VIETNAM GENERAL CONFEDERATION OF LABOR

**TON ĐUC THANG UNIVERSITY**

**FACULTY OF INFORMATION TECHNOLOGY**

****

**FINAL REPORT OF DESIGN AND ANALYSIS ALGORITHMS**

**A SURVEY OF ITEMSET MINING REPORT**

Instructor:**MASTER.NGUYỄN CHÍ THIỆN**

Student: **PHẠM PHƯỚC TẤN– 520H0418**

**NGUYỄN HOÀNG PHÚC KHANG– 520H0066**

**TRẦN LÊ GIA BẢO– 520H0516**

Class **: 20H50204**

Course  **: 24**

**HO CHI MINH CITY, 2022**

VIETNAM GENERAL CONFEDERATION OF LABOR

**TON ĐUC THANG UNIVERSITY**

**FACULTY OF INFORMATION TECHNOLOGY**

****

**FINAL REPORT OF DESIGN AND ANALYSIS ALGORITHMS**

**A SURVEY OF ITEMSET MINING REPORT**

Instructor: **MASTER.NGUYỄN CHÍ THIỆN**

Student: **PHẠM PHƯỚC TẤN– 520H0418**

**NGUYỄN HOÀNG PHÚC KHANG– 520H0066**

**TRẦN LÊ GIA BẢO– 520H0516**

Class **: 20H50204**

Course  **: 24**

**HO CHI MINH CITY, 2022**

ACKNOWLEDGEMENT

Thank you a lot teacher. We are grateful for teacher who name is Mr Nguyễn Chí Thiện because he helped us a lot to know more and answer questions about the final report for our final and we can complete that report as soon as possible.

MIDTERM ESSAY COMPLETED AT TON DUC THANG UNIVERSITY

I hereby declare that this is our report and is under the guidance of master Nguyễn Chí Thiện. The research contents and results in this topic are honest and have not been published in any form before. The data in the tables for analysis, comments and evaluation are collected by the author himself from different sources, clearly stated in the reference section.

In addition, the project also uses a number of comments, assessments as well as data of other authors, other agencies and organizations, with citations and source annotations.

**If I find any fraud I take full responsibility for the content of my report**. Ton Duc Thang University is not related to copyright and copyright violations caused by me during the implementation process (if any).

*Ho Chi Minh city, 03 December, 2022*

*Author*

*(sign and write full name)*

*Phạm Phước Tấn*

*Nguyễn Hoàng Phúc Khang*

*Trần Lê Gia Bảo*

TEACHER’S CONFIRMATION AND ASSESSMENT SECTION

**Confirmation section of the instructors**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ho Chi Minh city, day month year

(sign and write full name)

**The evaluation part of the lecturer marks the report**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ho Chi Minh city, day month year

(sign and write full name)

SUMMARY

In this report, we find out and present six algorithms as six chapters, respective in table five of A Survey of Itemset Mining topic.

First chapter, we introduce about itemset and frequent itemset mining.

Chapter 2 and 3, we present and show some definitions about Apriori algorithm in two database representation as horizontal and vertical(TID-lists) with breadth-first(candidate generation) search.

Chapter 4, we present and show definitions about Eclat algorithm in a database representation as vertical (TID-lists, diffsets) with depth-first(candidate generation) search.

Chapter 5, we present and show definitions about FP-Growth algorithm in a database representation as horizontial (prefix-tree) with depth-first (pattern growth) search.

Chapter 6, we present and show definitions about H-Mine algorithm in a database representation as horizontial (hyperlink structure) with depth-first (pattern growth) search.

Chapter 7, we present and show definitions about LCM algorithm in a database representation as horizontial (with transaction merging) with depth-first (pattern growth) search.

TABLE OF CONTENTS

[ACKNOWLEDGEMENT i](#_Toc3267)

[MIDTERM ESSAY COMPLETED AT TON DUC THANG UNIVERSITY ii](#_Toc13451)

[TEACHER’S CONFIRMATION AND ASSESSMENT SECTION iii](#_Toc13823)

[SUMMARY iv](#_Toc23851)

[TABLE OF CONTENTS 1](#_Toc25404)

[LIST OF DIAGRAMS, CHARTS AND TABLES 3](#_Toc9153)

[CHAPTER 1 6](#_Toc22969)

[Question : 6](#_Toc16937)

[Answers : 6](#_Toc22271)

[Privilege 1 : Add Patients 6](#_Toc9858)

[Privilege 2 : Manage Identifier Types 6](#_Toc25749)

[Privilege 3 : Manage Privileges 7](#_Toc24690)

[Privilege 4 : Manage Roles 8](#_Toc22221)

[EXERCISE 2 8](#_Toc20826)

[Question : 8](#_Toc13957)

[Answers : 9](#_Toc32641)

[Role 1 : System Developer 9](#_Toc5702)

[Role 2 : Provider 10](#_Toc11069)

[Role 3 : Privilege Level: Full 10](#_Toc5432)

[Role 4 : Application: Has Super User Privileges 10](#_Toc32407)

[EXERCISE 3 10](#_Toc9911)

[Question : 10](#_Toc13071)

[REFERENCE 12](#_Toc11152)

**LIST OF ABBREVIATIONS**

EHR: electronic health record

LIST OF DIAGRAMS, CHARTS AND TABLES

List of Figures

[Figure 1 : Register a patient 4](#_Toc28622)

[Figure 2 : Manage Patient Identifier Types 5](#_Toc32527)

[Figure 3 : Manage previleges 6](#_Toc18601)

[Figure 4 : Manage roles 6](#_Toc16027)

List of Table

[Table 1 : A transaction database 5](#_Toc4086)

CHAPTER 1 - FREQUENT ITEMSET MINING

* 1. Definition Itemset and Frequent Itemset Mining

Before finding out and showing the algorithms in the table 5 of this topic so I will show a litle bit about itemset and frequent itemset mining.

An itemset is a set of items together when items are grouped often referred to as k-itemset depending on the number of items existence. An itemset consists of two or more items. An itemset that occurs frequently is called a frequent itemset. Thus frequent itemset mining is a data mining technique to identify the items that often occur together and focuses on the discovery of itemsets in databases, a popular data mining task for analyzing symbolic data.

The task of discovering itemsets in databases was introduced in 1993 by Agrawal and Srikant3 as large itemset mining but this name is old and now it is called frequent itemset mining (FIM) that is an efficient algorithm to mine the hidden patterns of itemsets within a short time and less memory consumption. So for frequent itemset mining method, we consider only those transactions which meet minimum threshold support and confidence requirements. Insights from these mining algorithms offer a lot of benefits, cost-cutting and improved competitive advantage.

Why frequent itemset mining wide used in the world ? because of its wide applications in mining association rules, correlations and graph patterns constraint that is based on frequent patterns, sequential patterns, and many other data mining tasks.

The matter above is about the definition of itemset and frequent itemset mining. So next, there are a definition about frequent itemset mining:

- A set of items: I = {i1, i2, . . . im} and a transaction database D= {T1,T2,…Tn}

is a set of distinct items. For instance I have a table transaction database in below:

|  |  |
| --- | --- |
| TID | Transaction |
| T1  T2  T3 | {a,c,d}  {a,b,c}  {a,c,b,d} |

Table 1: A transaction database

The table above includes three transactions T1, T2 and T3 with a,b,c,d letter that represent by what in the algorithm as client and admin.

In Frequent Itemset Mining, the interestingness of a given itemset is traditionally defined using a measure called the support. The support (or absolute support) of an itemset X in a database D is denoted as sup(X) and defined as the number of transactions containing X, that is sup(X) = |{T|X ⊆ T ∧ T ∈ D}| and it is called key of itemset Mining but there are also remaining that is confidence. So I will show the definition and example about two key of itemset mining:

- Support: it notices the main of item product in a transaction database and a measure of interestingness. Support is got when it divide the number of transaction including product (of some interestingness) by the total number of transaction in a database. For example, Support (banana candy) equal (transactions about banana candy) divide (total transactions).

- Confidence: it shows the ability of customer who buy two or three items as the same time in a single transaction. Want to calculate the confidence so we have to divide the number of transaction with two or three items for the total number in a database. For instance, confidence equals (transactions about banana candy and strawberry) divide for (total transaction relating candy).

Maximal itemset: If none of an Itemset’s supersets is frequent, the itemset is called as maximal frequent.

So to know more the FIM that apply to design many algorithms as Apriori, FP-Growth, Eclat, H-Mine and LCM. These have the similar input and output when implementing that with itemsets. That avoid exploring the search space of all possible itemsets. But they has some differences as they use a depth-first or breadth-first search as graph that we have already learned in the school.

CHAPTER 2 - APRIORI

1. Definition and Application of algorithm

Apriori is basically an algorithm for discovering frequent itemset in transaction databases. It was proposed by Agrawal & Srikant (1994).

Apriori algorithm is a machine learning model used in Association Rule Learning to identify frequent itemsets from a dataset. This model using Apriori algorithm that has a big retailers on over the world who apply this model for their stores to determine items that their customers frequently purchase together with high probability.

Using that knowledge, most of retailers use for arrange deals on their products such as offering a discount on best rules or attaching a free item depending on the quantity of best-associated rules customers buy together. Whichever the case, customers end up spending extra more to benefit from these deals. As a result, the business experience not only high sales but also a high profit.

1. Formulate the problem (Input and output)

Input: set of transactions D and minimum support minsup.

Output: L- frequent item set in D.

First, I will show Apriori Principle.

* Count the number of each items after that, find the most appear items on itemset .
* Find pairs of candidates: Count pairs => pairs of the most appear items.
* Find a candidate trio: Count the number of triplets => the most appearing three items. And continue with set 4, set 5,…
* The main rule: Every subset of a frequent set must be a frequent subset.

Second, describing algorithm following step by step:

* Step 1: Count the number of supports for each set of one element and treat them as a Large itemset. Their support is minsup.
* Step 2: For each Large item set that adds items and creates a new Large itemset, this set is called a candidate set (Candidate itemset - C). Count the number of supports for each set C on the database, then decide which set C is really Large Item, and we use it as the seed for the next step.
* Step 3: Repeat step 2 until no more can be found, another Large itemset set.

**Example 1:** Give an example set of transactions from purchase invoices as follows:

|  |  |
| --- | --- |
| **TID** | **Items purchased (Item)** |
| 1 | { b,m,t,y } |
| 2 | { b,m } |
| 3 | { p,s,t } |
| 4 | { a,b,c,d } |
| 5 | { a,b } |
| 6 | { e,t,y } |
| 7 | { a,b,m } |

For *Min Support = 30%,* *Min Confidence = 60%*

**Calculating the Large 1-item, we have F1:**

|  |  |
| --- | --- |
| **Set of item** | **Number of appearances** |
| {a} | 3 |
| {b} | 5 |
| {m} | 3 |
| {t} | 3 |

In the closing step From F1 above we have the set C2 consisting of pairs of 2-items:

{{a, b}, {a, m}, {a,t}, {b,m}, {b,t}, {m,t}}

**Calculating the Large 2-item, we have F2:**

|  |  |
| --- | --- |
| **Set of item** | **Number of appearances** |
| {a, b} | 3 |
| {a, m} | 1 |
| {a, t} | 0 |
| {b, m} | 3 |
| {b, t} | 1 |
| {m, t} | 1 |

Only take pairs of 2-items with Support > Min Support ( = 30% ) including: {a, b} and {b, m}

**Arising laws:**

a → b has degrees Confidence 3/3 = 100%

b → a has degrees Confidence 3/5 = 60%

b → m has degreesConfidence 3/5 = 60%

m → b has degrees Confidence 3/3 = 100%

In the omission step we have F2 = {{a, b}, {b,m}}

At the end of F2 we have the C3 set of 3-item pairs {}

The algorithm is out.

1. Present algorithms (pseudocode)

|  |  |
| --- | --- |
|  | **Input:** Set of transactions D and minimum support minsup |
|  | **Output:** L- tập mục phổ biến trong D |
|  | **Method:** |
| 1. | L1=***Large\_1\_ItemSets***() |
| 2. | **for** (k=2; Lk-1 ≠ ∅; k++) **do** |
| 3. | **begin** |
| 4. | Ck=apriori-gen(Lk-1); |
| 5. | **for** (mỗi một giao dịch TD) **do** |
| 6. | **begin** |
| 7. | CT = subset(Ck, T); |
| 8. | **for** (mỗi một ứng cử viên c CT) **do** |
| 9. | c.count++; |
| 10. | **end;** |
| 11. | Lk = {c ∈ Ck| c.count ≥ minsup} |
| 12. | **end;** |
| 13. | return ∪kLk |

* Explain function above ***Large\_1\_ItemSets***() will return items that have value of support higher than or equal mindsup.

|  |  |
| --- | --- |
| 1. | ***for all*** *transaction* t ∈ D ***do*** |
| 2. | ***for all*** *item* i ∈ t ***do*** |
| 3. | i.*count* ++; |
| 4. | L1={i | i.*count* ≥ *minsup*}; |

* Hàm ***Apriori\_Gen*** (Lk-1) thực hiện việc kết các cặp (*k-1*) *ItemSet* để phát sinh các tập *k\_ItemSet* mới. Tham số của hàm là Lk-1 – tập tất cả các (*k-1*)-*ItemSet* và kết quả trả về của hàm là tập các *k-ItemSet*.

|  |  |
| --- | --- |
| 1. | ***Join*** Lk-1 ***with*** Lk-1; |
| 2. | ***Insert into*** Ck |
| 3. | ***select*** p.item1,p.item2, . . .p.itemk-1, q.itemk-1 |
| 4. | ***from*** Lk-1 ***as*** p, Lk-1 ***as*** q; |
| 5. | ***where*** (p.item1= q.item1)∧...∧(p.itemk-2 = q.item k-2)∧(p.item k-1<q.item k-1);  Điều kiện (p.item k-1<q.item k-1) sẽ bảo đảm không phát sinh các bộ trùng nhau. |

1. Analyze asymtotic time and space complexity

Definition of Apriori:Apriori is an algorithm for discovering frequent itemsets in transaction databases.

1. Implement algorithms using a compiled programming language

Definition of Apriori:Apriori is an algorithm for discovering frequent itemsets in transaction databases.

1. Measure the running time, memory usage and draw their graphics

Definition of Apriori:Apriori is an algorithm for discovering frequent itemsets in transaction databases.

CHAPTER 3 - APRIORI-TID

1. Definition and Describe algorithm

Apriori TID has the same candidate generation function as Apriori. The interesting feature is that it does not use database for counting support after the first pass. An encoding of the candidate itemsets used in the previous pass is used. In later passes the size of encoding can become much smaller than the database,thus saving reading effort.

Each member of the episode Ck takes the form <TID, {Xk}> with Xk is set of *k-ItemSet* represents a part of the t transaction coded TID, or we can write <t.TID, {c∈Ck | c in t}>.

If a transaction does not contain any one set *k-ItemSet* which candidate, this assignment is not included . Thus, the number of applicants is included  may be smaller than the number of transactions in the database.

1. Formulate the problem (Input and output)

Input: vertical database

Output:

Simulation this algorithm in step by step:

* **Step 1**: Scan all transactions to find all items with greater support than Min Support and put the Large 1-Item set into F1.
* **Step 2**: Pass the entire Tid of the transaction together with the Items into C’1 as <Tid,{X1}>
* **Step 3**: Build pairs of 2-items from F1 into the C2 candidate set. Scan all transactions in C'1 to find all Large 2-Item sets from C2 included in C'2 as <Tid,{X2}>, also put Large 2-Item sets into F2.
* **Step 4**: arising Act. Build k items pairs from Fk-1 to put into Ck candidates. Scan all transactions in C’k-1 to find all Large k-Item sets from Ck and put in C’k as <Tid,{Xk}>, and put into Large k-Item sets in Fk. Repeat Step 4 until new candidates run out.

For an example, the set of Tid transactions with Items is as follows:

|  |  |
| --- | --- |
| **Tid** | **Items** |
| 100 | {1, 3, 4} |
| 200 | {2, 3, 5} |
| 300 | {1, 2, 3, 5} |
| 400 | {2, 5} |

For *Min Support* = 50%, *Min Confidence* = 60%

Calculating the Large 1-item, we have F1:

|  |  |
| --- | --- |
| **Volume 1-item** | **Number of appearances** |
| { 1 } | 2 |
| { 2 } | 3 |
| { 3 } | 3 |
| { 5 } | 3 |

Take all <Tid,{X1}> put into C’1

|  |  |
| --- | --- |
| **Tid** | **Volume 1-item** |
| 100 | {{1 }, {3}, {4}} |
| 200 | {{2}, {3}, {5}} |
| 300 | {{1}, {2}, {3}, {5}} |
| 400 | {{2}, {5}} |

In the closing step From F1 above we have the set C2 consisting of pairs of 2-items:

{{1,2}, {1,3}, {1,5}, {2,3}, {2,5}, {3,5}}

Identify candidates from C2 when browsing Tid in C'1 and assign in C'2

|  |  |
| --- | --- |
| **Tid** | **Volume 2-item** |
| 100 | {{1,3}} |
| 200 | {{2,3}, {2,5}, {3,5}} |
| 300 | {{1,2}, {1,3}, {1,5},  {2,3}, {2,5}, {3,5}} |
| 400 | {{2,5}} |

In the closing step From F1 above we have the set C2 consisting of pairs of 2-items:

{{1,2}, {1,3}, {1,5}, {2,3}, {2,5}, {3,5}}.

Calculate Large 2-Item, we have F2:

|  |  |
| --- | --- |
| **Volume 2-items** | **Number of appearances** |
| {1,3} | 2 |
| {2,3} | 2 |
| {2,5} | 3 |
| {3,5} | 2 |

At the end of F2 we have the C3 set consisting of a pair of 3-items: {{2,3,5}}

Identify candidates from C3 when browsing Tid in C'2 and assign into C'3

|  |  |
| --- | --- |
| **Tid** | **Volume 3-item** |
| 200 | {{2, 3, 5}} |
| 300 | {{2, 3, 5}} |

Calculating the Large 3-Item, we have F3:

|  |  |
| --- | --- |
| **Volume 3-items** | **Number of appearances** |
| {{2, 3, 5}} | 2 |

Arising laws:

2,3 🡪 5 has degrees Confidence 2/2 = 100%

2,5 🡪3 has degrees Confidence 2/3 = 66,66%

3,5 🡪 2 has degrees Confidence 2/2 = 100%

At the end of F3 we have the set C4 consisting of pairs of 4-items which are  **{∅}.**

The algorithm ends.

1. Present algorithms

Definition of Apriori:Apriori is an algorithm for discovering frequent itemsets in transaction databases.

1. Analyze asymtotic time and space complexity

Definition of Apriori:Apriori is an algorithm for discovering frequent itemsets in transaction databases.

1. Implement algorithms using a compiled programming language

Definition of Apriori:Apriori is an algorithm for discovering frequent itemsets in transaction databases.

1. Measure the running time, memory usage and draw their graphics

Definition of Apriori:Apriori is an algorithm for discovering frequent itemsets in transaction databases.

CHAPTER 4 - ECLAT

1. Definition and Describe algorithm

The Eclat algorithm stands for Equivalence Class Clustering and bottom-up Lattice Traversal. It is one of the popular methods of Association Rule mining. It is a more efficient and scalable version of the Apriori algorithm. While the Apriori algorithm works in a horizontal sense imitating the Breadth-First Search of a graph, the Eclat algorithm works in a vertical manner just like the Depth-First Search of a graph. This vertical approach of the Eclat algorithm makes it a faster algorithm than the Apriori algorithm.

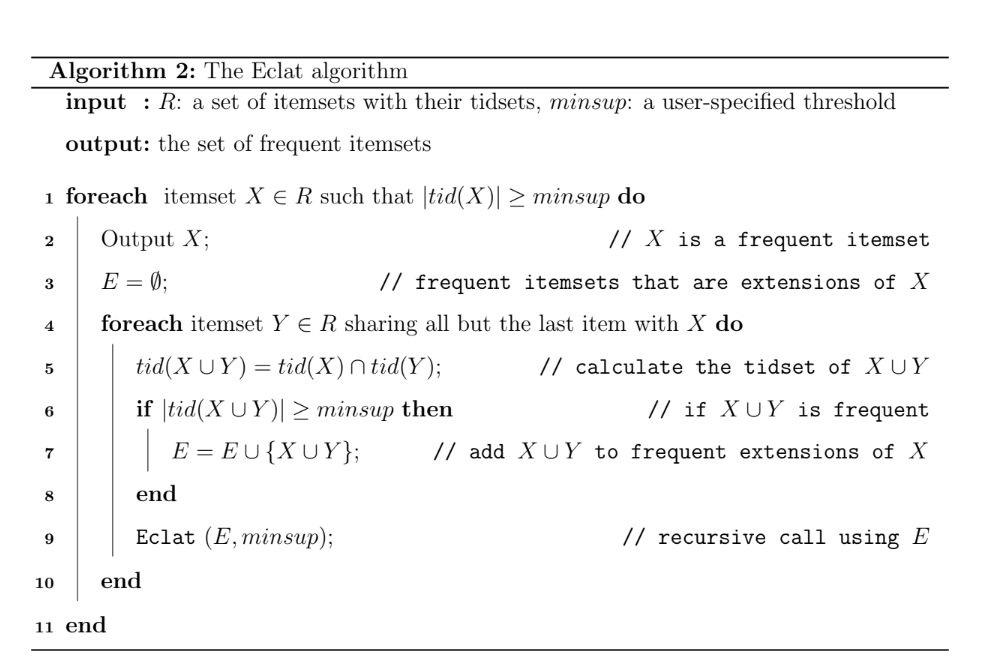
The basic idea is to use Transaction Id Sets (tidsets) intersections to compute the support value of a candidate and avoiding the generation of subsets which do not exist in the prefix tree. In the first call of the function, all single items are used along with their tidsets. Then the function is called recursively and in each recursive call, each item-tidset pair is verified and combined with other item-tidset pairs. This process is continued until no candidate item-tidset pairs can be combined.

1. Formulate the problem (Input and output)

Input: A transaction database is a set of transactions. Each transaction is a set of items. For example, consider the following transaction database. It contains 5 transactions (t1, t2,…, t5) and 5 items (1,2, 3, 4, 5). ( It is important to note that an item is not allowed to appear twice in the same transaction and that items are assumed to be sorted by lexicographical order in a transaction).

Output: (don’t completed)

1. Present algorithms



1. Analyze asymtotic time and space complexity

Definition of Apriori:Apriori is an algorithm for discovering frequent itemsets in transaction databases.

1. Implement algorithms using a compiled programming language

Definition of Apriori:Apriori is an algorithm for discovering frequent itemsets in transaction databases.

1. Measure the running time, memory usage and draw their graphics

Definition of Apriori:Apriori is an algorithm for discovering frequent itemsets in transaction databases.

CHAPTER 5 - FP-GROWTH

1. Definition and Describe algorithm

FP-Growth represents transaction data using a data structure called FP - Tree.

FP-Growth uses FP-Tree to directly determine frequent item sets (does not generate candidate item sets from previous candidate item sets).

Once an FP-Tree has been constructed, the FP-Growth uses a recursive divide-and-conquer approach to mine frequent sets.

For each transaction, the FP-Tree builds a path in the tree.

If two transactions contain the same number of entries, their paths will have some part (segment) in common.

The more paths with common elements, the more compact the FP-Tree representation.

1. Formulate the problem (Input and output)

Input: The input of FPGrowth is a transaction database (aka binary context) and a threshold named minsup (a value between 0 and 100 %).

Output: : FP-tree, the frequent-pattern tree of DB.

1. Present algorithms

Definition of Apriori:Apriori is an algorithm for discovering frequent itemsets in transaction databases.

1. Analyze asymtotic time and space complexity

Definition of Apriori:Apriori is an algorithm for discovering frequent itemsets in transaction databases.

1. Implement algorithms using a compiled programming language

Definition of Apriori:Apriori is an algorithm for discovering frequent itemsets in transaction databases.

1. Measure the running time, memory usage and draw their graphics

Definition of Apriori:Apriori is an algorithm for discovering frequent itemsets in transaction databases.

The end.

REFERENCES

1. [Introduce the itemset and Frequently Itemset Mining](https://www.section.io/engineering-education/introduction-to-frequent-itemset-mining-with-python/)
2. [Apriori algorithm](https://www.section.io/engineering-education/apriori-algorithm-in-python/)
3. [Apriori as well](https://www.philippe-fournier-viger.com/spmf/Apriori.php)
4. [FP-Growth algorithm](https://www.softwaretestinghelp.com/fp-growth-algorithm-data-mining/)
5. <https://viblo.asia/p/khai-pha-du-lieu-va-lop-bai-toan-khai-thac-cac-tap-pho-bien-p2-m68Z0W06KkG>
6. <http://www.philippe-fournier-viger.com/spmf/FPGrowth.php>
7. <https://www.semanticscholar.org/paper/An-efficient-parallel-FP-Growth-algorithm-Chen-Gao/1f8dd4242399b3cde9ce416db5e295f9f5b76b90>