Partitioning Itemset on Transactional Data of Configurable Items for Association Rules Mining

Faiz Muhammad*, Rifkie Primartha*, Adi Wijaya, Muhammad Ihsan Jambak*

*Faculty of Computer Science, Sriwijaya University, Palembang, Indonesia
Informatics Engineering Department, Universitas MH Thamrin, Jakarta, Indonesia

Email: faizm26@hotmail.com, rifkie@ilkom.unsri.ac.id, adiwj@gmail.com, jambak@ilkom.unsri.ac.id

*corresponding author

Abstract. Association Rules Mining on transactional data is a very good way to help decision making on business strategy by finding what items people likely to buy together. In some cases, item is configurable, like PCs, cars, or gaming laptop, where people can choose what parts they want on their purchased item. This method can be used to determine what kind of configuration people most likely to choose and help seller on making business strategy. This kind of configurable item transactional data has one major difference from other transactional data, which is the fact that once one part in a category is selected, other parts in the same category will automatically be eliminated. This paper examines the uniqueness of these data and uses them to improve Association Rules Mining by proposing a method of partitioning itemset to dramatically reduce the number of scan needed to find frequent itemset, hence reduce execution time. From the experiment, the result shows that the proposed method can reduce the number of scan needed to generate frequent itemset while still delivering the same result, and when applied on FP-growth the proposed method generated the same rules as the standard FP-growth while still maintaining lower number of scan. The first section in your paper

1. Introduction

Transactional data is a list of items purchased by customer in one transaction, every item in this data is called itemset. Collecting these data is very useful to find relationship between items and help on making business strategy such as designing products bundle, planogram, which item needs to be restocked more often, and which to put on sale [1-4]. This transactional data is showed as table of every item available, and a mark of which of them exists on a transaction. Items on this table are not related to each other. But there is a unique kind of transactional item, where item appearance can affect other items appearance, which is transactional data on configurable items. Configurable items can be seen as bundle, where it consists of several classes of itemsets [5].

Take custom-built PC as an example, as one of the biggest market industries [6], there are vast selections of parts customer can pick, but for every kind of component, they will only need one. Like processor, motherboard, etc. This can be describe in the form of:

\[ i \rightarrow i. \text{ is every item available in } T \text{ transaction, there are group of items (P) consist of few items that are a part of } i \text{ in } T. \text{ where an } i \text{ in a } P \text{ presents, other } i \text{ in the same } P \text{ will not present.} \]
Data mining is the extraction of information or patterns that are important or interesting from the data residing on large data base which had been unknown but potentially useful information [7]. Association rule mining is one of many techniques in data mining. Association rules is the technique of finding the rules that govern association among items [8]. The rule is used to discover unknown relationship between items to help on decision making process. In general, association rule mining can be viewed as two-step process: the first one is finding all frequent itemset: this is done by counting how many times each item appears, each of those has to be at least as frequently as the predetermined minimum support count. And the second one is generating the strong association rules from the frequent itemset: where rules must satisfy the minimum support and minimum confidence that has been predetermined too [9].

There are many kinds of algorithm used in association rules mining, all with their own approach [10]. But they all do one same thing, which is calculating the appearance of each item to find the frequent itemset. This is done by scanning each item one by one for every transaction. And since each item’s appearance is as important as another, this one by one scan has to be done. So, for every X number of transactions with Y number of items, XY scans have to be done. The entire counting can be done several times depending on the algorithm, and this a waste of time.

Association rules mining in general also have some flaws, such as: (i) the whole database needs to be scanned, as mention above. (ii) the whole process of scanning does not get faster even though it needs to be done several items, it does not learn from the previous run [11].

The process of finding frequent itemset by checking each item on every transaction can be called as scan. There has been a study regarding on minimalizing the number of scan specific for Apriori algorithm, in this study the number of scan needed for frequent 2-itemset and frequent 3-itemset can be decreased to 0, while the scan needed for frequent 1-itemset remains the same as the standard Apriori algorithm [12].

In this paper, the uniqueness of transactional data of configurable items will be examined to be used to improve the process of the scan, which will improve the association rules mining in general regardless of what algorithm used in the process.

The rest of the paper is organized as follows. In section 2, the proposed method is presented. The experimental results of comparing the proposed method with the existing method are presented in section 3. Finally, our work of this paper is concluded in the last section along with the possible future works that can be done from this research.

2. MATERIAL AND PROPOSED METHOD
Dataset used in this study was obtained from custom-built computers database from pcpartpicker.com [14]. We took database of several computer builds from both Amd and Intel based computer from several generations with the total of 38 different parts present. We took only the main parts of the computer such as processor, ram, motherboard, graphic card, and PSU [15]. Every part on each build would be the itemset and the computer itself is the transactional data for this test.

We proposed a method of partitioning itemsets in transactional data of configurable item. Each partition consists of itemsets from the same category [13]. The idea of this partitioning method can be described in the form of:

\[ i_1 \ldots i_n \] is every item available in T transaction, there are group of items (P) consist of few items that are a part of i in T, where an i in a P presents, other i in the same P will not present.

By partitioning items based on their category on configurable items, the number of itemset scan done when finding frequent itemset can be greatly reduced simply by not continuing the scan of other items in the same category when an item in that category already present.
Table 1. Dataset Example

| I_1 | I_2 | I_3 | I_4 | I_5 | I_6 | I_7 | I_8 | I_9 |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| T1  | 1   | 0   | 1   | 0   | 0   | 0   | 1   | 0   |
| T2  | 0   | 1   | 0   | 1   | 1   | 0   | 0   | 0   |
| T3  | 0   | 0   | 0   | 0   | 0   | 1   | 1   | 0   |
| T4  | 0   | 1   | 1   | 1   | 0   | 0   | 0   | 0   |
| T5  | 0   | 1   | 0   | 0   | 1   | 0   | 0   | 0   |

Suppose there are 3 partitions (P_i) on this dataset:
P_1 = I_1, I_2, I_6
P_2 = I_3, I_7, I_5
P_3 = I_4, I_8, I_9

Table 2. Partitioned Dataset Example

| I_1 | I_2 | I_6 |
|-----|-----|-----|
| T1  | 0   | 0   |
| T2  | 0   | 1   |
| T3  | 0   | 0   |
| T4  | 0   | 1   |
| T5  | 0   | 1   |

| I_3 | I_5 |
|-----|-----|
| T1  | 1   |
| T2  | 0   |
| T3  | 0   |
| T4  | 1   |
| T5  | 0   |

| I_4 | I_8 |
|-----|-----|
| T1  | 0   |
| T2  | 1   |
| T3  | 0   |
| T4  | 1   |
| T5  | 0   |

Table 2. Partitioned Dataset Example

| I_1 | I_2 | I_6 |
|-----|-----|-----|
| T1  | 0   | 0   |
| T2  | 0   | 1   |
| T3  | 0   | 0   |
| T4  | 0   | 1   |
| T5  | 0   | 1   |

| I_3 | I_5 |
|-----|-----|
| T1  | 1   |
| T2  | 0   |
| T3  | 0   |
| T4  | 1   |
| T5  | 0   |

| I_4 | I_8 |
|-----|-----|
| T1  | 0   |
| T2  | 1   |
| T3  | 0   |
| T4  | 1   |
| T5  | 0   |

Notice on P1 in T1, when I1 appears, I2 and I6 does not appear. Same thing happens on others transaction in every partition.

This partitioning method is done before the association rules mining process began. And can be used in several association rules mining algorithm with a simple change on their frequent itemset counting method. And since every association rules mining algorithm has the same way of counting frequent itemset, this change can be easily applied on several association rules mining algorithm that are already exists today. Figure 1 describe how the proposed method applied in flowchart:
The existing frequent itemset counting algorithm used in traditional association rules mining algorithm is as follows:

\( T \): number of transaction

\( I \): number of itemset

**Figure 1.** Flowchart of proposed method

```
for(i=0;i<T;i++)
for(j=0;j<Ip;j++)
if(itemset[i][j]==1)
support[j]++
```

**Figure 2.** Existing frequent itemset counting algorithm

The proposed change on frequent itemset counting algorithm for the partitioning method aims to reduce number of scan by skipping the unnecessary scans. The changes are as follows:

\( T \): number of transaction

\( Ip \): number of itemset in a partition

```
for(i=0;i<T;i++)
for(j=0;j<Ip;j++)
if(itemset[i][j]==1)
support[j]++
break
```

**Figure 3.** Frequent itemset counting algorithm in proposed method
The existing algorithm scans the whole itemset in each individual transaction while the proposed change scans the itemset on every partition of each transaction. The main focus on the proposed change is to find at one item appearance on every partition on each transaction. After counting each appearance of itemset, it will select which itemset satisfies the minimum support count for the association mining process and eliminate itemset that does not.

After all the scanning process completed, the support count has been generated, the whole process has to be done again. Based on what algorithm used in association mining rules, this need to be done several times. For example, on standard Apriori algorithm, it needs to scan for 1-itemset, 2-itemset, and so on until all possible rules generated [12]. In this proposed method, we put a process of reordering itemsets on each partition based on their frequency. On table 2, we saw the data have been partition based on their category. On table 3 we see the result of frequent itemset counting:

**Table 3. Frequent Itemset Result**

| Item set | Frequency |
|----------|-----------|
| I_1      | 1         |
| I_2      | 3         |
| I_6      | 1         |
| I_3      | 2         |
| I_7      | 1         |
| I_5      | 2         |
| I_4      | 2         |
| I_8      | 1         |
| I_9      | 2         |

After that, the proposed method will reorder itemsets on each partition as shown in table 4 :

**Table 4. Partitioned Dataset Example (Reordered)**

|   | I_2 | I_1 | I_6 |
|---|-----|-----|-----|
| T1 | 0   | 1   | 0   |
| T2 | 1   | 0   | 0   |
| T3 | 0   | 0   | 1   |
| T4 | 1   | 0   | 0   |
| T5 | 1   | 0   | 0   |

|   | I_3 | I_1 | I_2 |
|---|-----|-----|-----|
| T1 | 1   | 0   | 0   |
| T2 | 0   | 1   | 0   |
| T3 | 0   | 0   | 1   |
| T4 | 1   | 0   | 0   |
| T5 | 0   | 1   | 0   |

|   | I_4 | I_9 | I_8 |
|---|-----|-----|-----|
| T1 | 0   | 0   | 1   |
| T2 | 1   | 0   | 0   |
| T3 | 0   | 1   | 0   |
| T4 | 1   | 0   | 0   |
| T5 | 0   | 1   | 0   |

3. Experimental Result
The experiments are conducted using a computer platform based on Intel G4560 3.5 GHz CPU, with 8GB of DDR4 ram and Windows 10 Pro 1151 as operating system. The program used on this testing
is NetBeans version 8.2. The measurement on this testing is the number of scans needed to generate frequent itemset.

The first step is partitioning itemsets in the database by these categories: processor, ram, motherboard, graphic card, and PSU. The detail of this research flow is shown on figure 2:

![Research Workflow](image)

**Figure 4. Research Workflow**

After the partitioning complete. We began the test by counting the frequent itemset from dataset. We then compare the number of scans needed from the existing frequent itemset counting algorithm with the proposed method. The result is as follows:
The proposed method shows a significantly smaller number of scans while still generating the exact same result of frequent itemset as the existing method. Based on the test we also examine that if itemsets on each partition is reordered by sorting itemsets by their frequency from highest to the lowest, the number of scan will be reduce even more. This sorting can be done after the frequency of all itemset are counted. The result after sorting is as follows:

The result shows that the second run of the proposed method does even less scan than the existing method. The changes on itemset order can be saved to be used on the next itemset scan.

We used the proposed method on this paper on FP-growth association rules mining algorithm. And then we compare both results from FP-growth with standard itemset frequency counter algorithm with FP-growth with our proposed method. The result shows that FP-growth with our proposed method
generated the same rules as the enhanced FP-growth, this is because the proposed method does not interfere with the rules generating process. The results are as follows:

**Table 5. Generated Rules**

| Premises          | Conclusion | Premises          | Conclusion |
|-------------------|------------|-------------------|------------|
| GTX 1050Ti        | G4560      | GTX 1050Ti        | G4560      |
| G4560             | GTX 1050Ti | G4560             | GTX 1050Ti |
| 8GBDDR42400, GTX 1050Ti | G4560 | 8GBDDR42400, GTX 1050Ti | G4560 |
| 8GBDDR42400, G4560 | GTX 1050Ti | 8GBDDR42400, G4560 | GTX 1050Ti |
| 500W, 8GBDDR42400 | 500W       | 8GBDDR42400       | 8GBDDR42400 |
| GTX 1050Ti, G4560 | 1050Ti, G4560 | GTX 1050Ti, G4560 | 8GBDDR42400 |
| B250M             | 8GBDDR42400| B250M             | 8GBDDR42400 |

The number of scan needed to get the itemset frequencies on both tests are as follows:

**Table 6. Number of Scan Comparison (FP-Growth VS FP-Growth + Proposed Method)**

| Standard FP-growth | FP-growth + proposed method |
|--------------------|-----------------------------|
| 950 itemset scans  | 415 itemset scans           |

**Table 7. Number of Scan Comparison (FP-Growth VS FP-Growth + Proposed Method) – (Reordered Itemset)**

| Standard FP-growth | FP-growth + proposed method |
|--------------------|-----------------------------|
| 950 itemset scans  | 315 itemset scans           |

The number of scan result show the same number as the previous frequency itemset counting from test on dataset 3. This proved that the proposed method does not interfere with the rules generating process in FP-growth algorithm while improves the itemset frequency counting process in it.

4. Conclusion

The experimental result shows that the proposed method can reduce the number of scan significantly while still generating the same result as the existing method. The experiment also reveals that if the proposed method done several times with reordering the itemset of each partition based on their frequency from the previous run in between runs, the number of scan will decrease even more, which is very suitable for association rules algorithms that need to scan for itemset frequency more than once. This is also good for market basket analysis where the transactional data gets updated regularly while list of item stays the same. The mining process can be done several times to keep the result up to date based on the current data and the process will get faster on each run. This proposed method is apt for market analysis specifically for configurable items.

The future works will be concerned on the limitation of this proposed method that it only works with transactional data of configurable items. Negative association mining can be used to partition any transactional dataset by finding items that has no association to one another.
References

[1] B. Gayathri, "Efficient market basket analysis based on FP-Bonsai," in I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC), 2017 International Conference on, 2017, pp. 788-792.

[2] P. Giudici and S. Figini, "Market basket analysis," Applied Data Mining for Business and Industry, 2009.

[3] P. Pukach and K. Shakhovska, "The mathematical method development of decisions supporting concerning products placement based on analysis of market basket content," in Experience of Designing and Application of CAD Systems in Microelectronics (CADSM), 2017 14th International Conference The, 2017, pp. 347-350.

[4] A. Setiapawan, G. Satiabudhi, D. H. Setiabudi, and R. Djunaidy, "Data Mining Applications for Sales Information System by Using Market Basket Analysis on Stationery Company," Petra Christian University, 2017.

[5] Y. Fang, X. Xiao, X. Wang, and H. Lan, "Customized Bundle Recommendation by Association Rules of Product Categories for Online Supermarkets," in 2018 IEEE Third International Conference on Data Science in Cyberspace (DSC), 2018, pp. 472-475.

[6] P. Sirisawat, T. Kiatcharoenpol, N. Choomrit, and P. Wangphanich, "A study of reverse logistics practices: A case study of the computer parts industry in Thailand," in Industrial Engineering and Engineering Management (IEEM), 2015 IEEE International Conference on, 2015, pp. 356-360.

[7] E. Miranda, "Pengembangan Business Intelligence Bagi Perkembangan Bisnis Perusahaan," CommIT (Communication and Information Technology) Journal, vol. 2, pp. 111-116, 2008.

[8] L. Kusrini, "ET, 2009, "Algoritma Data Mining"," Andi Offset: Yogyakarta.

[9] J. Han, J. Pei, and M. Kamber, Data mining: concepts and techniques: Elsevier, 2011.

[10] W. Nengsih, "A comparative study on market basket analysis and apriori association technique," in Information and Communication Technology (ICoICT), 2015 3rd International Conference on, 2015, pp. 461-464.

[11] M. Al-Maolegi and B. Arkok, "An improved apriori algorithm for association rules," arXiv preprint arXiv:1403.3948, 2014.

[12] S. Tirumalasetty, A. Jadda, and S. R. Edara, "An Enhanced Apriori Algorithm for Discovering Frequent Patterns with Optimal Number of Scans," arXiv preprint arXiv:1506.07087, 2015.

[13] Y. Yang and H. Peng, "Modular Order Picking Approach Based on FP-tree Algorithm," in Computational Intelligence and Design (ISCID), 2014 Seventh International Symposium on, 2014, pp. 455-458.

[14] PCPARTPICKER. (2018). Computer Builds [Online]. Available: pcpartpicker.com/builds/

[15] L. Jacobson, "Computer System Components: Computer Parts & Functions", Business 104: Information Systems and Computer Applications, 2018.