Consumer Purchase Patterns using File Structures

Swarna Kadagadkai
Information Science and Engineering
JSS Academy of Technical Education
Bengaluru, India

Sahana V
Information Science and Engineering
JSS Academy of Technical Education
Bengaluru, India

Malini M Patil
Information Science and Engineering
JSS Academy of Technical Education
Bengaluru, India

***Abstract* - Consumer Purchase Patterns has seen an eclectic range of algorithms that are implemented to improve its efficiency and optimal analysis. One such is the well-known Apriori Algorithm specifically used for association analysis based on generating frequent itemsets by reading transaction data for a given time period. This task in itself is computationally quite expensive and it has been refactored in multiple ways to mitigate the cost. This paper aims to study the working of this algorithm on market basket data using files to store the transaction data. Analysis is made using experimental results.**

Keywords* - *Apriori, Static Hashing, Dynamic Hashing, File Structures, Frequent item sets, Market basket data.

I. INTRODUCTION

Consumer behavior analysis refers to the investigation of the process involved while consumers evaluate and purchase or dispose of a product or service. Performing this study proves to be highly rewarding for a marketer to achieve a greater insight towards what influences consumers' purchasing decisions. Data mining is the process of scrutinizing and analyzing data in order to derive meaningful inferences. The investigation of consumer purchase patterns, precisely is carried out by the Apriori algorithm and it does so by generating frequent item sets from a transaction dataset accumulated over a significant time period. Further, it produces association rules using the frequent item sets in order to define a relationship among item sets and this is known as Association Mining. The objective of this paper is to discuss an experiment that makes use of the advantages of file systems over databases to store the transaction data and the Apriori algorithm is then implemented. The intersection of file structure concepts with Apriori is observed through experimental results.

II. LITERATURE SURVEY

A. *Hashing*

Interaction with large volumes of data has exorbitantly burgeoned in the last decade or so. Dealing with this interaction poses a rudimentary challenge such as fast access and retrieval, minimal storage space and quick delivery. File structure schemes are broadly applied to two distinct areas – data structures for central memory and file systems for secondary storage devices. However, this distinction has merged to a good extent to assume an access method applicable to both. Access methods have a simple purpose i.e., to ensure fast retrieval and updating of data and they have undergone improvisations in order to suit the needs of growing databases. Access methods are broadly classified into three categories. 'n' here represents the number of elements in the database. Sequential schemes with O(n) retrieval time,

tree-structured schemes with O (log n) retrieval time, and hashing schemes with O (1) retrieval time. [R. J. Enbody and H. C. Du. "Dynamic Hashing schemes." ACM Computing Surveys, Vol. 20, No. 2, June 1988]. Owing to the minimal retrieval time of hashing schemes, it was considered to be the most adaptable access method, thereby introducing, traditional hashing methods such as static hashing which then evolved into dynamic hashing to ensure optimal performance in a dynamic environment and eliminate space overflow. [Lianhua Chi and Xingquan Zhu. 2017. "Hashing techniques: A survey and taxonomy". Hashing has many applications, such as text classification, image retrieval, multimedia search, and data verification. A hashing function is a function that can be used to map an arbitrary size of data to a fixed interval [0, p]. The key used for hashing is passed as an argument to the hash function which returns a logical address for the record which is in turn mapped onto a physical address. [Ronald, Fagin, Jurg, Nievergelt, Nicholas Pippenger, H. Ryamong Strong. "Extendible Hashing – A Fast access method for dynamic files." ACM Transactions on Database Systems, Vol. 4, No. 3, September 1979]. The process described above is known as Static hashing. However, static hashing imposes a major limitation: when two keys map to the same address, known as collision. In order to combat this limitation, collision resolution techniques are implemented. Collision resolution techniques can be broadly classified into:

- Open Addressing
  - Linear probing
  - Quadratic probing
  - Double hashing
- Chaining

[Lianhua Chi and Xingquan Zhu. 2017. "Hashing techniques: A survey and taxonomy." ACM Comput. Surv. 50, 1, Article 11 (April 2017)]. Dynamic hashing is dynamic file access and an ability to accommodate dynamic files whilst maintaining the constant file access time. It is one of the most outstanding access methods used for modern databases with large volumes of data. The comparison of various collision resolution techniques is shown in the graph below as a measure of the average number of probes for successful and unsuccessful search.

B. *Apriori Algorithm*

As the size of data being generated burgeoned, the need to organize and draw inferences from the data grew as well. Database systems bolster accommodating large volumes of data but not the process of extracting useful information from the data in order to benefit a business, organization, and the general welfare of the society. Data mining was thus introduced which achieves the said goal. Data mining thrives in the intersection of statistics, machine learning, database systems and information retrieval. [Mohammed Al-Maolegi, Bassam Arkok. "An Improved Apriori Algorithm for Associating Rules." International Journal on Natural Language Computing (IJNLC) Vol. 3, No.1, February 2014.] Apriori is Data mining's most typical style of data mining technique that studies data and establishes an association or relationship among the dataset. It involves two important steps: Generating Frequent itemsets and Generating Association rules. Association rule problems were first introduced by Agarwal and others in 1993 and were further researched and studied by others. [Jiao Yabing. "Research of an Improved Apriori Algorithm in Data Mining Association Rules." International Journal of Computer and Communication Engineering, Vol. 2, No. 1, January 2013]. In order to improve its efficiency, the original algorithm was exposed to a series of optimizations such as changing storing framework, random sampling, adding reference points, etc. The most significant limitation of Apriori is exceeding computational time by storing itemsets with much less frequency and relevance, (support and confidence) and repeated scanning of the database for generating various candidates and eventually frequent itemsets. This paper focuses on the latter by using file structures to minimize the time taken to perform read operation. Apriori algorithm can be improved using various techniques such as, Hash-based itemset counting which refers to declaring an itemset as infrequent if its corresponding hashing bucket count is below a certain threshold value; Transaction reduction which is simply discarding a transaction without any frequent itemsets as its subsets will also be infrequent; Partitioning is done by using the principle that any itemset that is potentially frequent in a database is also frequent in a partition of the database; Sampling, mines on a subset of the dataset by using a lower threshold value and Dynamic itemset counting is defined as the addition of more itemsets only when the current ones are estimated to be frequent.

III. RESEARCH METHODOLOGY

A. *Data Mining*

Data Mining is a well-defined, diverse set of techniques that creates an assemblage of the conventional data analysis methods and the sophisticated, more efficient algorithms. These techniques are deployed to scour large databases in order to find novel and useful patterns that might otherwise be unknown.

B. *Association Rules*

Association analysis is one of the methodologies in Data mining used to uncover relationships among datasets. These relationships are represented in the form of association rules or sets of frequent item sets. An association rule is written in the form of an implication expression such as, X -> Y, where X intersection of Y gives a null set. The rule is generated based on two indigenous values known as support and confidence values. Support offers a fair idea of how often a rule is applicable to a given data set and confidence estimates how frequently itemset Y appears in a transaction that contains itemset X.
A rule with low support value doesn't contribute to a pattern and is thus not beneficial from a business perspective because it indicates that the product or service was seldom consumed. Therefore, support is used to eliminate uninteresting rules. Confidence measures how relevant the rule is to the inference made and provides an estimate of the conditional probability of Y given X. Minimum Support value is a threshold specified to control the degree of itemsets being declared as frequent.

C. *Apriori Algorithm*

The common strategy adopted by association mining is to divide the problem into two subtasks:

1. Frequent item set generation; generates all the itemsets that qualify to a minimum support threshold and are called frequent itemsets.
2. Rule generation; extracts all the high-confidence rules from the frequent itemsets found in the previous step.

The following steps are involved in the process of Association Analysis:

1. The algorithm scans the database and calculates the support value for each item and compares this value with the minimum support threshold value. If the threshold is met or exceeded by any item, it is added into the frequent-1(set containing one item) itemset.

2. This is followed by iteratively generating frequent-k itemsets using the frequent-(k-1) itemsets derived in the previous step and this process is known as Candidate generation. Of all the Candidate itemsets, some of them are eliminated using the support-based pruning strategy known as Candidate Pruning.
3. The candidate formation process which involves combining and pruning is carried out until the itemset is null.
4. Apriori uses a novel approach known as level-wise approach to establish association rules. Each level corresponds to the number of items that belong to the rule consequent. Initially, all high-confidence rules that have one item are extracted then these rules are used to generate new rules. For example, if {XYZ} -> {Y} and {XYW} -> {Z} are high-confidence rules, then the candidate rule {XW} -> {YZ} is generated by merging the consequent of both rules.

IV. IMPLEMENTATION

A. *Objective*

To simulate an application that lies in the intersection of file structures and Apriori in order to gain a deeper understanding of both areas of study.

B. *File Structures*

File systems versus databases has always been an important discussion for data storage. Depending on the objective of the application and the operations to be performed on the data, an appropriate structure is embodied. In this case, file system is preferred over a database for the following reasons:

1. For large files, performance of file systems is better than that of databases where the performance is slowed down because a simple query to retrieve the filename will also load the file data if Select* query is used. However, retrieval and access of data from files is much easier and light weight. The transaction file used in our case consists of transactions over a period of 4-6 months, thus, storing it in a file would be a better option.

2. Read and write operations on files are computationally less expensive than on databases. The transaction file used is a static file which doesn't change throughout the implementation of Apriori and undergoes simple read operation for iterating over the transactions, thus, a file would be a better option.

C. *Input – Output Structure*

There are two types of input taken from the user: a command line input for the inventory list of the business and a transaction file as a filename passed onto the Apriori function.

The inventory file is written in the form of:
[item-id]\t[item-name]
[item-id]\t[item-name]
[item-id]\t[item-name]
[item-id]\t[item-name]
and so on, where item-id represents the unique identification number for the item, and item-name represents the name of the product or service and each row corresponds to one item.

The transaction file is written in the form of:
[item-id]\t[item-id]\t[item-id]
[item-id]\t[item-id]\t[item-id]
[item-id]\t[item-id]\t[item-id]
[item-id]\t[item-id]\t[item-id]
and so on, where item-id represents the unique identification number for the item, each row represents one transaction and there is no duplication of values in one row.

The final result is written in the form of:
[item-set-name(s)]\t[associative-item-set name(s)]
[item-set-name(s)]\t[associative-item-set name(s)]
[item-set-name(s)]\t[associative-item-set name(s)]
[item-set-name(s)]\t[associative-item-set name(s)]

and so on, where item-set-name(s) represents the primary item-set of the association rule (X in X -> Y) and associative-item-set-name(s) represents the inferred itemsets from the frequent itemsets (Y in X -> Y).

V. RESULTS AND DISCUSSIONS

Consider a coffee shop business. (Table 1) The Inventory file represents a sample of the inventory list which is given as a command line input to the application. (Table 2) The Transaction data represents a sample of the transaction file over a period of 23 days. (Table 3) The Apriori result file represents the intermediate result returned by the Apriori function in the form of association rules with their corresponding support and confidence values. In the first stage of Apriori analysis, the system scans the transaction file and calculates the support for each itemset for a *minsup* value of 15% and generates candidate(k) itemset. Further it generates candidate(k+1) itemsets based on candidate(k) itemsets and evaluates for frequent itemsets. In the last stage, it derives association rules as shown in Table 3.
The dynamic hashing algorithm with chaining uses the itemset and associative item sets returned by Apriori and effectively returns the names of the same. Hashing is implemented using a HashMap for static hashing and using a linked list for hashing with chaining. Data is stored in data buckets whose address is generated by the hash function.

Inventory file after Static Hashing:
Capacity/ number of data buckets is fixed and doesn't change dynamically.
Consider the capacity fixed at 10.
Hash function is defined as – h(x) = key % capacity where key = item identification number.

After hashing, the inventory file is shown in Figure 1. After item-id 10, the hash function gives recurring values, thereby, leading to collision.

Inventory File after Dynamic Hashing with Chaining:
Capacity/ number of data buckets changes dynamically using chaining. Consider the capacity fixed at 10.
Let the hash function be the same, h(x) = key % capacity where key = item identification number. After hashing with chaining, the inventory file is shown in Figure 2.

The final output of the program is depicted in (Table 4), it represents the item sets name and associative itemsets name present in the association rules. From the table, we understand that consumers who buy Black Coffee also buy Green Tea and this rule has a fixed weightage measured by how frequently it occurs (support =20.83) in the database and how relevant it is (confidence = 55.86). Similarly, there is another rule which says that consumers who buy Black Coffee also buy Blueberry Cheesecake with similar support and confidence values indicating both are strong relationships.

10
9
8
7
6
5
4
3
2
1

Figure 1. Inventory File after Static Hashing

9 → 19
8 →18
7 → 17
6 → 16
5 → 15
4 → 14
3 → 13
2 → 12
1 → 11
10 → 20

Figure 2. Inventory File after Dynamic Hashing with haining.

TABLE 1. INVENTORY FILE

Item identification number	Item name
1	Black Coffee
2	Green tea
3	Cold Coffee
4	Pineapple pastry
5	Chocolate pastry
6	Apple pie
7	Muffins
8	Pancakes
9	Bagels
10	Vanilla cupcakes
11	Strawberry shake
12	Chocolate shake
13	Blueberry cheesecake
14	Scrambled eggs
15	Cheese toast
16	Grilled and Cheese Sandwich
17	Sausage and beans
18	Croissants
19	Waffles(plain)
20	Waffles with whipped cream

TABLE 2. TRANSACTION DATA

Item identification number
9
18 2 4 5 1
1 11 15 2 7 16 4 13
2 1 16
15 7 6 11 18 9 12 19 14
11 2 13 4
11 13
7 4 2 17 19 3 8 16 1
18 16 15 10 2 8 6 0 4 5
9
10
1 13 8 9 3 16 4
16 0 6 11 8
19 6 3 8 16 17

18	2	9	1	13	15
	19	4	10		
6	13	1	5	4	12
	10	9			
13	17	12	6	10	1
	16	15			
15	14	11	12	10	1
	4	6			
7					
7	17				
7	13	3	8	16	5
	9	18			
8	2				
6	10	13	8	0	11
	2				

TABLE 3. APRIORI RESULT

X	Y	Support	Confidence
{1}	{2}	20.83	55.56
{2}	{1}	20.83	55.56
{1}	{4}	29.17	77.78
{4}	{1}	29.17	77.78
{1}	{13}	20.83	55.56
{13}	{1}	20.83	55.56
{1}	{16}	20.83	55.56
{16}	{1}	20.83	55.56
{2}	{4}	25.00	66.67
{4}	{2}	25.00	66.67
{4}	{13}	20.83	55.56
{13}	{4}	20.83	55.56
{6}	{10}	20.83	62.50
{10}	{6}	20.83	71.43
{8}	{16}	25.00	75.00
{16}	{8}	25.00	66.67

TABLE 4. FINAL RESULT

Black Coffee	Green Tea
Green Tea	Black Coffee
Black Coffee	Pineapple pastry
Pineapple pastry	Black Coffee
Black Coffee	Blueberry Cheesecake
Blueberry Cheesecake	Black Coffee
Black Coffee	Grilled and Cheese Sandwich
Grilled and Cheese Sandwich	Black Coffee
Green Tea	Pineapple pastry
Pineapple pastry	Green Tea
Pineapple pastry	Blueberry Cheesecake
Blueberry Cheesecake	Pineapple pastry
Apple pie	Vanilla Cake
Vanilla Cake	Apple pie
Pancakes	Grilled and Cheese Sandwich
Grilled and Cheese Sandwich	Pancakes

VI. CONCLUSION

The results and implementation suggest the following inferences:

- Black Coffee is clearly the most sold item among the given transactions and can be combined with other items to increase demand and attract more customers.
- Apriori is one of the most efficient and compatible algorithms designed to achieve this. It is highly flexible and offers a large range of improvisations that make it suitable for the given application at hand.
- With a support value of 15%, 16 association rules were generated.
- Dynamic hashing with chaining makes access and retrieval of data computationally less expensive in comparison with static hashing. Data is stored in a more structured manner allowing for fewer numbers of buckets.
- Studying consumer purchase patterns is highly beneficial regardless of the size of the business. It is a gateway to improving consumer experience and growing one's business.

ACKNOWLEDGEMENT

We would like to thank JSS Academy of Technical Education for providing us this opportunity.

REFERENCES

[1] R. Agrawal, T. Imieliński, and A. Swami, "Mining association rules between sets of items in large databases," in ACM SIGMOD Record, vol. 22, pp. 207–216, 1993

[2] W. Sun, M. Pan, and Y. Qiang, “Improved association rule mining method based on t statistical,” Application Research of Computers. vol. 28, no. 6, pp. 2073-2076, Jun, 2011.

[3] Suresh, Jogi, and T. Ramanjaneyulu. “Mining Frequent Itemsets Using Apriori Algorithm.” International Journal of Computer Trends and Technology (IJCTT), vol. 4, no. April, 2013, pp. 760–64.

[4] Dutt, Shalini, et al. “An Improved Apriori Algorithm Based on Matrix Data Structure.” Global Journal of Computer Science and Technology: C Software & Data Engineering, vol. 14, no. 5, 2014, pp. 1–5

[5] Wanjun Yu; Xiaochun Wang; Erkang Wang; Bowen Chen;, “The research of improved apriori algorithm for mining association rules,” Communication Technology, 2008. ICCT 2008 11th IEEE International Conference on, vol., no.,pp.513-516, 10-12 Nov. 2008

[6] Agrawal, Rakesh, “Fast Algorithms for Mining Association Rules in Large Databases”, Proceedings of the ACM SIGMOD International Conference Management of Data,Washington, 1993, pp.207-216.

[7] Jiawei Han, Jian Pei, Yiwen Yin. “Mining Frequent Patterns without Candidate Generation. “ SIGMOD Conference 2000: 1-12

[8] A. Savasere, E. Omiecinski, and S. Navathe, “An Efficient Algorithm for Mining Association Rules in Large Databases”, In VLDB’95, pp.432-443, Zurich, Switzerland

[9] F L J. Enbody. “Dynamic Hashing Schemes”. Department of Computer Science, Michigan State University, East Lansing, Michigan 48823 H. CDU Department of Computer Science, Minnesota, Minneapolis, Minnesota 55455

[10] RAMAKRISHNA, M. V., AND LARSON, P.-A. 1988. “File organization using composite perfect hashing.” ACM Trans. Database Syst. To be published.

[11] LARSON, P.-A. 1980. “Linear hashing with partial expansions.” In Proceedings of the 6th Conference on Very Large Databases (Montreal). Very Large Database Foundation, Saratoga, Calif.

[12] R. Agrawal, T. Imieliński, and A. Swami, “Mining association rules between sets of items in large databases,” in ACM SIGMOD Record, vol. 22, pp. 207–216, 1993

[13] Pang-Ning Tan, Michael Steinbach and Vipin Kumar, “Introduction to Data Mining,” Dorling Kindersely, 2009

[14] M. Scholl, “New File organisations based on Dynamic Hashing”, ACM TODS, 6 (3), 1981.