Received: XXX XX, 2022. Revised: XXX XX, 20XX. XX

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Improvement of Apriori Algorithm Performance Using the TID-List Vertical Approach and Data Partitioning

Moch. Syahrir[1] Anthony Anggrawan[2]

[1] *Software Engineering-Faculty of Engineering-University of Bumigora, Mataram*
[2] *InformationTechnology Education-Faculty of Engineering-University of Bumigora, Mataram*
* Corresponding author's Email: muhammadsyahriralfath@gmail.com

Abstract: The Apriori algorithm is one of the most widely developed and used association rule algorithms because it can produce optimal rules. The association rule a priori algorithm has three main problems: the old dataset scan process, less than optimal rules formed, and the use of large memory. Therefore, this study aims to improve the performance of the a priori algorithm in the itemset frequency search process with optimal rules, small memory usage, and fast dataset scans. The method used in this study is TID-List Vertical and data partitioning. This study result indicates that the TPQ-Apriori method improves the performance of the dataset scan process up to two times faster in scanning datasets compared to the study's results using the traditional a priori method. The performance of the dataset scan process from the results of this study is also superior in processing speed by up to 7% on scanning datasets compared to the ETDPC-Apriori method on testing three data sets (mushroom, chess, and c20d10k).

Keywords: Association Rule, Apriori, Eclat, Fp-Growth, Data Partitioning, TID-List Vertical.

1. Introduction

Data mining is a scientific discipline that aims to extract knowledge and find patterns from extensive data by studying and developing algorithms[1][2]. In its role, data mining consists of the roles of estimation, forecasting, classification, clustering, and association[3]. The use of data mining algorithms must be in harmony with the characteristics of the data [4]. Moreover, the complete data characteristics (both the number of attributes and the category of each attribute) will result in more optimal performance of data mining algorithms. The role of associations is widely implemented in the business fields such as e-commerce, retail, and restaurants, but it is possible that it can be applied to other fields such as software bug analysis and analysis of biological data and medical data[5] so that we ensure that the benefits are widely used in various fields.

Association, or what is commonly referred to in data mining, is an association rule which is one of the data mining techniques to find and find a set of association rules between a combination of items[6]. Or in another sense, it is an associative rule from the implications of a combination of relationships between an item[4][6]. Association rule algorithms that are commonly used and developed are Apriori, Fp-Growth, Eclat, and Hash Based[1][6][7][8]. In the association rule algorithm, the resulting rule is in the form of a rule that can be measured using support, confidence, lift ratio, leverage, conviction, and certainty factor. Support is the percentage of the combination of these items in the database. Confidence is the strength of the relationship between items in the association rules. The lift ratio tests the validity of the relationship between items as well as leverage and conviction to test how much influence and confidence there is between antecedent and consequent.

The problem in this study is the association rule algorithm has several shortcomings or weaknesses, which always lead to three main problems: dataset scan time, rule formation, and memory and processor usage.

In addition, considering that better performance concerning processing time consumption is a significant concern for most researchers to make it happen [9]. Therefore the priority of this research is to speed up the process of finding the frequency of itemset in the dataset and to maintain and establish optimal and consistent rules. The forerunner of association algorithms is the Apriori algorithm which was first presented in seminars[10], then improved a year later[11][12]. In its implementation, Apriori can produce optimal rules but the time used to scan the dataset is very long because the approach used by Apriori in finding frequency uses candidate generation where all items must be traced to determine the candidate k-itemset, k-itemset means how many iterations are will be formed from all searches and result in a lot of memory use[6][13]. The improvement of the Apriori algorithm was carried out by[14] with a hashing method approach. The hashing method is one method that can be used to form an array table and form a key address with the hashing function[15]. Although the Apriori algorithm has succeeded in improving its performance of the Apriori algorithm, the hashing function used in this study is still standard, and there are still many collisions[16]. A few years later, the Apriori algorithm was improved by[17] with the TID-List Vertical approach, transforming the table from a horizontal to a vertical form. This approach is very effective in trimming the tuples in the transaction table on the condition that the TID is more than the item. However, the weakness of this algorithm is that it cannot do pruning to the minimum frequency[18]. Now, this algorithm is known as Equivalent Class Transformation (ECLAT). And what is popularly known is that the improvements made by[5] with the fp-tree approach in terms of dataset scan time are very good because they only scan the dataset twice, but the rules generated by the Fp-Growthh algorithm are not as optimal as Apriori and also use memory is still quite large[19], which is now known as Frequent Pattern Growth(Fp-Growth).

In solving data mining problems, especially the role of associations, they are grouped into two categories, namely (1) candidate generation using the breadth-first search algorithm technique and (2) pattern growth using the depth-first approach. Algorithms that use a depth-first approach[20] are considered more complicated because they use data structures in their research [21]. Furthermore, the breadth-first search technique is more widely used because it is simpler, one of which is by using the a priori algorithm. The Apriori algorithm is the most classic algorithm and is quite important in frequent item-sets mining. Although many similar algorithms that are more efficient have been developed, such as Fp-Growth, Eclat, Hash-Based, and others, the a priori algorithm is still the most widely used and implemented in commercial products for data mining because it is considered a more established algorithm and the resulting rules. optimal[6][22].

There are several techniques and approaches that can be used to improve the performance of association algorithms, especially Apriori, namely Hash-Based format, partitioning, sampling, and vertical format[5][23][13]. The hash-Based format uses a hashing method to determine the address of the key to be accessed to find the frequency of an item[13][24]. Partitioning is done to divide or partition the items to be scanned or tuples of data[13] because in the association algorithm, all items will become attributes, the more items, the more attributes will be scanned, this is what results in scanning the old dataset. Sampling is done by taking some of the items that are considered to represent the contents of the dataset to be processed. Vertical format transforms the table format change from horizontal to vertical [13][17]. With this technique, tuples can be reduced on condition that the TID is greater than the item, and also this technique also simplifies the item partitioning process. In general, research for association rule algorithms in their experiments uses two databases, namely vector arrays and SQL as dataset management. However, the use of vector arrays is more than SQL, even though the actual implementation in the field is SQL-based. Empirically the results show that the Apriori algorithm can be improved with SQL–based approach [5][25][26][27][28][29]. Moreover, recently most database systems have included the ability to support parallelization[30][12], and there are also recent studies that have started to form the SQL-based Apriori framework [31][32]. The strength of the method proposed in this study is the combination of partitioning and vertical TID-List techniques to improve the performance of the a priori algorithm, which is the novelty of this research.

This research aims to integrate the TID-List Vertical approach and data partitioning to find the frequency of itemsets in the Apriori algorithm so that the runtime scan of the dataset can be faster and produce optimal and consistent results rules. The sequence of the next writing structure of this manuscript is as follows: the second sub-section discusses related works of previous research results. The third sub-section describes the materials and methods of this study. Meanwhile, the fourth sub-section focuses on explaining the results and discussion, which is then closed with the conclusion sub-section.

2. Related works

There are several papers that discuss the performance of association rule algorithms, both Apriori, Eclat, Hash-Based, and Fp-Growth. For example, to develop the results of research and papers by previous authors, which are the following development materials:

2.1 Apriori Method

Although Apriori is weak in long dataset scan times, the resulting rules are optimal[11]. Below Figure 1 is a flowchart of the traditional Apriori proposed by[33].

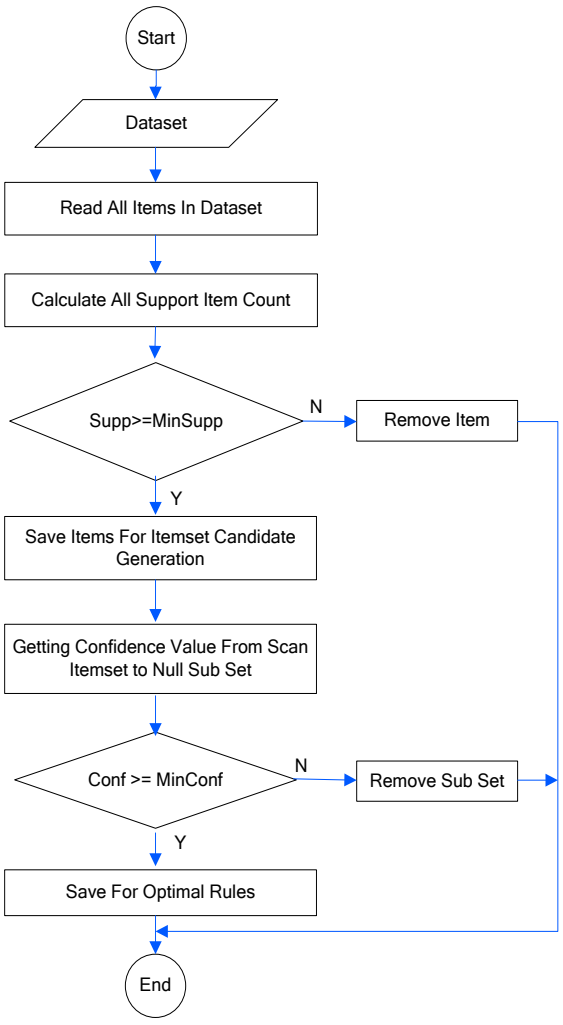

Figure 1 Apriori Method Flowchart

2.2 Bi-Eclat Method

Research related to the improvement of the method proposed by[34] in this study improves the performance of the Eclat algorithm with a lattice and prefix-based theoretical approach to split the original database into a minor form so as to speed up the dataset scan process.

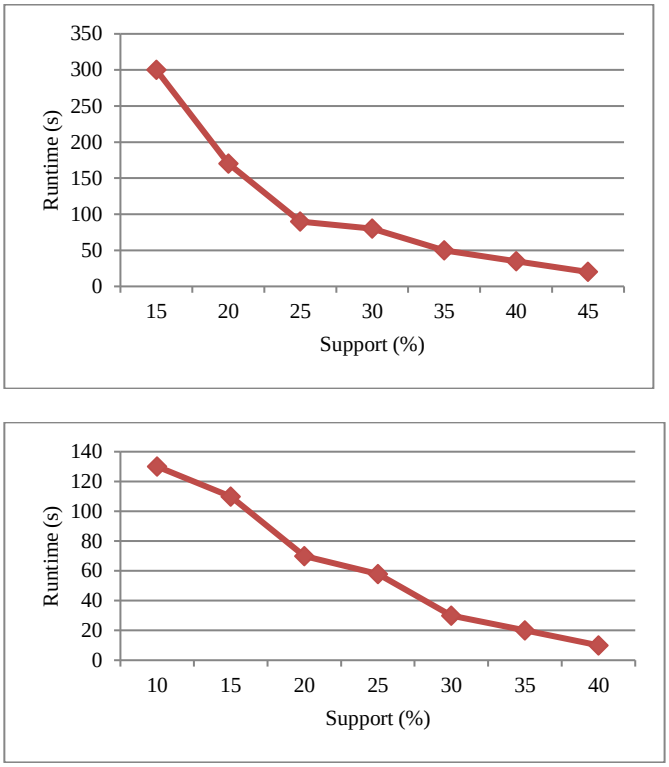

Figure 2 Testing Results of Dataset Connect and Mushroom With Bi-Eclat

However, the resulting rule is not explained because when the database is split or partitioned, it will reduce the optimization of the resulting rule. The public dataset is at the top of the connect dataset and below the mushroom dataset for testing. In the above test, there are two algorithms, namely traditional Eclat with orange Bi-Eclat, while the one developed in this study is Bi-Eclat, but what is shown in Figure 2 is only the result of improvised Eclat.

2.3 FRESTM Method

In another study conducted by[35], the approach used is a subtree combined with a candidate generation technique which is given a new name, namely the frequent restrictedly embedded subtree miner (FRESTM). The dataset used in this study is a private dataset of 10,000 records and result Figure 3.

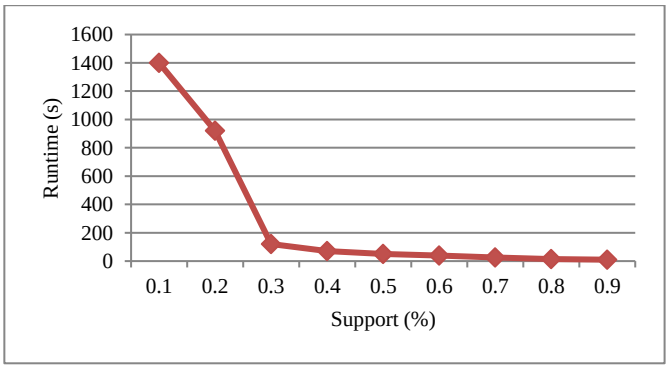

Figure 3 Private Dataset Test Results Using the FRIESTM Method.

2.4 ETDPC-Apriori-Based MapReduce and Hadoop

Research on improving the performance of the association rule algorithm but in the form of a collection of methods in a single framework is carried out by[36]. In this study, improvements are made to the Apriori-based method that has been packaged in the MapReduce and Hadoop frameworks to scan the dataset using several approaches. Packaged in a framework, namely VFPC (variable size based fixed passed combinate-counting), SPC (single pass counting), FPC (fixed passed combinate-counting), DPC (dynamic passed combined-counting), ETDPC (elapsed time-based dynamic passed combined-counting) while being developed is VFPC and ETDPC.

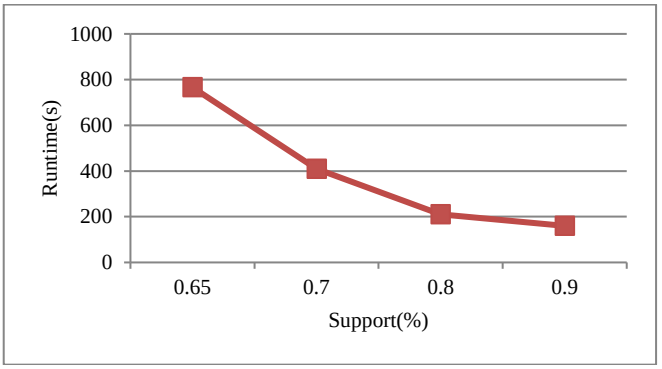

Figure 4 Chess Dataset Test Results With ETDPC-Apriori.

The dataset used for the experiment is the public cherry and mushroom dataset. Figure 4 is the chess dataset and Figure 5 is the mushroom dataset.

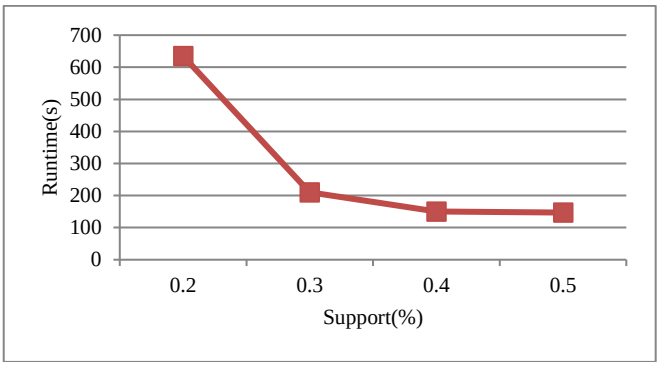

Figure 5 Mushroom Dataset Test Results With ETDPC-Apriori

This study does not discuss the rules generated by the framework, only showing a comparison of the four methods in the MapReduce and Hadoop frameworks, two of which have been improved and the other two without improvement. But what is contained in Figure 4 and Figure 5 is only the result of improvisation.

2.5 NIS-Apriori

Research on the development of an SQL-based Apriori framework called NIS-Apriori[21]. In this study, the functions have been made directly usable because the research is still very new. The framework they offer cannot be experimented with because it has not been disseminated. It can only run on web-based applications, and no database partitioning model is implemented in this framework.

3. Materials and Method

This section contains the stages of the research method. This stage is also used to explain the solutions to the research problems and to achieve the research objectives.

3.1 Research Design

The research method used is an experimental research method, with the following research stages:

- Dataset Collection, The dataset used is a public dataset. The public dataset obtained is a dataset sourced from http://www.philippe-fournier-viger.com, www.uci.com, http://fimi.uantwerpen.be/data/
- Proposed Method, At this stage, the data will be analyzed after the data has been analyzed and then applied with methods that are in accordance with the type of data.
- Initial Data Processing (Pre-processing), Initial data processing includes data cleaning and adjustment to the required formats and extensions.
- Experiment And Testing Method, At this stage, experimental steps are explained, covering how to choose the right architecture from the proposed method so that results are obtained that can prove that the method is appropriate and the results are better.
- Evaluation, At this stage, an evaluation of the results of the application of the proposed method is carried out in determining the optimal rule formation, faster scan or runtime times, and relatively lower memory and processor usage although the third point is not a priority in this study.

3.2 Dataset Collection

Data collection is the initial stage carried out, where from the existing problems, related data will be

collected in the form of public and private datasets. The public dataset obtained is data sourced from http://www.philippe-fournier-viger.com, www.uci.com, and http://fimi.ua.ac.be/data/ with different record variations and attributes. -different. The details of the dataset are as follows:

1. Mushroom
2. Chess
3. Foodmart
4. C20d10k

3.3 The Proposed Method

This study will propose a method where the initial stage is carried out, namely pre-processing to adjust the format as needed in this study with the dot SQL (.sql) extension. The next stage is partitioning the dataset, for the partitioning process itself in this study only comes to stage four dataset partitions. After the dataset partitioning process is complete, each partition will apply the TID-List Vertical approach, which aims to reduce the records. It makes the itemset frequency search process faster, and also, in the proposed method, it is possible to partition items. Item partitioning can be done after the application of TID-List Vertical if needed. The complete flow of the proposed method can be seen in Figure 3.1. The stages are preparing the dataset and then transforming the format and form of the tables in the dataset to a form that is in accordance with the format required in this study so that it can be easily processed and the extension changed to dot SQL (.sql). Perform the process of partitioning the dataset as a result of the transformation that has been carried out previously. Datasets are partitioned based on their records. In this study, a maximum of four datasets were partitioned. The more partitions, the faster the itemset frequency search process, but the implementation process in the database is getting more complicated.

$$Partitioning\ Transaction = \frac{TDB}{\sigma_r} \quad (1)$$

Where:
$TDB = Transaction\ Database$
$\sigma_r = Number\ of\ Partitions$

The partition formula used is the partition model in the study[37]. Next, apply the TID-List Vertical approach to each dataset that has been previously partitioned. With this approach, the records in the dataset can be reduced. We can see the illustration in Figure 7. Reads all items or attributes in each partitioned dataset to search for support count items, calculates the support count of each item or attribute, then determines the minimum support. If the support count is greater than the minimum support, the item will be stored for processing the next generation of candidates. Otherwise, it will be deleted. The results of the generation of candidate items whose support count is selected will continue in the next generation of candidates until the k-itemset is selected. Then calculate the confidence, lift ratio, conviction, leverage, and certainty factor from the results of the support count or frequency obtained from the scan itemset until the sub-set is null. The confidence calculation formula is:

$$confidence, c(X \rightarrow Y) = \frac{\sigma\{X \cup Y\}}{X} \quad (2)$$

Calculation formula liftrasio

$$lift(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{\sigma(X) * \sigma(Y)} \quad (3)$$

Calculation formula conviction

$$conv(X \rightarrow Y) = \frac{1 - \sigma(Y)}{1 - c(X \rightarrow Y)} \quad (4)$$

Calculation formula leverage

$$lev(X \rightarrow Y) = \sigma(X \cup Y) - (\sigma(X) * \sigma(Y)) \quad (5)$$

Calculation formula certainty factor(CF)

$$CF(X \rightarrow Y) = \frac{c(X \rightarrow Y) - \sigma(Y)}{1 - \sigma(Y)} \quad (6)$$

Next, Determine the minimum confidence; if the confidence is greater than or equal to the minimum confidence, then the sub-set will be stored for optimal rule formation. Otherwise, it will be deleted. Rule based limit for antecedent and consequent is 4-Itemset.

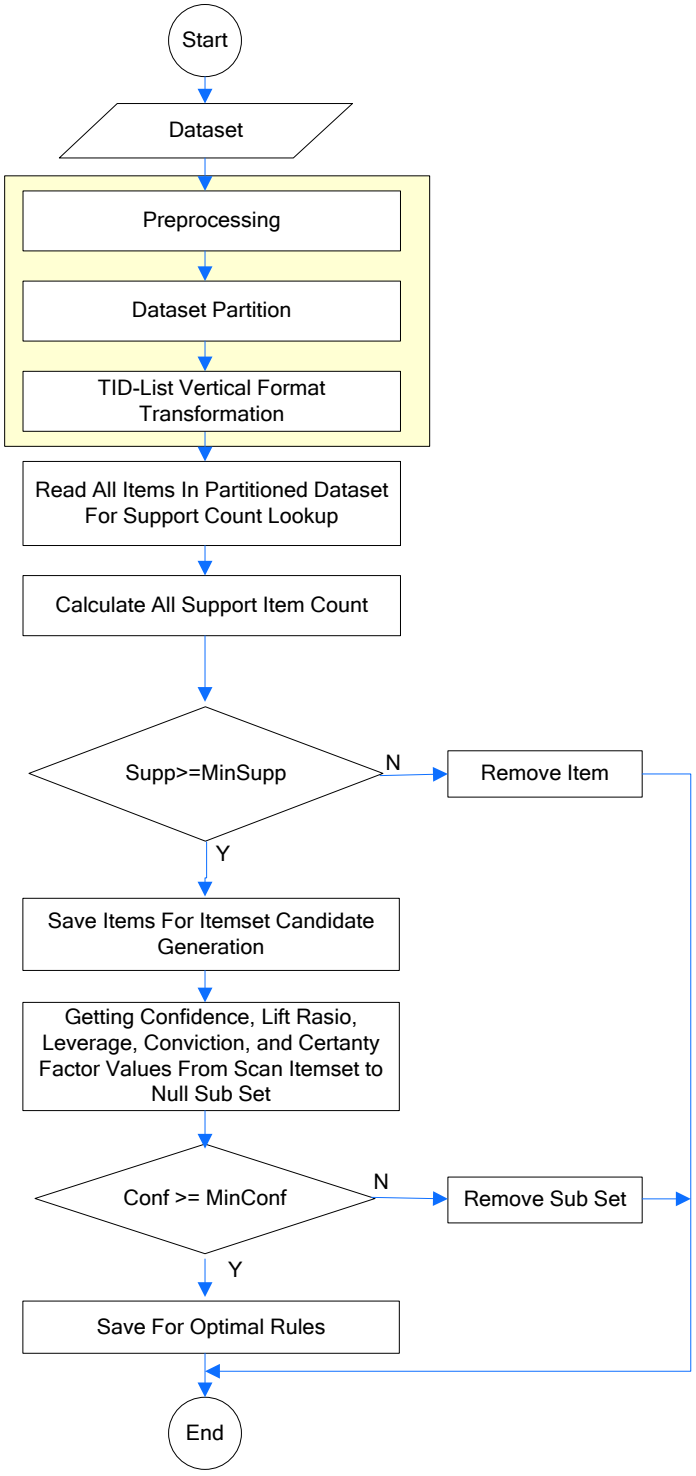

Figure 6 Flowchart of the Proposed Method

In Figure 6, the section marked with a blue box is an approach added to speed up the itemset frequency search process in the Apriori algorithm. Table 1 shows the comparison of the proposed method with the methods that previous researchers have proposed. The process of calculating the scan time of the dataset starts when the dataset is scanned for item set frequency searches, not during pre-processing. This is because the pre-processing stage only fixes and adjusts the dataset format so that it is easy to process, not to find the frequency of the itemset.

Table 1 Comparison of Proposed Methods

Researcher	Pre-Processing	Data Partitioning	TID-List Vertical	Evaluation
Agrawal, 1994[38]	No	No	No	Support, Confidence
Xiaomei Yu 2014[34]	Yes	No	Yes	Support, Confidence
Sen Zhang 2015[35]	Yes	Yes	No	Support, Confidence
Sudhakar Singh, 2018[36]	No	Yes	No	Support, Confidence
Our Research (this research)	Yes	Yes	Yes	Support, Confidence, Liftrasio, Conviction, Leverage, Certainty Factor

The method proposed by Agrawal and Srikant[11] is still a pure and classical Apriori algorithm where there is no pre-processing, data partitioning approach, and TID-List Vertical so that it purely searches for the frequency of itemset from transaction data. Bi-ECLAT in this study improves the performance of the éclat algorithm with a lattice and prefix-based theory approach to split the original database into a smaller form size so as to speed up the process of scanning datasets by Xiaomei Yu[34]. The FRESTM approach used is a subtree combined with a new candidate generation technique, namely frequent restrictedly embedded subtree miner (FRESTM) by Sen Zhang[35]. Furthermore, the method used by Sudhakar Singh[36] is a method in the form of the MapReduce and Hadoop frameworks released by Google to process big data, which is increasing day by day. Because there are many collections and bases of Apriori algorithms in the framework, the researcher chose the ETDPC-Apriori algorithm to improve its performance. In this framework, data partitioning is applied to large data. Data is grouped or clustered into small datasets so that the itemset frequency search scan process can be faster and supports a parallelization system where dataset scans can be performed on different personal computers. The results of the support count or frequency obtained will be put back together at the end of the process. However, in each partitioned dataset, there is no application of TID-List Vertical. Meanwhile, in the proposed method, apart from using the data partitioning approach for the dataset partitioning process, it also applies the TID-List Vertical approach.

3.4 Initial Processing of Data

Before entering the algorithm model, the data must be pre-processed first to ensure that the format that will enter the model is as expected. The better the data to be processed, the more optimal the results of the algorithm process. In this study, the pre-processing process was carried out using the rapid manner tools with cleansing and blending techniques if the data extension was excel or CSV, while for the dot data extension format (.dat).) whose contents are in the form of text format using a technique developed by myself in this study to change the format to the required form and change the extension to dot SQL(.sql). However, the four datasets processed in the proposed method have dot data(.dat) extensions.

3.5 Experiment And Testing Method

Before starting the process of experimentation and method testing, the name of the method proposed in this study is TPQ-Apriori to make it easier to understand the images and graphs of the runtime test results later. The experimental stages of the proposed method are:

- Setting up the dataset
- Testing traditional Apriori runtime results per itemset
- Testing TPQ-Apriori runtime results per-itemset
- Testing the ETDPC-Apriori runtime results
- Testing the results of the TPQ-Apriori runtime
- Comparing the TPQ-Apriori runtime test results with the traditional Apriori per itemset, the test uses four datasets: mushroom, chess, food mart, and c20d10k.
- Comparison of TPQ-Apriori runtime test results with ETDPC-Apriori, this test uses three datasets, namely mushroom, chess, and c20d10k.

In the runtime testing process, four public datasets, namely chess, mushroom, food mart, and c20d10k datasets, can be downloaded at http://www.philippe-fournier-viger.com and www.uci.com. The chess, mushroom, and c20d10k datasets will be compared with the ETDPC-Apriori which was formed in the framework of the research results[36] conducted by Sudhakar Singh, while all datasets will be tested between TPQ-Apriori and traditional Apriori. In this study, testing was carried out using the TPQ-Apriori tools, which were developed by themselves in this study using Microsoft Visual Studio 2013 tools and Mysql server 5.5, which supports parallelization. While the computer specifications used in this test can be seen as shown in Table 2.

Table 2 Hardware and Software Specifications for Testing

No	Hardware	Deskripsi
1	Processor	Intel ® Dual Core G630 2.70Ghz
2	Ram	4GB DDR3
3	Storage	120GB SSD
4	Operating System	Windows 8 Profesional 64Bit
5	Developer Tools	Microsoft Visual Studio 2013
6	Testing Tools	TPQ-Apriori
7	Database Tools	Mysql Server 5.5

3.6 Evaluation

The evaluation process will provide results about the value of support, confidence, lift ratio, conviction, leverage, and certainty factor. Support determines how often the rule is applied in the dataset[1][5][23][39]. Support is an indication of how often the itemset appears in the dataset. Support can be formulated as Formula 3.7.

$$Support, s(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{N} \quad (7)$$

Where :
$N = Total\ Transaction$
$X = Antecedent$
$Y = Consequent$

Confidence determines the frequency with which items in Y appear in transactions containing X. Confidence is how often the rule or rule is proven true. Confidence can be formulated as in Formula 3.8.

$$confidence, c(X \rightarrow Y) = \frac{\sigma(X\ \cup Y)}{X} \quad (8)$$

To test whether or not a rule is valid or the relationship between items uses lift ratio. Lift ratio is a value that measures the magnitude of the relationship between antecedent and consequent that is not independent[1][4][6][35] Liftratio has a range from 0 to . A value close to 1 indicates that the antecedent and consequent are not dependent. A value far from 1 indicates that the antecedent provides information about the consequent. Or in other words, if the lift is > 1, it lets us know to what extent the two events are dependent on each other and makes the rule potentially useful for predicting the consequence in the dataset. And if the lift is < 1, it lets us know that the items replace each other. This means that the presence of one item has a negative effect on the presence of another item and vice versa.

$$lift(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{\sigma(X) * \sigma(Y)} \quad (9)$$

Where :
$\infty = Infinity\ or\ Null$
$\sigma = Support\ Count$
$\cup = Relate$

In addition to the lift ratio, it can test with another formula: conviction or belief value. Conviction is a value that measures the level of implication of a rule[1][4][6][35]. Conviction is very concerned about the direction of an association rule. Conviction indicates that conviction(X→Y) conviction(Y→X).

$$conv(X \rightarrow Y) = \frac{1 - \sigma(Y)}{1 - c(X \rightarrow Y)} \quad (10)$$

For the number of antecedent and consequence items that are sold simultaneously in a dataset that is more than expected, we can use Leverage[1][5][23][39]. A value of 0 indicates independent antecedent and consequent. Leverage has a range of values from -0.25 to 0.25. Can be formulated as in formula 3.11.

$$lev(X \rightarrow Y) = \sigma(X \cup Y) - (\sigma(X) * \sigma(Y)) \quad (11)$$

This study takes for additional evaluation a decision in assessing the relationship or correlation of the rules formed using the certainty factor method. According to David McAllister, the certainty factor method "is a method to prove whether a fact is certain or uncertain in form, a metric usually used in expert systems[40]".

$$CF(X \rightarrow Y) = \frac{c(X \rightarrow Y) - \sigma(Y)}{1 - \sigma(Y)} \quad (12)$$

And finally, to calculate the scan time of the dataset, we can use a formula like in Formula 3.13.

$$Runtime = End - Start \quad (13)$$

Where :

$Start$
$= Record\ the\ start\ time\ of\ the\ scan\ process$
$End = Record\ the\ end\ time\ of\ the\ scan\ process$

To measure the runtime scan of the dataset, it starts when the dataset reads the contents of the dataset in the process of searching for the itemset frequency until there are no more items or attributes left.

4. Result and Discussion

Experiments to overcome the problem of scanning old datasets, the formed rules are not optimal, and the use of memory and processors that are still quite large by integrating the TID-List Vertical approach and data partitioning. Where data partitioning is used to partition the dataset so that the volume of the dataset can be partitioned in a smaller size than the original dataset, and for each dataset that has been partitioned, the results of the partition table format initially in the horizontal form will be transformed into vertical form and with these two approaches, the frequency search process in the Apriori algorithm developed becomes faster. The runtime testing process is repeated ten times to ensure consistent dataset scan time. The ten-times test found that the time difference in the dataset scan process was in the range of 0 to 4 seconds.

4.1 Data Collection

In this study, the datasets used are public and private datasets. The pubic datasets were obtained from uci (https://archive.ics.uci.edu) and the philippe fournier-viger website (http://www.philippe-fournier-viger.com) and (http://fimi.ua.ac.be/data/). The number of datasets used are four public datasets. The public dataset consists of mushroom, chess, foodmart, c20d10k datasets. The characteristics of the dataset used in this study can be seen in Table 3.

Table 3 Characteristics of the Dataset Used for Testing.

No	Dataset	ΣTransaction	ΣRecord	Attribute	Type
1	Mushroom	8,124	186,825	119	*Public*
2	Chess	3,196	118,252	75	*Public*
3	Foodmart	4,141	41,410	1,559	*Public*
4	C20D10k	10,000	200,000	192	*Public*

For the data sets: mushroom, chess, food mart, and c20d10k have an extension of dot dat (.dat).

4.2 Pre-Processing Data

Before entering the algorithm model, the data must be pre-processed to ensure the format that will enter the model is as required. The better the data to be processed, the more optimal the results of the algorithm process will be. In this study, the pre-processing process was carried out using rapid miner tools with cleansing and blending techniques if the data extension was excel or CSV, while for the dot data extension format (.dat).) whose contents are in the form of text format using a technique developed by myself in this study to change the format to the required form and change the extension to dot SQL(.sql). However, the four datasets processed in the proposed method have dot data(.dat) extensions.

4.3 Dataset Partition

The pre-processed dataset will then be partitioned. In this study, the maximum partition is four partitions. It can use excel or the tools developed for the partition process in this study. The partition that is done is the partition against the volume and its attributes. Volume partitioning is mandatory, while attribute partitioning is not mandatory. Attribute partitioning is performed after the first iteration of the itemset frequency search process, while volume partitioning is performed before the first iteration is performed. In ensuring the correctness of the volume partitioning process, one thing must be known: The TID (Transaction Identifier) cannot be entered on a different partition to prevent TID duplication of the dataset from being scanned. So it will be easier to understand. The illustration can be seen in Figure 7.

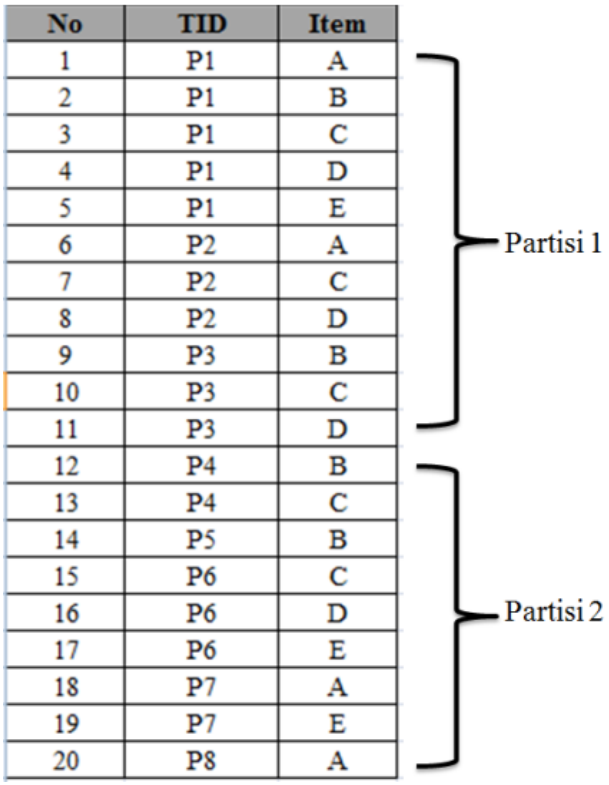

Figure 7 Dataset Partition Process Illustration

In Figure 7, the assumption is that we have a dataset with details of the volume or records being 20 records, while the items or attributes are 5 attributes, namely A, B, C, D, E, and there are eight customer transactions, namely P1, P2, P3, P4, P5, P6, P7, P8. This study separates the dataset into two partitions. With two partitions of 20 records that exist, it means that each partition consists of ten records. On the 11th record, the TID remains P3 or the 3rd customer. The 11th record must be placed in the 2nd partition to avoid TID duplication. The placement in the first partition for the 3rd customer is because the transaction is dominant. For all datasets, the dataset partitioning process is not explained in this study, considering that the datasets used have very large volumes and their attributes are numeric numbers.

4.4 TID-List Vertical Format Transformation

TID(Transaction Identifier)-List Vertical is a technique used to convert horizontal transaction formats to vertical formats. With this technique, the volume or record of the dataset can be reduced to make it easier to understand. Figure 8. shows the illustration.

TRA	SALT	RICE	TOMATOES	UNION	SALTED FISH	CHILE	VETSIN
P001	1	1	1	1	1	0	0
P002	1	1	1	0	1	1	1
P003	0	1	0	0	1	1	0
P004	1	0	1	1	0	1	1
P005	1	1	0	1	0	0	1
P006	1	1	1	1	1	0	0
P007	0	1	0	0	1	0	0
P008	1	1	1	1	1	1	1
P009	1	0	1	0	0	1	1
P010	1	0	1	1	0	1	1
FREK	8	7	7	6	6	6	6

NO	ITEM	TID	FREK
1	SALT	P001,P002,P004,P005,P006,P008,P009,P010	8
2	RICE	P001,P002,P003,P005,P006,P007,P008	7
3	TOMATOES	P001,P002,P004,P006,P008,P009,P010	7
4	UNION	P001,P004,P005,P006,P008,P010	6
5	SALTED FISH	P001,P002,P003,P006,P007,P008	6
6	CHILE	P002,P003,P004,P008,P009,P010	6
7	VETSIN	P001,P004,P005,P008,P009,P010	6

Figure 8 Illustration of Horizontal to Vertical Format Transform

In Figure 8, the dataset assumptions used are the same as the dataset assumptions used in Figure 4.1 regarding the illustration of dataset partitions. In the Apriori algorithm, the transaction table is generally represented in binary or binominal form, and the position of the item or attribute stretches horizontally; when the horizontal attribute format has a total of 8 records, the horizontal format is changed to a vertical format where the item or attribute is made up vertically. This technique can reduce the number of records, initially eight records, to 5 records, this can be done with the condition that the TID is more than the item. And this is very possible considering that datasets generally have more transaction records than their items or attributes.

4.5 Comparison of Apriori Runtime Algorithms Testing Result with TPQ-Apriori.

Figure 9 is a graphic description of the test results with the mushroom, chess, and the food mart, c20d10k dataset:

4.5.1 Comparison of 2-Itemset and 3-Itemset Runtime Test Result for Mushroom Dataset.

Figure 9 and Figure 10 show the Mushroom dataset test.

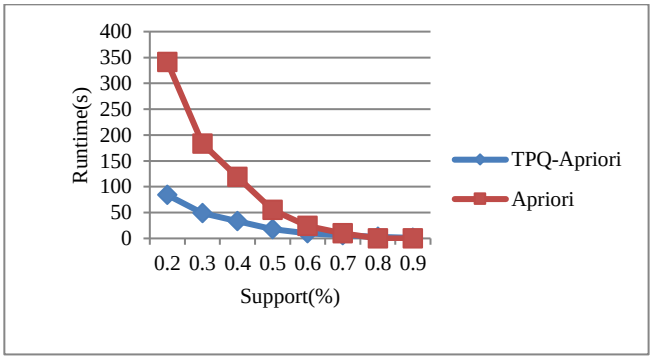

Figure 9 Comparison of TPQ-Apriori Runtime Test

Results With Traditional Apriori For 2-Itemset Mushroorm Datasets

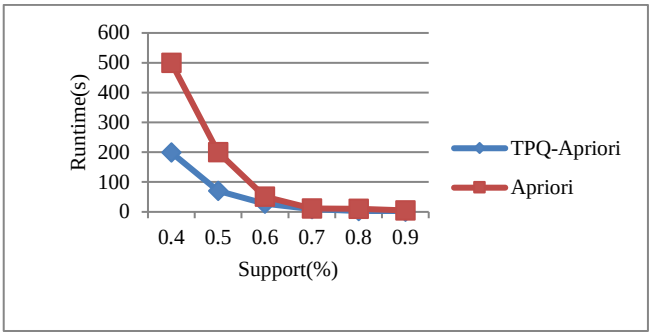

Figure 10 Comparison of TPQ-Apriori Runtime Test Results With Traditional Apriori For 3-Itemset Mushroom Datasets

The mushroom dataset test results show a huge time difference between TPQ-Apriori and traditional Apriori for support 0.2 and support 0.3. It is because the dataset has been partitioned for TPQ-Apriori so that dataset scans are faster, although the number of filtered attributes is not much different either carried out by TPQ-Apriori and traditional Apriori, namely 41 items filtered by TPQ-Apriori and 43 items filtered by traditional Apriori, but the distribution in the TID has decreased. Meanwhile, there is no significant time difference between support 0.6 to support 0.9 because the frequency of itemset has been reduced or trimmed by high minimum support.

Meanwhile, for the 3-Itemset, the larger the itemset to be scanned, the longer it will take. In the 2-itemset test for the mushroom dataset, the time required by traditional Apriori with support 0.2 is close to 350 seconds, while testing at the 3-itemset stage starts from support 0.4 instead of support 0.2 because with minimum support 0.4, it has reached 500 seconds, especially with more minimum support. low, of course it takes a lot of time. However, the results with TPQ-Apriori are still quite fast and stable because they are only about 200 seconds and have a significant time difference of up to 300 seconds with traditional Apriori for support 0.4

4.5.2 Comparison of 2-Itemset and 3-Itemset Runtime Test Results for Chess Datasets

For testing the Chess dataset, it can be seen in Figure 11 and Figure 12.

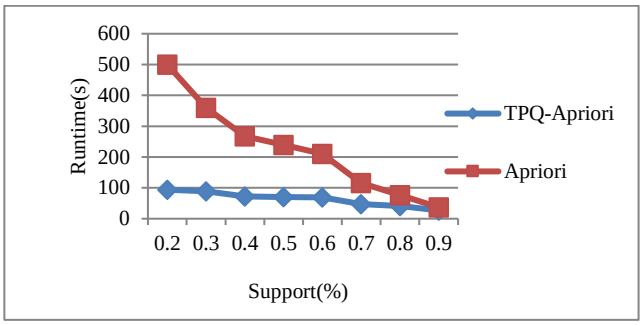

Figure 11 Comparison of TPQ-Apriori Runtime Test Results With Traditional Apriori For 2-Itemset Chess Datasets

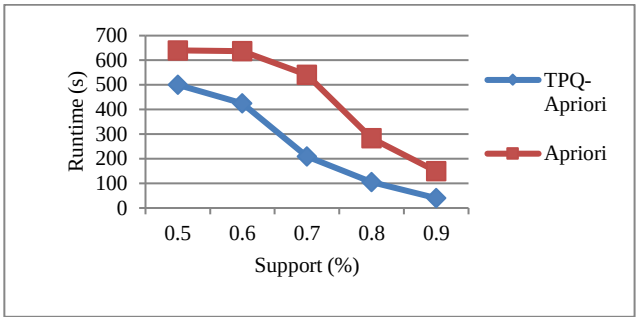

Figure 12 Comparison of TPQ-Apriori Runtime Test Results With Traditional Apriori For 3-Itemset Chess Datasets

From the test results using the chess dataset, there is a very large time difference between the TPQ-Apriori and the traditional Apriori starting from support 0.2 to support 0.6, as well as in the mushroom dataset test where the difference in frequency distribution causes the itemset frequency search process to be faster. However, in the chess dataset filtered items from the TPQ-Apriori, seven items are fewer than the traditional Apriori, where the TPQ-Apriori are 47 items while the traditional Apriori has 54 items. For support 0.2 and support 0.3, the difference is five items, support 0.4, as much as two items, while for support 0.5 and support 0.6, there is no difference.

While the scan time test for the chess dataset starts from support 0.5, this is because the characteristics of the chess dataset have an even frequency. So there are always many filtered attributes, which causes long dataset scan times. We can see in Figure 4.6 that support 0.5 and support 0.6 are still around 600 seconds more for traditional Apriori and approximately 500 seconds for TPQ-Apriori. Started down at support 0.7 and beyond.

4.5.3 Comparison of 2-Itemset and 3-Itemset Runtime Test Results for Foodmart Dataset

For testing, the Foodmart dataset, it can be seen in Figure 13 dan Figure 14.

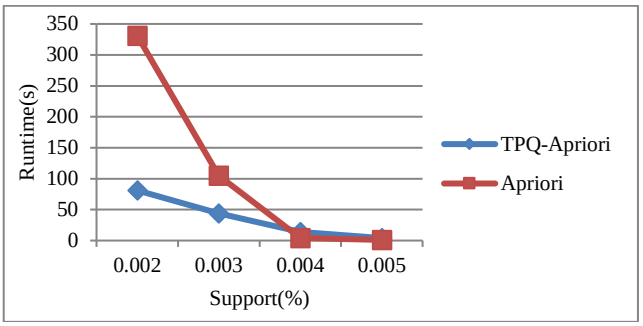

Figure 13 Comparison of TPQ-Apriori Runtime Test Results With Traditional Apriori For 2-Itemset Foodmart Datasets

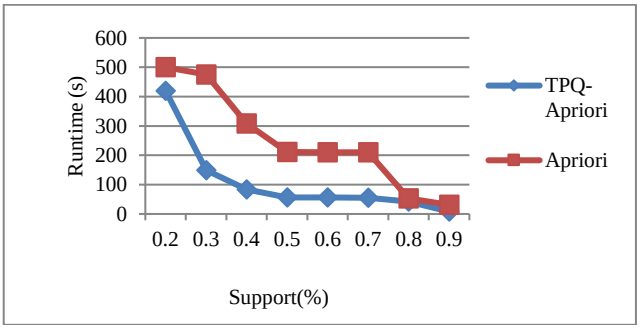

Figure 14 Comparison of TPQ-Apriori Runtime Test Results With Traditional Apriori For 2-Itemset Foodmart Datasets

The food mart dataset has quite different specifications from the three public datasets used. Mushroom, chess, and c20d10k have a large and even distribution of frequencies, but it is different with the food mart dataset, where the frequency distribution of itemsets in this dataset is very few and uneven. it has a very large number of items, namely 1,559 items. Because the frequency distribution of the itemset is small, the minimum support is set lower. For example, for the mushroom, chess, and c20d10k datasets, we use support 0.2, but the food mart dataset is 0.002.

Testing the c20d10k dataset is the same as the mushroom and chess dataset, but the support used is more from support 0.2 to support 0.9; there is a very significant time difference between support 0.2 and support 0.3 on the TPQ-Apriori results, but the traditional Apriori results do not occur, in fact, there is a significant time difference between support 0.3 and support 0.4. This is due to the dataset partitioning process resulting in differences in the number of attributes to be scanned even though they are still in a single database.

4.5.4 Perbandingan Hasil Pengujian Runtime 2-Itemset Dan 3-Itemset Untuk Dataset C20d10k.

For testing the C20d10k dataset, it can be seen in Figure 15 dan Figure 16.

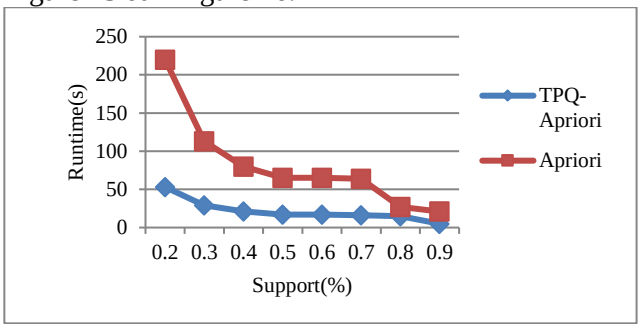

Figure 15 Comparison of TPQ-Apriori Runtime Test Results With Traditional Apriori For 2-Itemset C20d10k Datasets.

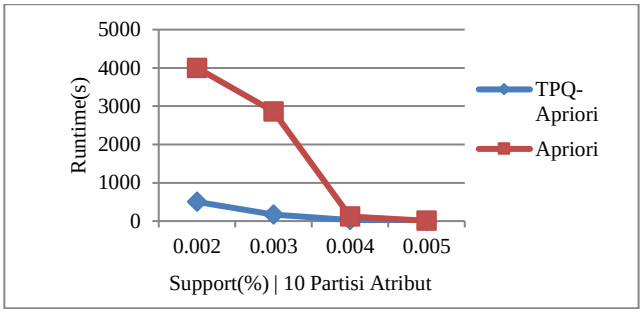

Figure 16 Comparison of TPQ-Apriori Runtime Test Results With Traditional Apriori For 3-Itemset C20d10k Datasets.

Scan time testing for the c20d10k dataset starts from support 0.2 to 0.9. There is a very significant time difference between the TPQ-Apriori and the traditional Apriori for support 0.2. In this case, the time difference is more than 100 seconds and gradually decreases at support 0.3 and support 0.4. Meanwhile, at support 0.5 to support 0.7, there is no change in the decline in time. It is because the items filtered at support 0.5 are the same as support 0.7, with 11 items for traditional Apriori and 11 items filtered with the same support for TPQ-Apriori, 11 items.

In testing, the food mart support dataset used 0.002 to 0.005, unlike the previous mushroom and chess datasets which used support 0.2 to support 0.9. This is because the frequency of scanned attributes in the dataset is small. However, this dataset has very many attributes, more than 1,000 attributes, so in this test, attribute partitioning is carried out by dividing ten partitions for a 3-item set and 20 partitions for a 2-item.

4.6 Comparison of ETDPC-Apriori Algorithm Runtime Test Results with TPQ-Apriori.

To compare the results of the ETDPC-Apriori runtime with the proposed method using three public datasets with characteristics as shown in Table 4.

Table 4 Characteristics of Public Datasets for ETDPC-Apriori Testing With TPQ-Apriori.

No	Dataset	ΣTransaction	ΣRecord	Attribute
1	Mushroom	8,124	186,825	119
2	Chess	3,196	118,252	75
3	C20D10k	10,000	200,000	192

This study uses more or less equivalent specifications or even lower hardware specifications, even though there are differences in memory, to make sure the proposed method works well. The difference in memory is more to keep the dataset scan process more stable. Information on hardware differences for testing can be seen in Table 5.

Table 5 Comparison of Hardware Specifications Used By Researchers With Hardware Specifications Used In This Research.

No	Komponen	ETDPC-Apriori Virtual	ETDPC-Apriori Physical	Our Research (this research)
1	Prosesor	Intel Xeon 2.30Ghz	Intel Xeon 2.00Ghz	Intel Dual Core G630 2.70Ghz
2	Memori	12GB	2GB	4GB

4.6.1 Comparison of Runtime Testing Results for the Mushroom Dataset.

The comparison of the results of the TPQ-Apriori and ETDPC-Apriori testing using the Mushroom dataset can be seen in Figure 17.

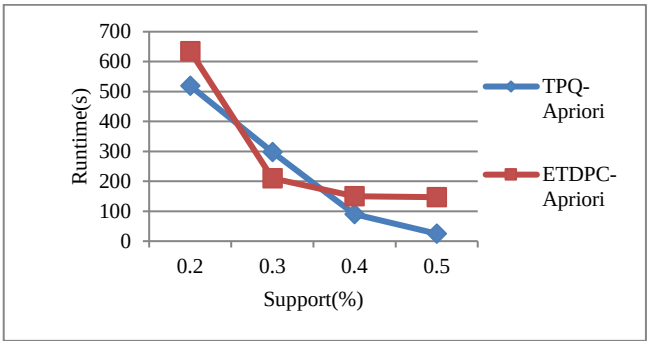

Figure 17 Comparison of TPQ-Apriori Runtime Test Results with ETDPC-Apriori for the Mushroom Dataset.

For the mushroom dataset at support 3.0 ETDPC-Apriori the scan time of the dataset is faster than TPQ-Apriori, but at support 0.2, support 0.4 and support 0.5 TPQ-Apriori is faster and even at support 0.5 it is almost close to zero seconds.

4.6.2 Comparison of Runtime Testing Results for the Chess Dataset

The comparison of TPQ-Apriori test results with ETDPC-Apriori using the Chess dataset can be seen in Figure 18.

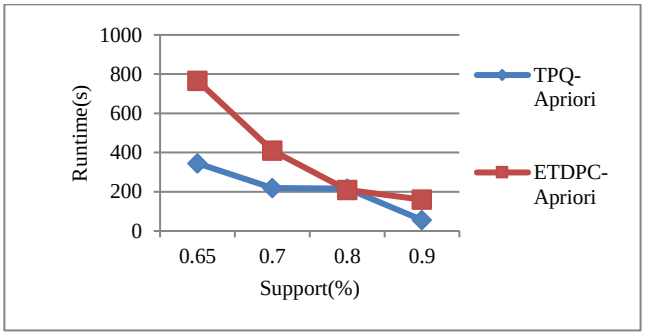

Figure 18 Comparison of TPQ-Apriori Runtime Test Results with ETDPC-Apriori for Chess Datasets.

For the chess dataset at support 0.65, there is a very significant time difference between TPQ-Apriori and ETDPC-Apriori where TPQ-Apriori is much faster as well as for support 0.7, but at support 0.8 it is almost the same but returns to support 0.9 TPQ-Apriori is faster.

4.6.3 Comparison of Runtime Testing Results for the C2010k Dataset.

The comparison of the results of the TPQ-Apriori and ETDPC-Apriori tests using the C20d10k dataset can be seen in Figure 19.

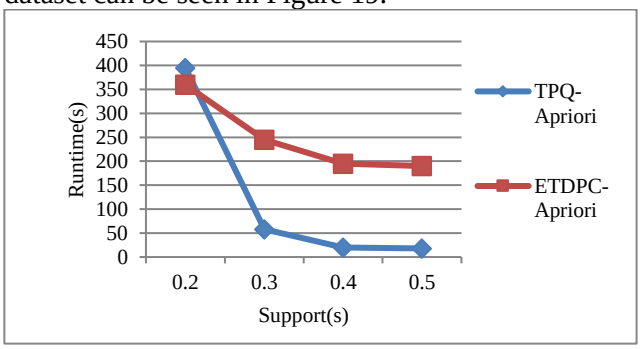

Figure 19 Comparison of TPQ-Apriori Runtime Test Results with ETDPC-Apriori for C20d10k Dataset.

For the C20d0k dataset at support 0.2, ETDPC-Apriori is faster. At the same time, TPQ-Apriori is not faster than ETDPC-Apriori this is due to the filtered items on TPQ-Apriori for support 0.2 more than ETDPC-Apriori. However, at support 0.3 to support 0.5 TPQ-Apriori is much faster.

5. Conclusion

The results of this study indicate that the developed TPQ-Apriori method can improve the performance of the dataset scan process, which is superior to the results of previous research conducted by Agrawal using the traditional Apriori method. The TPQ-Apriori method developed in this study is able to increase (accelerate) the time very significantly, which is twice as much as the traditional Apriori method, in scanning the 2-Itemset and 3-Itemset datasets on testing four datasets, where the 2-Itemset test Itemset starts from support 0.2 to support 0.9 and 3-Itemset starts from support 0.5 to support 0.9. The increase in the performance of the scan dataset process from the results of this study is also superior in processing speed up to 7% in scanning datasets when compared to the latest method developed by sudhakar singh, which uses ETDPC-Apriori, which both uses three datasets (mushroom, chess, and c20d10k) for testing.

The novelty of this study contributes to speeding up the process of finding the frequency of itemset in the Apriori algorithm and being able to produce optimal and consistent rules that previous researchers have never done.

The results of this study can encourage further research in the future in the form of library development and Application Programming Interface (API).

Conflicts of Interest

The authors declare no conflict of interest.

Author contributions

All authors participated in completing this study and writing this article. The level of roles and tasks of research work is the basis for placing each correspondent/first author and second author.

References

[1] C. C. Aggarwal, *Data Mining the Textbook*, vol. 14, no. 3. 2015. doi: 10.1016/0304-3835(81)90152-X.

[2] H. Witten, Frank, *Data Mining Practical Machine Learning*. Burlington USA: Morgan Koufmann.

[3] D. T. Larose, *Data Mining Methods and Models*, vol. 28. 2007. doi: 10.1017/S0068245400011199.

[4] A. Anggrawan, Mayadi, C. Satria, and L. G. R. Putra, "Scholarship Recipients Recommendation System Using AHP and Moora Methods," *Int. J. Intell. Eng. Syst.*, vol. 15, no. 2, pp. 260–275, 2022, doi: 10.22266/ijies2022.0430.24.

[5] J. H. Caru C Aggarwal, *Frequent Pattrn Mining*, vol. 9783319078. 2014. doi: 10.1007/978-3-319-07821-2_7.

[6] Fitria, Nengsih, and Qudsi, "Implementasi Algoritma FP-Growth Dalam Penentuan Pola Hubungan Kecelakaan Lalu Lintas," *J. Sist. Inf.*, vol. 13, no. 2, p. 118, 2017, doi: 10.21609/jsi.v13i2.551.

[7] X. Wu, *The Top Ten Algorithm in Data Mining*. 2008.

[8] W. Gan *et al.*, "Utility Mining across Multi-Dimensional Sequences," *ACM Trans. Knowl. Discov. Data*, vol. 15, no. 5, 2021, doi: 10.1145/3446938.

[9] S. M. El Abyad, M. M. Soliman, and K. M. El Sayed, "Deep Video Hashing Using 3DCNN with BERT," *Int. J. Intell. Eng. Syst.*, vol. 15, no. 5, pp. 113–127, 2022, doi: 10.22266/ijies2022.1031.11.

[10] J. Manimaran and T. Velmurugan, "Analysing the quality of association rules by computing an interestingness measures," *Indian J. Sci. Technol.*, vol. 8, no. 15, pp. 1–12, 2015, doi: 10.17485/ijst/2015/v8i15/76693.

[11] S. Agrawal, "Fast Algorithms for Mining Association Rules," *Jisuanji Gongcheng/Computer Eng.*, vol. 30, no. 15, p. 34, 2004.

[12] W. Z. Cheng and X. Li Xia, "A fast algorithm for mining association rules in image," *Proc. IEEE Int. Conf. Softw. Eng. Serv. Sci. ICSESS*, pp. 513–516, 2014, doi: 10.1109/ICSESS.2014.6933618.

[13] H. Xue, Q. Yang, and S. Chen, *The Top Ten Algorithms in Data Mining*, vol. 6, no. SVM. New York: CRC PRESS TAYLOR & FRANCIS GROUP, 2009. doi: 10.1007/s10115-007-0114-2.

[14] M. S. C. Jong Soon Park, "An Effective Hash-Based Algorithm for Mining Association Rules," *IBM Thomas J. Watson*.

[15] M. Redekopp and D. Kempe, *Hash Tables Dan Functions Dictionaries*. California US: USCViterbi.

[16] R. Rathinsabapathy and R. Bhaskaran, "Performance comparison of hashing

algorithm with Apriori," *ACT 2009 - Int. Conf. Adv. Comput. Control Telecommun. Technol.*, pp. 729–733, 2009, doi: 10.1109/ACT.2009.185.

[17] Zaki, "An Efficient Algorithm for Closed Itemset Mining."

[18] F. L. Ma, Yang, "An Improved Eclat Algorithm for Mining Association Rules Based on Increased Search Strategy," *Int. J. Database Theory Appl.*, vol. 9, no. 5, pp. 251–266, 2016, doi: 10.14257/ijdta.2016.9.5.26.

[19] E. Darrab, "Vertical Pattern Mining Algorithm for Multiple Support Thresholds," *Procedia Comput. Sci.*, vol. 112, pp. 417–426, 2017, doi: 10.1016/j.procs.2017.08.051.

[20] R. Patel, Chaudhari, Karan, "Optimization of Association Rule Mining Apriori Algorithm Using ACO," *Int. J. Emerg. Technol.*, vol. 2, no. 1, pp. 87–92, 2011.

[21] A. Kumar, Karang, "Improved Aprori Algorithm Based on bottom up approach using Probability and Matrix," *IJCSI Int. J. Comput. Sci. Issues*, vol. 9, no. 2, pp. 242–246, 2012, [Online]. Available: http://www.ijcsi.org/papers/IJCSI-9-2-3-242-246.pdf

[22] C. Kumar, "A Survey on Association Rule Mining using Apriori Algorithm," *Int. Conf. Adv. Comput. Commun. Technol. ACCT*, vol. 2015-April, no. 5, pp. 212–216, 2015, doi: 10.1109/ACCT.2015.69.

[23] S. Z. Chengqi Zhang, *Association Rules Mining*, vol. 1317. 1997.

[24] park j.s, "An Effective Hash Based Algorithm For Mining Association Rules." 1995," *ACM SIGMOD Rec.*, vol. 24, no. 2, pp. 175–186, 1995, doi: 10.1145/568271.223813.

[25] H. Sakai, M. Nakata, and J. Watada, "NIS-Apriori-based rule generation with three-way decisions and its application system in SQL," *Inf. Sci. (Ny).*, vol. 0, pp. 1–17, 2018, doi: 10.1016/j.ins.2018.09.008.

[26] X. Shang, "SQL Based Frequent Pattern Mining," 2005, [Online]. Available: http://diglib.uni-magdeburg.de/Dissertationen/2005/xueshang.htm

[27] A. C. Karthick Rajamani, "Efficient Mining for Association Rules with Relational Database Systems.pdf."

[28] M. Danubianu, S. Pentiuc, and I. Tobolcea, "Mining association rules inside a relational database - A case study," *Iccgi 2011*, no. c, pp. 14–19, 2011, [Online]. Available: http://www.thinkmind.org/index.php?view=article&articleid=iccgi_2011_1_30_10215

[29] U. Ahmed, J. C. W. Lin, G. Srivastava, R. Yasin, and Y. Djenouri, "An Evolutionary Model to Mine High Expected Utility Patterns from Uncertain Databases," *IEEE Trans. Emerg. Top. Comput. Intell.*, vol. 5, no. 1, pp. 19–28, 2021, doi: 10.1109/TETCI.2020.3000224.

[30] C. Chen, *Sql Implementation of Value Reduction*. Ohio USA: University Of Akron, 2014.

[31] M. Albarak, M. Alrazgan, and R. Bahsoon, "Identifying Technical Debt in Database Normalization Using Association Rule Mining," *2018 44th Euromicro Conf. Softw. Eng. Adv. Appl.*, pp. 437–441, 2018, doi: 10.1109/SEAA.2018.00077.

[32] W. Song and W. Ye, "Mining Unexpected Sequential Patterns from MOOC Data," *Proc. - 12th IEEE Int. Conf. Big Knowledge, ICBK 2021*, no. February, pp. 434–439, 2021, doi: 10.1109/ICKG52313.2021.00064.

[33] M. Mandeep, "Efficient Ordering Policy for Imperfect Quality Items Using Association Rule Mining," *Encycl. Inf. Sci. Technol. Third Ed.*, no. August 2014, pp. 773–786, 2014, doi: 10.4018/978-1-4666-5888-2.ch074.

[34] X. Yu, "Improvement of Eclat Algorithm Based on Support in Frequent Itemset Mining," *J. Comput.*, vol. 9, no. 9, pp. 2116–2123, 2014, doi: 10.4304/jcp.9.9.2116-2123.

[35] S. Zhang, Z. Du, and J. T. L. Wang, "New techniques for mining frequent patterns in unordered trees," *IEEE Trans. Cybern.*, vol. 45, no. 6, pp. 1113–1125, 2015, doi: 10.1109/TCYB.2014.2345579.

[36] S. Singh, R. Garg, and P. K. Mishra, "Performance optimization of MapRe duce-base d Apriori algorithm on Hadoop cluster R," *Comput. Electr. Eng.*, vol. 67, pp. 348–364, 2018, doi: 10.1016/j.compeleceng.2017.10.008.

[37] K. Amphawan, P. Lenca, and A. Surarerks, "Mining top-k regular-frequent itemsets using database partitioning and support estimation," *Expert Syst. Appl.*, vol. 39, no. 2, pp. 1924–1936, 2012, doi: 10.1016/j.eswa.2011.08.055.

[38] R. Agrawal, T. Imieliński, and A. Swami, "Mining Association Rules Between Sets of Items in Large Databases," *ACM SIGMOD Rec.*, vol. 22, no. 2, pp. 207–216, 1993, doi: 10.1145/170036.170072.

[39] P. Fournier, V. Jerry, C.-W. Lin, R. Nkambou, B. Vo, and V. S. Tseng Editors, *High-Utility Pattern Mining Theory, Algorithms and*

Applications. 2018. [Online]. Available: http://www.springer.com/series/11970

[40] P. Fjällström, *A way to compare measures in association rule mining*. Umea Swedia: Umea Universitet, 2016.