**Optimization of Apriori Algorithm for Association rule mining based predictive systems by applying parallelism technique using High performance parallel GPU and CUDA Programming**.

Muhammad Zubair

Paf Kiet City Campus

Mzubair\_700@gmail.com

M.Azam Khan

Paf Kiet City Campus

Azamkhan15@yahoo.com

Abstract

The advancement in the Information technology for storing and retrieval of information in large transaction databases has led to an increase in storage of data and extract useful information from it by using Association rule mining techniques of data mining which enables to explore knowledge in the form of a set of repetitive items or associative rules[1]. But with the increase in the amount of data, the speed of extraction of such pattern decreased. The Apriori algorithms was the first successful attempt to solve such type of problem[2], but the performance matrix of the sequential Apriori algorithm facing performance challenges due to memory for Analysis for such type of predictive systems. This paper proposed a new parallel GPU-Apriori algorithm (GPUAA) that is optimized to meet the limitation of serial version of Apriori with help of (Graphics Processing Unit) GPU high throughput performance and the Compute Unified Device Architecture (CUDA) programming model. This paper also discusses the experimental results that we have obtained from implementing the GPUAA. In the end we present possible future directions to improve the implementation of our GPUAA.

Keywords— Apriori algorithms; Association rule mining; GPU Computing; CUDA; Parallel Algorithms

I. INTRODUCTION

Rapid advancement in the digital technology in storage and information processing has led to an increase in storage of data and extract useful information from it. This process is known as the Knowledge Discovery in Databases or KDD, which is a repetitive process and involve data mining technique to determine that can assist to create precise future decisions. The main aim of Data mining approach is to extracts useful information from a huge set of data. With the increase of the size of the data, the sequential search algorithms for conventional association rules cannot deliver the required results in reasonable times [3]. One of the most famous data mining algorithms is association rules mining which is widely used in the field business, finance and health. Apriori algorithm has a special place as compare to other algorithm as it is used for frequent item sets generation to generate all association rules which was published by Agrawal in 1994[2]. Usually it is used to find the relevant pattern in huge transaction database also known as frequent item set mining (FIM). FIM use the measurement technique of frequent item set is minimal support which represents the frequency of the occurrence of this item set in the transactional database, as Apriori based on serial execution manner of scanning the database repeatedly for a large number of candidate items. So it is becomes inefficient with the increase in data set. Recently techniques uses high throughput computing device like GPU to speed up this algorithm. This paper proposed a new parallel GPU-Apriori algorithm that is optimized to meet the limitation of serial version of Apriori with help of GPU and the Compute Unified Device Architecture (CUDA) programming model. In the implementation of our algorithm we use a strings of bit data structure which use CUDA parallel programming model to generate candidates for a better performance than serial version implementations. We have tried to show that our GPUAA is more feasible and optimal and can be used as a guide line for those application developer interested in to developed frequent item set mining applications.

II.Background

Association rule data mining has become one of the core data mining techniques and has becomes an interest area among the data mining researchers. This technique works on variable length data, and produces proper and useful results. Data Mining Association Rules is aimed to provide from a set of transactions of items, to find correlations among the items that occurs in sales transaction. In business terminology this whole process is known as, market Basket analysis. The purpose of GPUAA is to helps managers to know about the buying behavior of their customers and ultimately help them to determine the marking planning approaches for their products and also help them to promote the sale of this type of related items.

Section wise division of this paper is as follows. Section 2 presents a previous survey which provides to collect the evidence in support to find of the previous approaches use by research to solve the same problem. What are the current limitation and problem in sequential based A algorithm. Section 3 discusses the design of our purposed algorithm. It also introduces the design description of our algorithm. This includes descriptions on how our algorithm solve this problem, it will also show how GPUAA used to enhance predictive analysis. In Section 4 will conclude of work and Section 5 summarizes this paper and give description about our future work.

III. Literature Review

Systematic literature review provides us to identify and collect the evidence from different previous research papers in support of our working for GPUAA. This also provides to collect the evidence in support to find of the importance of our GPUAA technique as compare to other formal techniques and why GPUAA is better than others. What are the limitation and problem in other technique?

**IV.Related Work**

Our study supports that the all-important and relevant feature identified for limitation of serial over parallel computation approach can be accomplished by using GPUAA approach. In addition with the advancement of GP-GPU programming some researchers also agrees that GPUAA approaches have more capabilities and speed for satisfying the information requirements for analysis of large Transaction based Predictive System and it’s also enables knowledge extraction in timely manner as compare to its serial contra part.

A significant amount of research has already been carried out in evaluation different approaches to increase the performance matrix of the sequential apriori algorithm for analysis of predictive systems. For example, according to [3] conclude that the Apriori algorithm is one of the first and most popular algorithm used to explore the collection of repetitive items, with support and confidence interval, to predict association between different items, but is also faces some problems. The Apriori algorithm has a higher computational complexity due to surface-levelness than other algorithms exploring the repetitive patterns of the transaction database. To overcome these, improvements have been made using various techniques on it. One of these is the parallelization techniques. Parallel algorithms based on FP-growth and Éclat, due to the nature of the pattern search, increase only the number of processor cores in order to increase the number of exploring processes of repetitive items. A lot of work has been done in the past with the aim to support to increase the performance matrix of the sequential apriori algorithm for analysis of predictive systems. These approaches have cover various aspects of to increase the performance matrix of the sequential apriori algorithm for analysis of predictive systems, but they have some limitation for example they provide weak support for to increase the performance matrix in term of knowledge extraction. One of the reason found by researcher that they are not built on a basic foundation of Parallelism.

Apriori Algorithm uses frequent item set mining and association rule learning by creating Item permutations and then computes confidence given a minimum support. On the other hand GPUAA Algorithm parallel approach instead of permutations create combinations, the combination set is loaded in GPUs and compute all possible sets. Normally we use Index table instead of counting over and over again.

Our Literature review reveals that Apriori, Eclat [4], and FP-growth [5] are found to me popular algorithms for frequent item set mining. Apriori uses breadth first search order to generate candidate subsets and its work on works on horizontal layout based database and. Eclat uses vertical database layout algorithm and represent data in bit matrix format with depth first search and prefix tree and found to be memory efficient with backtracking techniques. FP-growth also uses tree based algorithm approach with the least memory usage. A performance comparison analyses is performed in [6] which show that FP-growth outperforms Apriori and Eclat in serial algorithms, but due to data structure make it less suitable for parallelization as compare to Apriori. Most of studied literature review reveals that parallelizes FIM algorithm based on Apriori or ECLAT is a common approach. [7]. As serial algorithms cannot take advantage of the high-performance parallel computing platform, [8] proposed the first parallel version of some algorithm. l. [9] demonstrated an improved parallel version of Apriori that applies data parallelism in multi-core processor environment. [10] developed GPUMiner-Apriori that adapts Apriori implementation on the GPU. Also an Apriori Like algorithm also introduced by [11] known as GPU-FPM which realized 15x speedup ratio.

**V. Methodology**

In this section, we will describe our designing detail of GPUAA with a scenario to provide support for our GPUAA in real life application. The purpose of doing so is that how we can take the advantage of GPUAA, in order to minimize the average time to extract relevant information.

**An Example real life Scenario for GPUAA**

The majority of the business organizations have huge amount of data about transaction history of their customers in their databases. An association rule is A⇒B, A and B are sets of items and the rule represents those customers who purchase A also purchase B with probability %c where c is known as the confidence. An example of such type of rule may be like: “70 % of customer who purchase butter also purchase breed". Such rules helps to determine the future strategy that with which item the Butter are sold with. This is an ideal example for illustrating association rule mining. It helps managers to know about the buying behavior of their customers and ultimately help them to determine the marketing planning approaches for their products and also help them to promote their sales level of this type of related items. But this type of association rule mining also required some level of minimum support and confidence constraint to discover all the rules offered in the database.

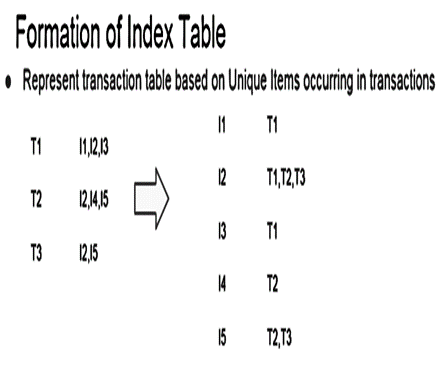
So our research will also provide a way to present market basket analysis for any type of product where mined data associations help business management to sell frequently purchased combinations of products to their customers which in turn provide a way to guide the decision makers to make use of these extracted knowledge to predict future purchasing occurrences trends of their prospective customer to maximize their profits.

**Design Consideration**

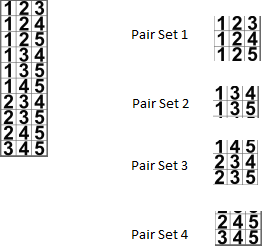
Now we will discuss the design and working of GPUAA. The flow of our proposed GPUAA is illustrated in figure 1 and 2 respectively.

**Steps of GPUAA:**

1. Transpose the transaction/item table
2. Convert the table into TF-Idf representation

**Figure 1: Transpose phase**

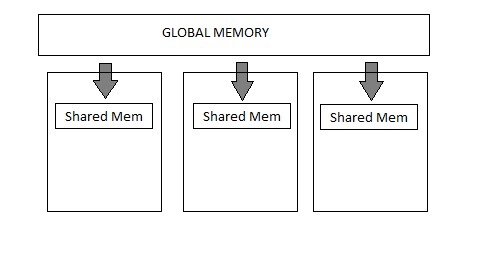
1. Given the item pairs to calculate support for , accumulate pairs which seek same rows into same blocks

**Figure 2: Graphical illustration of pair’s accumulation for blocks**

1. Then assign these blocks on GPU

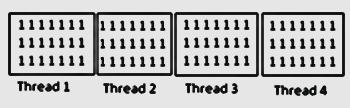
Working on GPU:

1. First all the rows which are required for the threads are fetched into shared memory
2. Each thread computes single pair

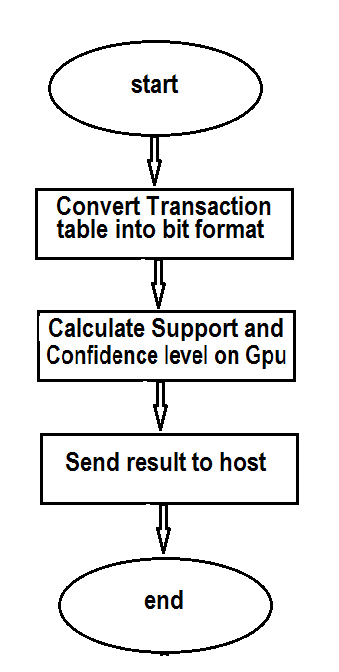


**Figure 3: GPU processing phase**

1. For all items in pair the thread fetches the data from shared memory and then performs AND operation on all bits of the rows
2. Finally having the AND result all bits which are 1 are summed and SUPPORT is compute for the pair set



**Figure 4: Thread computation on SMs**



**Figure 5: Flow chart of GPUAA**

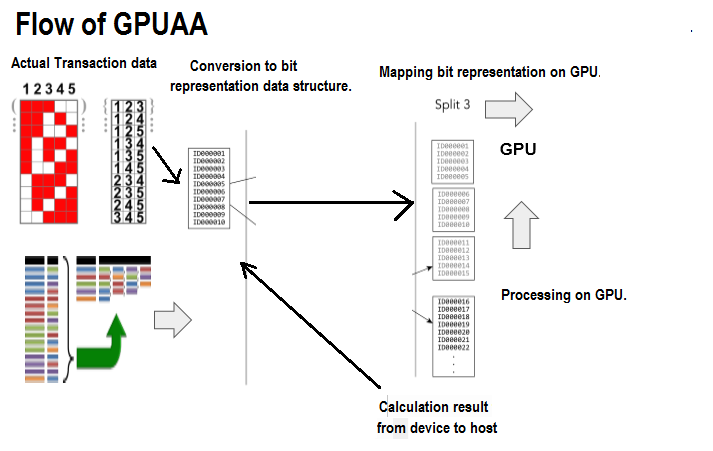
Inputs for our kernel function:

1. Binary array representation of an item being present in particular transaction
2. Sequence pattern array
3. Length of pattern
4. No of pattern

**Kernel processing**

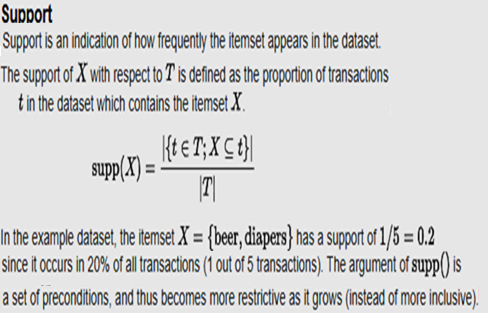
Each thread computes and operation on all the rows provided in sequence pattern and finally adds all the bits resulting in support for particular sequence and send the result an array of integers containing support count for each pattern.

This above figure illustrate that how our algorithm use parallelism, first instead of permutation as in Sequential version we used combination, then we inverse our transaction tables row and column and store that it as a binary representation in array. When any query comes to find the confidence level for example I2, I3, I5 for the support =2, now our version not only prevents to access memory to compute permutation again and again like in case of serial version but also shift all data through data mapping on GPU at once to perform combination and get the result which return level of support in 1 time. This will speed up the process as we have not to search to the count the item as in serial version, instead we get the same thing by ANDING rows of the 3 items. To measure the relation of association rules by using their support(s) and confidence(c) properties. The support property here means that how many times a rule can be utilized in a dataset and confidence property measures occurrence of X with respect to the transactions that also have Y. Formulas in figure 3 and figure 4 are used to calculate support and confidence of item Y respectively:

**Figure 6: Overall flow of GPUAA**

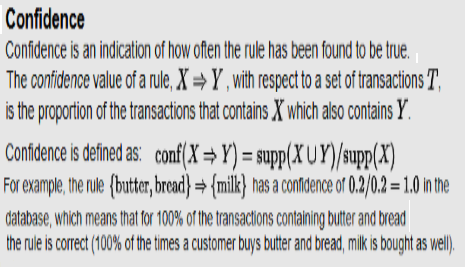
**Support**

Here support means how much times a pair will come to fulfill the requirement to calculate confidence, for example, if a pair appears less than the minimum support required that we cannot able to calculate its confidence because it already failed.

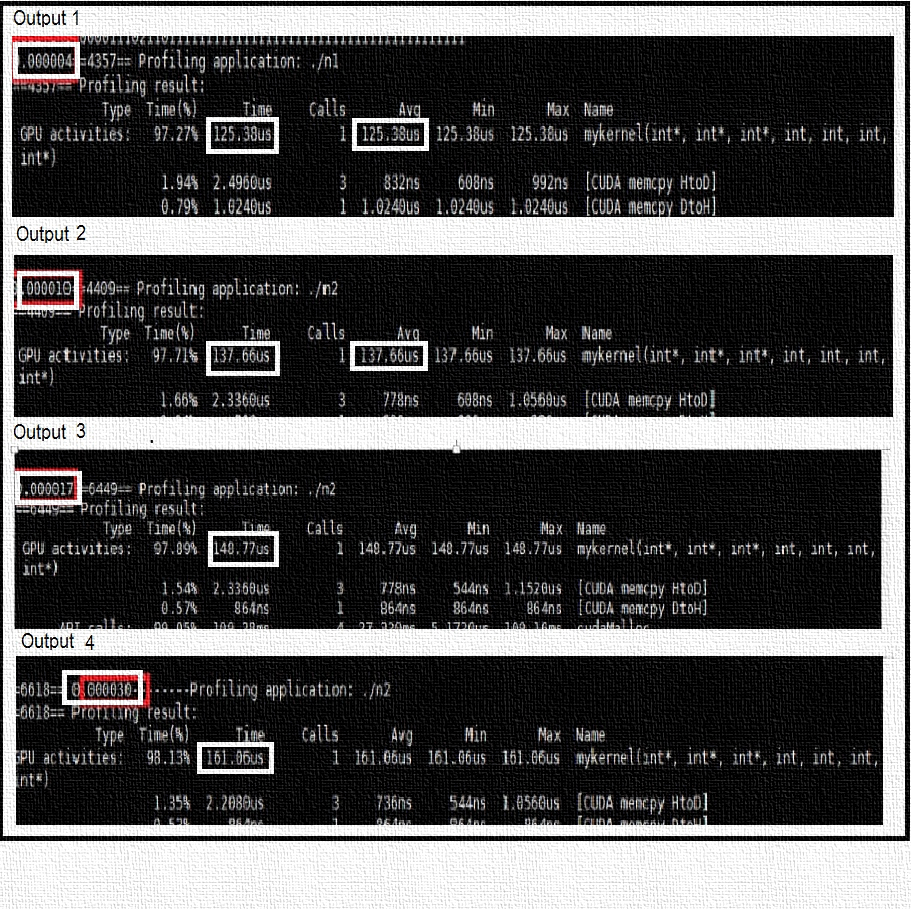
**Figure 7: Mathematical model of support**

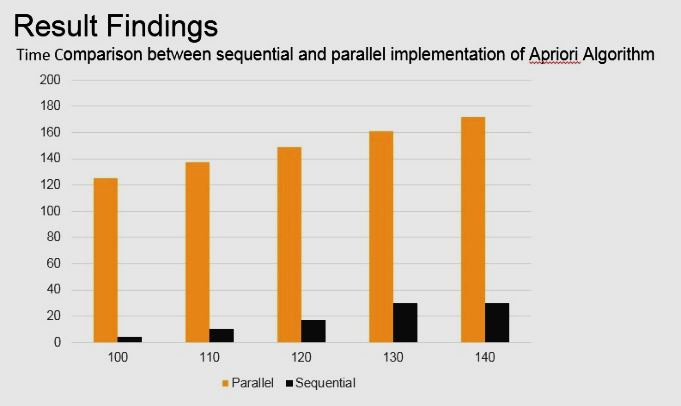
**Confidence**

**Calculating Confidence for transactions**

**Figure 8: Mathematical model of Confidence**

To calculate the confidence we use the formula, given the figure 1, suppose that if query consist of for bread and butter and we have calculate confidence on the bases of milk so we use it as follow

**Number of transaction contains for bread and butter occurs/ Number of transaction contains in which only milk occurs**

****Result / Findings

We run our CUDA c implementation program of our algorithm to find the difference between serial version and GPUAA versions with different randomly generated data sets. The output of these experiments are illustrated in the following diagram:

The above graph show that still sequential versions performance is better than parallel version but hopefully we are confident by changing some logic in our algorithm we will able to remove this problem in our future work, because the reason for parallel implementation performing worse than sequential is due to each thread access global memory for same data. That was the actual flaw in our solution. Our future work will be focus on to divide the threads to calculate count and confidence level into different blocks so that each thread use shared memory to speed up the computation process that will result to achieve our desired goals.

**Figure 9: Time comparison between serial and parallel GPUAA**

**VI. CONCLUSION**

Our study in this paper reveals that parallel technique based algorithm for associated rule based data mining is most feasible and relevant for supporting for data parallelism application. The design consideration of our parallel algorithm take care of an data parallelism feature by implementation of a parallel computing approach to speed up the process. Although much work is done to build such type of model in the past, but as discussed earlier that they have some weakness. Our GPUAA is a try to remove all type of these previous weakness.

There is, however, more research is required in this field to provide clear path for better management for data parallelism. Some of the future works which can be done in this field that still some improvement is required because our model did not fulfill its goal as shown in the graph. In addition there is a strong need for focus on shared memory and increase the number of block size with the help of different case studies. Future the implementation of GPUAA will we enable to measure the quality of its performance by removing the flaws find in our experimental result. But hope fully this model will provide a guideline for application developer to develop predictive system application.

**VII. ACKNOWLEDGMENT**

The author/s would like to thank Dr. Ayaz to give the valuable knowledge of Gp-Gpu computing field in this course, to provide guidelines and support for writing this research work and also for his technical support during whole semester.

**VIII .References**

[1] Implementation and Analysis of Enhanced Apriori Using ,MapReduce Mercy Nyasha Mlambo ,Naison Gasela and Michael Bukohwo Esiefarienrhe, 2018 Tenth International Conference on Advanced Computational Intelligence (ICACI)

[2] S. Chakrabarti, M. Ester, U. Fayyad, J. Gehrke, J. Han, S. Morishita, G. Piatetsky-Shapiro, and W. Wang, Data mining curriculum: A proposal (Version 1.0), 2006.

[3] A Fast Algorithm Based on Apriori Algorithms to Explore the Set of Repetitive Items of Large Transaction Data Javad Ghofrani Hannover Hanover, Amir Panah Hadaf Institute of Higher Ed

[4] M. J. Zaki and K. Gouda. Fast Vertical Mining Using

Diffsets. in Proc. SIGKDD. 2003. p. 326-335

[5] J. Han, H. Pei, and Y. Yin. Mining Frequent Patterns without

Candidate Generation. in SIGMOD. 2000. p. 1-12

[6] F. Bodon, A Trie-based APRIORI Implementation for Mining

Frequent Item Sequences, in OSDM. 2005. p. 56-65.

[7] C. Borgelt. Efficient Implementations of Apriori and Eclat. in

Proc. FIMI. 2003.

[8] M. Zaki and S. Parthasarathy, New Algorithms for Fast

Discovery of Association Rules, in KDD. 1997. p. 283-296.

[9] NVidia. Data Parallel Algorithm in CUDA SDK Available

from: http://developer.download.nvidia.com.

[10] Frequent Itemset Mining Dataset Repository Available from:

http://fimi.ua.ac.be/data.

[11] J. Zhou, K.-M. Yu, and B.-C. Wu, “Parallel frequent patterns mining algorithm on gpu,” pp. 435–440, 2010