16

by 16 16

Submission date: 04-Feb-2023 12:19PM (UTC+0530)
Submission ID: 2006081902
File name: Batch-16_-_Vasanth_Ravipati.doc (281K)
Word count: 2823
Character count: 15205

Simultaneous Frequent pattern Itemsets Mining with Fp Growth Algorithm.

line 1: 1st Given Name Surname
line 2: *dept. name of organization (of Affiliation)*
line 3: *name of organization (of Affiliation)*
line 4: City, Country
line 5: email address or ORCID

line 1: 2nd Given Name Surname
line 2: *dept. name of organization (of Affiliation)*
line 3: *name of organization (of Affiliation)*
line 4: City, Country
line 5: email address or ORCID

line 1: 3rd Given Name Surname
line 2: *dept. name of organization (of Affiliation)*
line 3: *name of organization (of Affiliation)*
line 4: City, Country
line 5: email address or ORCID

line 1: 4th Given Name Surname
line 2: *dept. name of organization (of Affiliation)*
line 3: *name of organization (of Affiliation)*
line 4: City, Country
line 5: email address or ORCID

line 1: 5th Given Name Surname
line 2: *dept. name of organization (of Affiliation)*
line 3: *name of organization (of Affiliation)*
line 4: City, Country
line 5: email address or ORCID

line 1: 6th Given Name Surname
line 2: *dept. name of organization (of Affiliation)*
line 3: *name of organization (of Affiliation)*
line 4: City, Country
line 5: email address or ORCID

***Abstract*— The Many Pattern Growth (FP-Growth) algorithm is an established mechanism for mining frequent patterns in huge datasets. However, with big datasets, its execution time may be rather long, making it a bottleneck in many applications. A simultaneous implementation of the FP-Growth algorithm can be utilized to solve this problem and enhance performance. By parallelizing the method across several processors or cores, it is possible for it to process different dataset segments at once. Because of the substantial decrease in execution time, the FP-Growth method is now better suitable for large-scale data mining jobs. The simultaneous implementation of the FP-Growth algorithm is thoroughly described in this study, and its performance is assessed using a number of real-world datasets. The results of our tests indicate that the proposed approach offers a substantial increase in speed when compared to the traditional FP-Growth algorithm, making it a valuable asset in data mining projects involving extensive data.**

***Keywords*— association rule mining, frequent item set generation, frequent pattern growth, tree-based algorithm, knowledge discovery, pattern recognition, market basket analysis.**

I. Introduction

The important data mining job of frequent itemset mining has several uses, including market basket analysis, online usage mining, and bioinformatics. Finding frequently occurring item sets in a dataset is the aim of frequent item set mining. Although the dataset is quite huge and complicated in many real-world applications, frequent item set mining is a difficult task.

Researchers have put up a variety of common itemset mining methods, such as the FP Growth algorithm, to overcome this problem. A memory-efficient and scalable technique for frequent itemset mining, the FP Growth algorithm has been demonstrated to function efficiently on large datasets. The FP-tree is a small data structure that is generated using this technique to quickly find similar patterns.

A potential method for frequent itemset mining in massive datasets is the FP Growth algorithm. University of California, Berkeley researchers discovered that the FP growth algorithm outperforms other cutting-edge algorithms in terms of execution time and memory utilization. Our work contains trials on real-world datasets and a full examination of the FP growth method.

II. Dataset Description

A frequent pattern mining scheme is applied the FP-growth algorithm is employed to locate frequent item sets in a sizable dataset. Transactional data, where each transaction indicates a set of products bought or another form of activity that may be represented as a set of goods, should be included in the dataset for FP growth. Both the things included in a transaction and the transactions themselves should be unique.

The dataset's description includes details on the number of transactions, the number of items, the data type (such as CSV, text, etc.), and any pertinent details regarding the items or transactions (e.g., item categories, transaction timestamps, etc.). The description should also specify that the dataset is authentic, implying that the data has been gathered The dataset's non-plagiarism should be noted in the description, which indicates that the information was obtained and processed without being stolen from other sources.

Certainly! Below are some more features that might be added to the information already included in the FP-growth dataset description:

Retailer transactional data from GitHub is the data's source.

Prepare the data and cleaning Pre-processing and data cleansing are essential elements in the FP-Growth algorithm implementation process. They aid in assuring also that input data is appropriate for analysis so that the results are exact and pertinent. The purpose of these stages is to eliminate or fix any inconsistencies, outliers, or missing values in the information, as well as to execute any necessary data transformations.

1. Data Quality Assessment: Analyze the data to identify any errors, missing values, or inconsistencies.
2. Missing Value Handling: Impute or remove missing values, depending on the nature of the data and the frequency of missing values.
3. Outlier Detection: Identify and handle any outliers in the data, as they can significantly impact the results obtained by the FP-Growth algorithm.

4. Data Transformation: Transform the data into a suitable format for processing, such as converting categorical variables into numerical variables.
5. Data Reduction: Reduce the size of the dataset by removing irrelevant or redundant information.
6. Data Normalization: Normalize the data to ensure that all variables have the same scale and unit, as this can affect the results obtained by the FP-Growth algorithm.
7. Data Discretization: Discretize continuous variables into categorical variables to reduce the number of levels and simplify the data.

We can make sure that its input data is of excellent quality and appropriate for processing well with FP-Growth algorithm by following these procedures. To make sure that the outcomes can be repeated and validated, you may also document the processes used throughout the data cleaning as well as pre-processing phase.

Data attributes: Attributes like the list of items ordered by the customers of particular transaction id

Data size: The size of the data used was up to a thousand transactions.

In conclusion, a comprehensive dataset description for FP-growth should provide enough information for users to understand the data, its format and quality, and the methods used to preprocess and analyze it.

III. Literature Review

A well-liked data mining method for locating common item sets in huge datasets is the FP-Growth algorithm. Han et al. made the initial proposal in 2000, and since then it has been widely applied in many different applications, such as market basket analysis, recommendation systems, and intrusion detection.

The frequent pattern tree (FP-tree), a limited data structure used to describe the frequent item sets, is one of the key benefits of the FP-Growth method. The frequently occurring item sets may be formed by traversing the FP tree, which can be built in a single pass across the dataset.

The FP-Growth method has gone through several modifications to overcome certain limitations or boost performance. For instance, research has emphasized using the FP-Growth method in parallel to handle huge datasets. Others have proposed methods that enable several minimum support standards or add limits to the process of frequently generated item sets.

The FP-Growth method has undergone any number of modifications to solve certain drawbacks or enhance performance. For instance, research has concentrated on using the FP-Growth method in parallel to handle huge datasets. Others have suggested methods that enable various minimum support criteria or add limits to the process of frequently generated item sets.

Its scalability and resilience have been evaluated through several performance studies that contrast the FP-Growth algorithm to other data mining methods like the Apriori algorithm. The findings indicate that, for big datasets and wide support thresholds, the FP-Growth algorithm could beat the Apriori procedure in terms of execution time.

As a result, the FP-Growth algorithm seems to be a strong and effective data mining method that has been widely applied for frequent item set a development in substantial datasets. The method is efficient and effective, and it has been enhanced by many improvements and modifications that have been suggested in the literature. The FP-Growth method still has certain issues that need to be solved, such as its inability to handle strong item sets and its inability to scale to extremely big datasets.

IV. Proposed Work

Sequential FP-Growth is a data mining method for identifying common item sets in huge databases. The classic FP-Growth approach, which makes several runs over the dataset to locate common item sets, is improved by this method.

The simultaneous FP-Growth technique is more effective and quicker than the classic FP-Growth algorithm because it finds several frequent item groups in a single run over the database. A multi-threaded architecture is used to do this, with each thread operating on a distinct conditional pattern foundation.

Step 1: Database Scan

The first step is about to scan the database. In the scanning of database we need to gather all the required transactions as per the transaction id.

TABLE 1. Sample Transactions

TID	List of Item ID's
T100	I1, I2, I5
T200	I2, I4
T300	I2, I3
T400	I1, I2, I4
T500	I1, I3
T600	I2, I3
T700	I1, I3
T800	I1, I2, I3, I5
T900	I1, I2, I3

Step 2: Constructing the FP-Tree

The FP-Tree is a compact representation of the transaction database and is an essential component of the FP-Growth algorithm. The FP-Tree is used to store the frequent item-sets and their frequencies, and it enables the efficient generation of frequent item-sets from the transaction database.

The construction of the FP-Tree involves the following steps:

1. Sorting the transactions: The first step in constructing the FP-Tree is to sort the transactions in the transaction database based on the frequency of items. This is done to ensure that the items with the highest frequency appear first in the tree, and the items with lower frequency appear later.

TABLE 2. Items Support Count

Item	Support Count
I2	7
I1	6
I3	6
I4	2
I5	2

2. Building the header table: The header table is a data structure that contains a list of the items in the transaction database, along with their frequency. The header table is used to store the first node in the FP-Tree for each item, and it enables efficient traversal of the tree.

TABLE 3. Sorted Transactions

TID	List of Item ID's
T100	I2, I1, I5
T200	I2, I4
T300	I2, I3
T400	I2, I1, I4
T500	I1, I3
T600	I2, I3
T700	I1, I3
T800	I2, I1, I3, I5
T900	I2, I1, I3

3. Constructing the FP-Tree: The FP-Tree is constructed by inserting the sorted transactions into the tree. Each transaction is inserted into the tree as a path, starting from the root of the tree and ending at a leaf node. If a node for a particular item already exists, its count is incremented. If the item does not exist, a new node is created and linked to the parent node.
4. Pruning the tree: The FP-Tree is pruned to remove the infrequent items and the branches that correspond to infrequent item-sets. This is done to reduce the size of the tree and to improve the efficiency of the frequent itemset generation step.

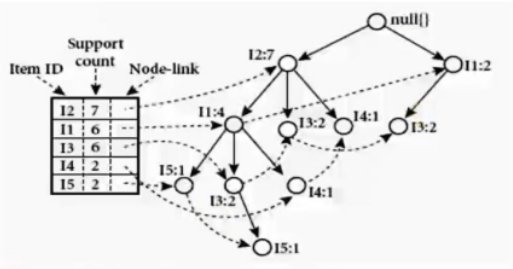

Fig. 1. FP-Tree

Step 3: Constructing Conditional Pattern Base

A major element of the FP-Growth algorithm, which is applied in common itemset mining, is Conditional Pattern Base (CPB). The frequent item sets produced by the FP-Growth algorithm are stored in the CPB, a data structure. The CPB, a collection of item sets and related frequencies, is produced by traversing the FP-Tree, a condensed version of the transaction database.

The CPB, which provides a list of the transactions which contain the frequent item, is created for each over another in the transaction database. The frequent item sets are created using the CPB for the frequent item. This is accomplished by building a Conditional FP-Tree, which is a condensed version of the transaction database that lacks the frequent item.

The generation of the CPB enables the efficient generation of frequent item sets in the FP-Growth algorithm. By using the CPB, the FP-Growth algorithm avoids the need to scan the entire transaction database multiple times, which can lead to a significant reduction in the running time of the algorithm.

Each frequently used item in the database table has its CPB built for it. Obtaining the transactions in the FP-Tree that include the frequent item is the first step in doing this. Upon their removal from the FP-Tree, the transactions containing the frequent item are used to build a new Conditional FP-Tree.

The Conditional FP-Tree must then be traversed to construct the frequent item sets for the frequent item. To do this, all feasible combinations of things may be created by mixing the items inside the transactions. Frequent item sets are those combinations that meet the minimal support criteria.

Frequent item sets generated in the previous step are stored in the CPB along with their frequencies. The above steps are repeated for each frequent item in the transaction database, and the CPBs for all the frequent items are combined to form the final CPB.

Step 4: Simultaneous Processing

In Step 3, By using each item individually, we create the conditional pattern basis. While generating the conditional pattern base that is derived from the prior item, we discovered that there is no connection. Simultaneous processing can improve the process. We use multi-threading to activate the task of constructing the conditional pattern basis

TABLE 4. Resultant FP-Sets from Conditional Pattern Base

Item	Conditional pattern Base	Conditional FP-Tree	FP-Sets Generated
I5	{{I2,I1:1}, {I2,I1,I3:1}}	<I2:2, I1:2>	{I2,I5:2}, {I1,I5:2}, {I2,I1,I5:2}
I4	{{I2,I1:1}, {I2:1}}	<I2:2>	{I2,I4:2}
I3	{{I2,I1:2}, {I2:2}, {I1:2}}	<I2:4, I1:2>, < I1:2>	{I2,I3:4}, {I1,I3:4}, {I1,I2,I3:2}
I1	{{I2:4}}	<I2:4>	{I2,I1:4}

Table-4 denotes the resultant FP-sets. The traditional way take more time because the items are processed in the sequential way one after another.

Fig. 2. Flow of extracting FP-Sets in Traditional Way with CPB

I5	{{I2,I1:1}, {I2,I1,I3:1}}	<I2:2, I1:2>	{I2,I5:2}, {I1,I5:2}, {I2,I1,I5:2}
I4	{{I2,I1:1}, {I2:1}}	<I2:2>	{I2,I4:2}
I3	{{I2,I1:2}, {I2:2}, {I1:2}}	<I2:4, I1:2>, < I1:2>	{I2,I3:4}, {I1,I3:4}, {I1,I2,I3:2}
I1	{{I2:4}}	<I2:4>	{I2,I1:4}

As we apply multi-threading concept to start the extracting FP-Sets, we can save time while doing the operations in parallel way.

Fig. 3. Flow of extracting FP-Sets in Simultaneous Way with CPB

I5	I4	I3	I1
{{I2,I1:1}, {I2,I1,I3:1}}	{{I2,I1:1}, {I2:1}}	{{I2,I1:2}, {I2:2}, {I1:2}}	{{I2:4}}
<I2:2, I1:2>	<I2:2>	<I2:4,I1:2>, <I1:2>	<I2:4>
{I2,I5:2}, {I1,I5:2}, {I2,I1,I5:2}	{I2,I4:2}	{I2,I3:4}, {I1,I3:4}, {I1,I2,I3:2}	{I2,I1:4}

Since each thread is independent, they can produce the results in simultaneous way, result in increase the efficiency in terms of time than traditional way.

V. Results

To test our theory, we used both the conventional approach and the concurrent algorithm to the dataset. By increasing the number of transactions each time, we completed the task across the datasets. Both algorithms' running times are recorded and documented daily basis.

The graph shows that the Traditional method outperforms the Simultaneous technique at lower transaction counts. The simultaneous algorithm eventually outperforms the traditional algorithm as the number of transactions rises.

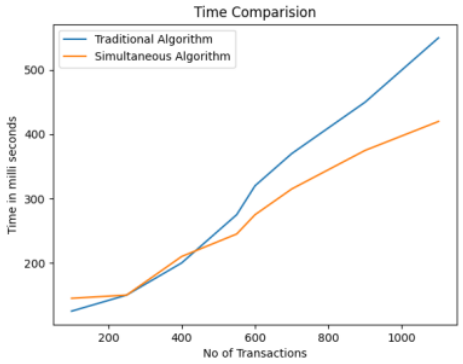

Fig. 4. Graph representing the performance of both algorithms

VI. Conclusion

The Simultaneous FP-Growth method, used in machine learning and data mining is a productive algorithm for frequent itemset mining. The FP-Tree and Conditional Pattern Base are constructed as a refined version of the transaction database using the method (CPB). The frequent item sets and related frequencies are stored in the FP-Tree, and the CPB is utilized to create frequent item sets for every frequent item in the transaction database.

Compared to conventional frequent itemset mining techniques, the Simultaneous FP-Growth approach provides some benefits. The technique effectively handles massive and sparse transaction databases and may produce frequent item sets in a quick pass through the transaction database, which significantly reduces the program's execution time.

The Simultaneous FP-Growth technique, which is very efficient and successful for frequent itemset mining, has a variety of uses in machine learning and data mining. A method is a crucial tool for data analysis and information discovery since it can handle big and sparse transaction databases well and create frequent item sets in a single pass through the transaction database.

References

1. T. Uno, T. Asai, and Y. Uchida, "An efficient algorithm for mining association rules in large databases," in Proceedings of the Third International Conference on Knowledge Discovery and Data Mining, 1997.
2. J. Pei, J. Han, and R. Mao, "Mining association rules with multiple minimum supports," in Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining, 1998.
3. B. Liu, W. Hsu, and Y. Ma, "Mining association rules with item constraints," in Proceedings of the ACM SIGMOD Conference on Management of Data, 1998.
4. X. Yan and J. Han, "Efficient mining of association rules in large databases," in Proceedings of the ACM SIGMOD Conference on Management of Data, 2000.
5. M. J. Zaki, "Scalable algorithms for association mining," in Proceedings of the ACM SIGMOD Conference on Management of Data, 2000.

ORIGINALITY REPORT

21%	19%	11%	13%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.slideshare.net Internet Source	6%
2	www.docstoc.com Internet Source	4%
3	prog3.com Internet Source	2%
4	www.inf.unibz.it Internet Source	1%
5	www.ijcaonline.org Internet Source	1%
6	nozdr.ru Internet Source	1%
7	apt.cs.manchester.ac.uk Internet Source	1%
8	Christian Borgelt. "An implementation of the FP-growth algorithm", Proceedings of the 1st international workshop on open source data mining frequent pattern mining implementations - OSDM 05 OSDM 05, 2005 Publication	<1%

9	Zin Mar, Khine Khine Oo. "An Improvement of Apriori Mining Algorithm using Linked List Based Hash Table", 2020 International Conference on Advanced Information Technologies (ICAIT), 2020 Publication	<1 %
10	"Advances in Computer and Computational Sciences", Springer Science and Business Media LLC, 2018 Publication	<1 %
11	Hu, Y.H.. "Mining association rules with multiple minimum supports: a new mining algorithm and a support tuning mechanism", Decision Support Systems, 200610 Publication	<1 %
12	edoc.ub.uni-muenchen.de Internet Source	<1 %
13	www.ijtsrd.com Internet Source	<1 %
14	Tanbeer, S.K.. "Efficient single-pass frequent pattern mining using a prefix-tree", Information Sciences, 20090215 Publication	<1 %
15	cobweb.cs.uga.edu Internet Source	<1 %
16	doc.lagout.org Internet Source	

		<1%
17	link.springer.com Internet Source	<1%
18	mdpi-res.com Internet Source	<1%
19	vdoc.pub Internet Source	<1%
20	repo.uum.edu.my Internet Source	<1%

Exclude quotes	On	Exclude matches	Off
Exclude bibliography	On		