Zhang, and L. Chen, "Tracking frequent items over distributed probabilistic data," World Wide Web, vol. 19, no. 4, pp. 579–

604, Jul. 2016. [13] T. Qiu, R. Qiao, and D. O. Wu,

"EABS: An event-aware backpressure scheduling scheme for emergency Internet of Things," IEEE Trans. Mobile Comput., vol. 17, no. 1, pp. 72–84, Jan. 2018. [14] A.

Manjhi, V. Shkapenyuk, K. Dhamdhere, and C. Olston,

"Finding (recently) frequent items in distributed data streams," in Proc. 21st IEEE Comput. Soc. Int. Conf. Data Eng. (ICDE), Washington, DC, USA, vol. 1, no. 21, Apr. 2005, pp. 767–778. [15] R. Keralapura, G. Cormode, and J. Ramamirtham, "Communication efficient distributed monitoring of thresholded counts," in Proc. ACM SIGMOD Int. Conf.

Manage. Data, New York, NY, USA, 2006, pp. 289–300

