Comparative Analysis of Genetic Based Approach and Apriori Algorithm for Mining Maximal Frequent Item Sets

Mir Md. Jahangir Kabir
School of Engineering and ICT
University of Tasmania
Launceston, Australia
mmjkabir@utas.edu.au

Shuxiang Xu
School of Engineering and ICT
University of Tasmania
Launceston, Australia
Shuxiang.Xu@utas.edu.au

Byeong Ho Kang
School of Engineering and ICT
University of Tasmania
Hobart, Australia
Byeong.Kang@utas.edu.au

Zongyuan Zhao
School of Engineering and ICT
University of Tasmania
Launceston, Australia
Zongyuan.Zhao@utas.edu.au

***Abstract*—In the data mining research area, discovering frequent item sets is an important issue and key factor for mining association rules. For large datasets, a huge amount of frequent patterns are generated for a low support value, which is a major challenge in frequent pattern mining tasks. A Maximal frequent pattern mining task helps to resolve this problem since a maximal frequent pattern contains information about a large number of small frequent sub patterns. For this study we have developed a genetic based approach to find maximal frequent patterns using a user defined threshold value as a constraint.**

To optimize the search problems, a genetic algorithm is one of the best choices which mimics the natural selection procedure and considers global search mechanism which is good for searching solution especially when the search space is large. The use of evolutionary algorithm is also effective for undetermined solutions. Therefore, this approach uses a genetic algorithm to find maximal frequent item sets from different sorts of data sets. A low support value generates some large patterns which contain the information about huge amount of small frequent sub patterns that could be useful for mining association rules. We have applied this genetic based approach for different real data sets as well as synthetic data sets. The experimental results show that our proposed approach evaluates less nodes than the number of candidate item sets considered by Apriori algorithm, especially when the support value is set low.

Keywords— association rules; data mining; maximal frequent item sets; genetic algorithm; lexicographic tree.

I. INTRODUCTION

Mining frequent item sets is one of the most popular and well known methods in the data mining research area. Nowadays it is used in different tasks like finding interesting relationships between variables, although it was initially developed for business transaction problems like market basket analysis. The main aim of the frequent pattern mining task is to search interesting relationships among different item sets. It can be used to solve different problems such as revealing association rules, concluding sequential patterns, correlations, and other significant data mining tasks. A transactional database can be defined as a data set of transactions, each transaction consists of a set of items, termed as an item set. An item set is frequent if it appears abundantly in a given database. In other words, certain ratios of all the transactions exist with respect to a user define threshold value. Mining frequent item sets is a time consuming task especially for dense databases where long patterns of item sets are emanated, which have been prolonged to mining maximal frequent item sets. An item set is called maximal if it contains the longest frequent patterns under user define threshold value.

Let $D=\{t_1,t_2,t_3, \ldots, t_k,\ldots,t_n\}$ be the database, where $t_1,t_2,t_3, \ldots, t_k,\ldots,t_n$ are an n number of transactions in the database. Each transaction t_n is a set of items $I=\{i_1,i_2,\ldots.,i_k,\ldots,i_n\}$, where item number 1 is i_1, and i_n is item number n and so on. Transaction t_n is represented as a binary vector. If $t_n[k] = 1$ then it means that t_n bought the item i_k, otherwise $t_n[k] = 0$. Let X is a set of few items in I i.e. $X \subseteq I$ .The set $t_n(X) \subseteq I$ is true for all items in item set X for transaction t_n. The support value of an item is how many times the item appears in the transaction database as a subset. The support value of an item set is denoted by (1).

$$\sigma(X)=|\{t_1(X)+t_2(X)+ \ldots \ldots+t_{(n-1)}(X)+t_n(X)\}|/|D| \quad (1)$$

Here $t_n(X)$ gives the binary value. If the examined item set X appears as a subset in a transaction t_n, then $t_n(X) = 1$, otherwise $t_n(X) = 0$. An item set with 1 item is called a 1-itemset, an item set with k-items is called a k-item set. An item set is called frequent if its support value is more than or equal to a user defined threshold value, which is denoted by min_supp (minimum support) i.e. $\sigma(X) \geq$ min_supp. We denote frequent item sets by FI. If an item set X is frequent and no superset of X is frequent then we can claim that X is a maximal frequent item set and we denote the set of all maximal frequent item sets by MFI.

To generate maximal frequent item sets (MFI) from a large database is a most time consuming task in the present day. In this paper, we present an evolutionary approach to finding maximal frequent item sets from large databases by using the principles of Genetic Algorithm (GA).

Mining maximal frequent item sets using a genetic algorithm is the main approach of this research. The length of a frequent item set depends on its relationship among the item

sets. The major advantage of a GA based approach is that it performs a global search and its time complexity is less than that of other algorithms. Another advantage is that it generates frequent item sets independently of the size of the data-base.

This work differs from existing research [1] in the following aspects: 1) Unlike Apriori, this approach uses a lexicographic tree [2] as a search space and it does not need to enumerate frequent item sets level by level; 2) For comparative analysis, we applied Apriori and this approach in different real datasets as well as synthetic datasets. Finally the results are compared with the results of Apriori algorithm. 3) Unlike a Boolean based approach [3] and FP- growth algorithm [4], this approach does not need memory for loading a lexicographic tree which avoids the large consumption of memory space. This technique dramatically reduces the time for accessing a large database to calculate the support value of unnecessary individuals to find frequent item sets. Although it was invented a long time ago but still Apriori is one of the famous algorithms and it performs better than other existing algorithms like Eclat, Partition, DIC and so on when the support value is set high [5]. Hipp, Guntzer and Nakhaeizadeh showed the performance analysis of Apriori and other existing famous algorithms of present day. For this reason, we choose Apriori for comparison with our new approach. CPU time (Run time) is needed by the existing mining approaches for calculating support value of examined nodes. The efficiency of an algorithm depends on how many number of frequent or infrequent item sets it considers to get the final solution i.e. maximal frequent item sets [6]. In this paper we will examine how many number of nodes i.e. item sets are considered by GA based approach and compared the result with Apriori algorithm for different support values and data sets.

This paper is composed of the following sections: section 2 summarizes essential back-ground information including basic concepts of genetic algorithms and data mining. The methodology of traversing the search space and the proposed algorithm is described in section 3. Section 4 represents the comparative analysis of genetic based mining technique and Apriori algorithm. Finally, a short summary of our results is included in section 5.

II. BACKGROUND

Three aspects of the literature are reviewed in this section. Two of these are the fundamental component of the proposed approach, namely the genetic algorithm and frequent item sets mining, which are the techniques for pattern mining tasks. Finally, a related study of frequent item sets mining is presented. Elaborated information is given below.

A. Genetic Algorithm

The Concept of Genetic Algorithm.

Genetic algorithm considers adaptive methods which are used to solve search as well as optimization problems. This algorithm is inspired by natural selection and the "survival of the fittest" mechanisms which are clearly stated by Charles Darwin in the book name "The Origin of Species". Based on the fitness value, in a competing environment only the stronger individuals will survive. The processes in natural population which are essential for evolution are simulated by GA. Holland [7] first proposed the basic principles of genetic algorithm [8]. Thereafter, a large number of researchers worked on genetic algorithm [9]–[12]. Naturally individuals are competing with each other for their shelter, food, clothes, water and so on. Even members of the same class often compete to attract their partner. Those individuals are referred to as strong if they are successful in surviving and attracting a partner. Strong individuals will produce a large number of offspring. On the other hand poorly performing individuals are referred to as weak and have less probability to produce newer offspring. The combination of good attributes from different parents can produce "superfit" offspring. That is the fitness of this offspring is higher than the fitness of the parents. In this fashion, species becoming more and more well suited in the present environment.

Procedure of Genetic Algorithm.

Genetic algorithm plays a vital role for this study which simulates the natural behavior of biological organisms. Genetic algorithm based techniques are robust and can be used to solve a wide range of problems including those which are hard to solve by other methods. Researchers conclude that, it is not guaranteed that GA always provides optimum solutions to a problem rather it provides "acceptably good" solutions to a problem which is solved by other methods "quickly". Existing methods which are working well as a solution for a particular problem, improvement of those methods can be done by hybridizing with GA.

A Traditional Genetic Algorithm generates an initial population, and then computes the fitness value of that population. Two individuals are selected from the old generation and applying crossover, mutation operators to produce two offspring. It selects the survivors which have the best fitness value and inserts those in the new generation. If the population is converged to a solution then the algorithm is terminated. In this algorithm, fitness function provides the fitness value of an offspring which is a specification of the offspring.

B. Data Mining

Data mining is the process of finding relationships in large data sets applicable to methods of artificial intelligence, statistics and machine learning. Different activities are included by data mining. It considers data from different sources and then translates, formats and cleans these data sets for further use, like analysis, integration and validation. The goal of data mining is to extract patterns and knowledge instead of mining data itself from large amounts of data sets. By analysing large amounts of data, data mining is used for extracting unknown important patterns such as association rule mining, anomaly detection, clustering and so on. Predictive modelling, clustering techniques, summarization methods, link analysis and classification techniques are the five analytical domains which demonstrate the importance of data mining in real world applications. In business transactions, frequent pattern mining gives an idea about the popularity of buying item sets to the users. By using this information industries stock those popular products and get benefitted by it.

In human genetics research, the aim of sequence mining is to finding the changes in DNA sequences of individuals which are responsible for increasing the risk of common diseases like cancer. This study helps to develop the methods of diagnosing and preventing these diseases. Frequent pattern mining plays a vital role in mining correlations, associations and other interesting relationships among data sets. Moreover, it helps in different data mining tasks such as indexing, clustering, classification and so on. For this reason, mining frequent patterns is an important data mining task and a focused topic in data mining area.

C. Related Works

It is well known that the Apriori algorithm generates a candidate set and tests it in a breadth first manner. It discovers all the frequent item sets at level k before moving to its next level (k+1). It counts the support value of each node in level k and prunes those nodes if the support values of those nodes do not satisfy a user define support value. It generates candidate item sets at each level and scans the database so frequently that it is costly, especially when there is a long pattern [13].

The Princer-Search algorithm [14] traverses a lattice through a bidirectional method that follows both top-down and bottom-up approaches. To find maximal frequent item sets it applies pruning methods by the following two properties: all the subsets of frequent item sets are pruned and all the supersets of infrequent item sets are pruned.

Breadth first traversal (a level by level search strategy on a search space) is applied for a MaxMiner search algorithm. To prune the branches of a tree it performs a look-ahead method. MaxMiner uses a breadth first approach for limiting the number of passes over the database but look-ahead, which involves superset pruning, works better for depth first search methods [15].

DepthProject performs depth first traversal on a lexicographic tree along with variations of superset pruning. To order child nodes, it applies dynamic reordering methods. By trimming infrequent items out of each node's tail, it reduces the size of the search space. To eliminate non maximal frequent item sets DepthProject would require post pruning methods [16].

MAFIA, proposed by Burdick, Calimlim, and Gehrke [17], extends the idea of DepthProject. Similar to DepthPorject, MAFIA also uses vertical bitmap representation where the support value/count of an item set is based on AND operations among the item sets. The search strategy of MAFIA integrates a depth first method to traverse the tree to find maximal frequent item sets along with effective pruning methodology.

In [18], Gouda and Zaki proposed a novel approach called GENMAX to find maximal item sets. In this approach they used a novel technique called Progressive Focusing. This technique maintains local maximal frequent item sets (LMFI) which are used for making comparison with newly found frequent item sets (FI). GENMAX uses vertical representation of a database and stores a transaction identifier set (TIS) for each item set instead of a bit vector. Researchers of GENMAX concluded that, through experimental results this algorithm it performs better than existing algorithms on different types of databases.

Bilal Alataş and Erhan Akin [19] designed an efficient genetic algorithm as a search strategy to mine both positive and negative quantitative association rules. Association rules are deduced from frequent patterns. This method mined the association rules without generating frequent item sets. The proposed genetic algorithm does not depend on minimum support and a confidence value which is hard to define for a database. A new genetic operator named uniform operator is used in this approach which ensures genetic diversity.

In [20], a quick response data mining model based on a genetic algorithm has been de-signed. This approach gives more flexibility to the user. It only scans the database for those frequent item sets users are more interested in.

To mine quantitative association rules researchers proposed a new algorithm which is based on genetic algorithm named QUANTMINER [21], [22]. By optimizing support and confidence value, this system dynamically identify good intervals in association rules. Researchers applied this algorithm in different data sets and showed the usefulness of this algorithm as a data mining tool.

Hipp, Guntzer and Nakhaeizadeh [5] showed the performance analysis of Apriori and other existing famous algorithms of present day such as Eclat, Partition, DIC and so on. Performance analysis of different algorithms demonstrate that, Apriori algorithm performs better than other existing algorithms for high support value.

III. METHODOLOGY

The proposed method of frequent item set mining for different data sets is described in this section. The following subsections will describe Lexicographic tree, Problem definitions and the proposed method, a genetic algorithm based technique, respectively.

A. Lexicographic Tree

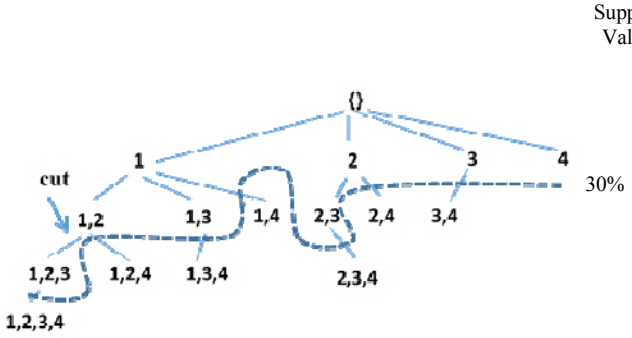
In this paper, we will consider a lexicographic tree as a search space which consists of all feasible solution [17], [23]. A Lexicographic tree maintains lexicographic ordering of items *I* in a datasets D. If item i occurs before item j in a dataset D, then it maintains lexicographic ordering, i.e., $i \leq_L j$. If two subsets S_1 and S_2, where $S_1 \subseteq S_2$ and $S_1, S_2 \in S$ then it maintains the following lexicographic order $S_1 \leq_L S_2$. There is no lexicographic ordering relationship between two subsets S_1 and S_2, if S_1 and S_2 are disjoint subsets.

Fig. 1. shows an example of a lexicographic tree which considers lexicographic ordering of four given item sets {1,2,3,4}. The root of the tree is an empty set and each k-level contains k-items. All the nodes in the tree contain a different size of item sets. In each level, k-item sets maintain lexicographic ordering with the tail nodes, containing items lexicographically larger than elements of the head node. The support value of the head node is more than that of the tail node. It can be seen that the nodes closer to the root are more frequent than those far from the root. There is a nonlinear line

(called a cut) in the tree which separates frequent item sets from infrequent ones. This cut is designed in such a way that it depends on a user defined support value. The nodes which are above the cut are frequent item sets and the elements below this cut are infrequent ones. All the nodes in lexicographic tree have a support value.

Lemma 1: If dataset D contains n items, then it enumerates 2^n-1 frequent sub item sets. Fig. 1. verifies Lemma 1, where there are 4 items, and it enumerates 2^4 - 1 = 15 nodes including the root node. Generally root node is an empty node.

Fig. 1. Lexicographic tree of four items based on a user defined support value.



Note that memory space will not be used for storing the nodes and links between the nodes of a lexicographic tree.

B. *Problem Definition*

A huge number of frequent patterns are generated from big data sets which satisfy user define threshold value especially when user assign a lower value for min supp. Because of generating enormous number of frequent item sets from large data sets is a major challenge in frequent pattern mining task. This happened because if an item set is frequent, all of its sub item sets are frequent. To solve this problem, researchers proposed closed and maximal frequent pattern mining task. For this study, we consider maximal frequent pattern mining task.

Definition (Maximal Frequent Pattern Mining).

An item set ε is a maximal frequent item set in a data set D, if ε is frequent and there exists no superset η such that ε⊆η which is frequent in data set D. That is a frequent item set is called maximal, if all of its sub sets are frequent whereas all of its supersets are infrequent.

In this paper, we consider the user define support value which acts as a constraint to mine useful and interesting item sets from large data sets D.

C. *The Proposed Algorithm*

Since the original database contains a large number of items, a transaction t_l of a data set D is of the form, $t_l=\{i_{l1},i_{l2},\ldots,i_{lk},\ldots,i_{ln}\}$. The value of item i_{lj}, $j \in 1...n$ is either 1 or 0, depends on either it is present or absent in transaction t_l.

Individual Representation.

An individual is represented by the set of items of the form $individual_i$=v, where $individual_i$ is the i^{th} individual. Here v is a value from its item sets domain. For example, a datasets D contains 4 items and 1000 transactions. Let's say, transaction k contains $item_2$ and $item_4$. This transaction represents the node (2, 4) in lexicographic tree of Fig. 1., and the individual coding of this node is:

0	1	0	1

Population Generation.

The population of genetic based system is generated as follows: for first individual, it considers the whole domain in lexicographic tree. And for the following individuals it considers those item sets which are not classified in frequent or infrequent ones. When the individual is generated it classifies into frequent or infrequent ones by using the user define support value. Through this way the search space become narrower for generation of next population.

Genetic Operators.

To improve the quality of next individual, crossover and mutation operator is used to transform one individual into another one. At the initial stage of population generation, two parent individuals are selected randomly from the domain of item sets in lexicographic tree and after applying crossover operator two new offspring are generated. Mutation changes a bit randomly in each segment of the individuals for the improvement of new offspring.

Fitness Function.

The fitness function of this proposed method provides fitness value of an individual which is equal to the support value of an item set i.e. $f_{individual_i}$ = support, where $f_{individual_i}$ is the fitness value of individual i. When an individual is generated, support value of that individual is counted from the datasets. If the fitness value of an item set satisfies the user defined support value i.e. $f_{individual_i} \geq$ min supp, then this item set is classified as frequent item set and store it in an array called FI Superset Member. Otherwise it will save in an array, called NFI Subset Member. Member of FI Superset Member array are the frequent item sets of a data sets and always replaced by the supersets of member item sets. Similarly, member of NFI Subset Member array are the infrequent item sets of a data sets and always replaced by the subsets of member item sets.

A prototypical genetic algorithm based scheme is followed by the proposed method. Minimum support value (min supp), number of generation (NbGen), mutation rate (MR), crossover rate (CR), a dataset (TuplesNb) are the inputs of the algorithm.

```
Fitness (item set) Function
Temp_Fitness = SupportCount(item set)
// Count the support value of item set from given dataset
if Temp_Fitness ≥ min_supp then
   return +Temp_Fitness, item set
else
   return –Temp_Fitness, item set
```

MFItemsets **Function**

Input: min_supp, NbGen, MR, CR, dataset composed of TuplesNb

Output: Maximal frequent item sets MFI, NbTestingNodes

Generate a random population

While i ≤ NbGen **do**

Select two parents from generated individuals and applying crossover and Mutation operators to get new two offspring

for each individual

check FI_Member_Add array, if this individual or any of its subset is in this array

check NFI_Member_Add array, if this individual or any of its superset is in this array

If none of the above array contain this individual or any of its subset or superset **then**

Fitness_Value = Fitness (individual)

NbTestingNodes++

if Fitness_value □ 0 **then**

Update FI_Member_Add array

else

Update NFI_Member_Add array

i++

MFI = FI_Member_Add

//FI_Mmeber_Add array contain the latest maximal frequent item sets

return MFI, NbTestingNodes

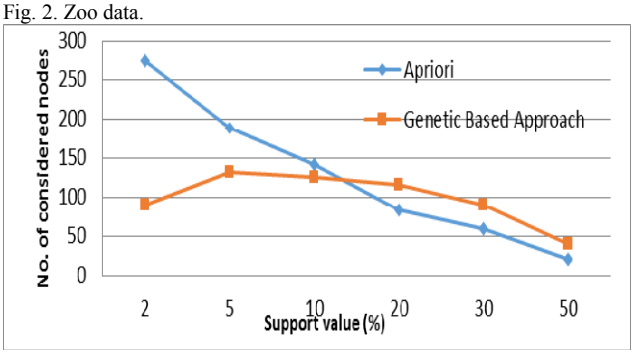
IV. EXPERIMENTAL RESULTS AND PERFORMANCE ANALYSIS

To verify the performance of the proposed method, we have used a most popular algorithm named Apriori algorithm for finding maximal frequent item sets for a comparison study. Both of these algorithms have been applied on the same datasets. C programming language is used for coding both of the algorithms. The experiments were performed on an Intel(R) core i5-3210M CPU @2.50GHz, 4 GB RAM running on Windows 7 Enterprise. Microsoft Visual Studio 2012 was used to compile the code of the proposed method. The experiments were carried out on Real data sets as well as synthetic datasets [13]. Real datasets were taken from the University of California at Irvine (UCI) machine learning repository (http://archive.ics.uci.edu/ml/datasets.html). In this experiment, we consider the following datasets T8I5D100K, T6I4D100K, Zoo, TicTacToe. For synthetic dataset, T10I5D100K where T represents the average size of the transaction is 10, I represents the average size of the maximal frequent item set is 5 and number of transactions is 100,000. Zoo and TicTacToe data sets contain 17 and 9 attributes and number of instances are 101 and 958, respectively. We use the following GA parameters to conduct the experiments. We consider the population size, popsize = 100 and initial population is produced by random generation. Other GA parameters such as selection (P_s), crossover (P_c), and mutation (P_m) probability are defined as follows: P_s=0.95, P_c=0.85 and P_m = 0.01.

Different support values were applied on these datasets to check how many nodes have been tested, and the numbers of individuals have been generated to get the exact number of maximal frequent item sets, run times, and so on. Here run time is the total execution time. The purpose of this new approach is for converging to a solution as fast as possible. A full experiment on these databases was conducted, demonstrating the proposed method's ability to yield solutions rapidly by accessing the databases for fewer numbers of nodes in a lexicographic tree. Unlike Apriori, the proposed method generates an individual X in any level which satisfies a minimum support value, then all the other subsets of X in any level will be automatically pruned which dramatically reduces the time for accessing a large database. This is also true the other way around: if the propose method generates an individual Y in any level which does not satisfy a minimum support value, then all the other supersets of Y in any level will be automatically pruned.

With Apriori algorithm, one would test all the nodes in a specific level and generate a candidate set. This candidate set generation needs a long time for finding maximal frequent item sets.

Fig. 2. Zoo data.



From the experimental results it can conclude that the proposed mining algorithm, calculated the support value for much fewer numbers of nodes than the conventional Apriori algorithm, especially when the support value was low. But for higher support value, Apriori gets the solution at level k which is near to the root node in the lexicographic tree. In that case it considers fewer number of candidate item sets than the proposed algorithm. The length of the maximal frequent item sets depends on the support value. A Lower support value provides longer patterns. From longer patterns, users can get a better idea about the relationships among frequent item sets. In

that case our proposed mining algorithm outperforms the conventional Apriori algorithm.

Fig. 3. TicTacToe data.

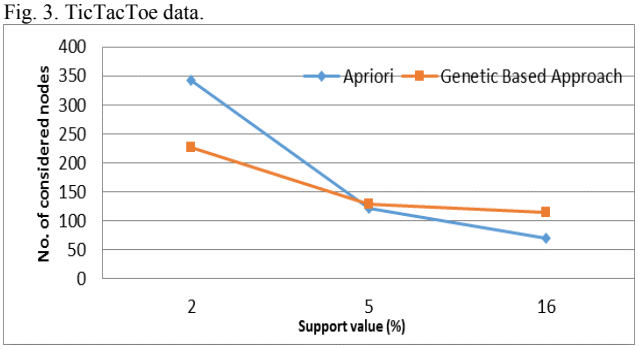


Fig. 4. T8I5D100K

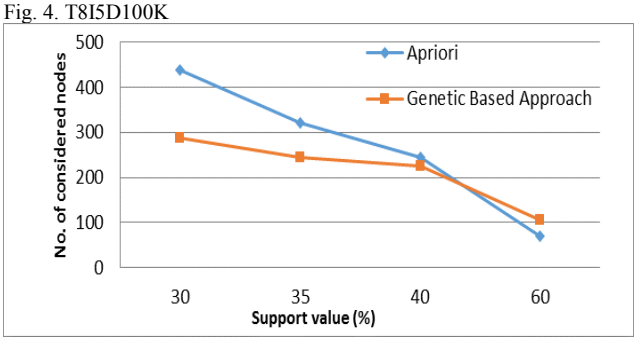
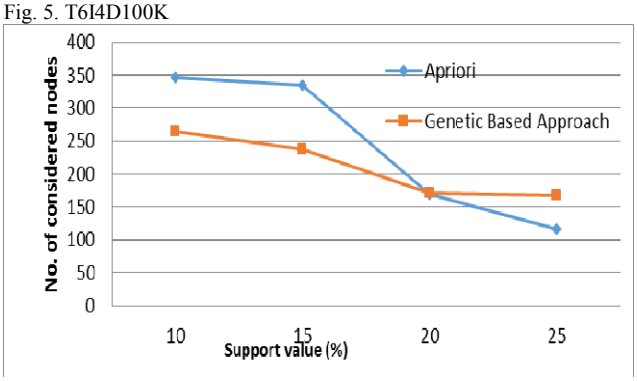


Fig. 5. T6I4D100K



V. CONCLUSIONS AND SUMMARY

In this paper, we have developed a genetic based approach and compared the results with the results given by the Apriori algorithm for mining maximal frequent item sets. We have obtained the results through experimental analysis on real data sets using both of these algorithms. Several advantages have been demonstrated by the experimental analysis of this algorithm in comparison with Apriori algorithm, which are as follows:

- It gives better results than the Apriori algorithm by accessing large data sets for less numbers of nodes especially when the support value is set low by the users.
- For large data sets and low support value, both of these algorithms give the same solution by giving the same number of maximal frequent item sets. To get this solution Apriori considers a large number of candidate item sets with respect to a genetic based approach.
- For large data sets and high support values, Apriori performs better than a genetic based approach, since the genetic algorithm uses global search mechanism. Apriori uses a level by level search procedure and it gets the solution by accessing less numbers of nodes because solution is near the root node. The nodes close to the root of lexicographic tree have higher support values.
- Low support value generates a long size frequent pattern which provides information like frequency of an exponential number of smaller sub patterns. In that case a genetic based approach performs better than other existing algorithms.
- The experimental results of a genetic based approach demonstrate the effect of generations of individuals, and prune all the subsets and supersets in a lexicographic tree, which is cost effective in the case of counting the support value and reducing the search space dramatically.

The other areas which should be focused for further research include developing genetic operators in such a way that it will help to reduce generating the same individuals and it can increase valid individuals for further generation. Considering repetitive individuals sometimes takes a longer time to get the solution.

ACKNOWLEDGMENT

This research work was funded by School of Engineering and ICT, University of Tasmania, Australia, and website: http://www.utas.edu.au/cricos, under CRICOS Provider Code 00586B.

REFERENCES

[1] M. M. J. Kabir, S. Xu, B. H. Kang, and Z. Zhao, "A Novel Approach to Mining Maximal Frequent Itemsets Based on Genetic Algorithm," in *International Conference on Information Technology and Applications (ICITA)*, 2014.

[2] R. C. Agarwal, C. C. Aggarwal, and V. V. V Prasad, "A Tree Projection Algorithm For Generation of Frequent Itemsets," *Parallel Distrib. Comput. Spec. Issue High Perform. Data Min.*, vol. 61, no. 3, pp. 350–371, 2001.

[3] A. Salleb, Z. Maazouzi, and C. Vrain, "Mining Maximal Frequent Itemsets by a Boolean Based Approach," in *European Conference on Artificial intelligence*, 2002, pp. 285–289.

[4] J. Han, J. Pei, and Y. Yin, "Mining Frequent Patterns without Candidate Generation," *ACM SIGMOD*, vol. 29, no. 2, pp. 1–12, 2000.

[5] J. Hipp, U. Güntzer, and G. Nakhaeizadeh, "Algorithms for association rule mining—a general survey and comparison," *ACM sigkdd Explor. ...*, vol. 2, no. 1, pp. 58–64, 2000.

[6] R. J. Kuo and C. W. Shih, "Association rule mining through the ant colony system for National Health Insurance Research Database in

Taiwan," *Comput. Math. with Appl.*, vol. 54, no. 11–12, pp. 1303–1318, Dec. 2007.

[7] J. H. Holland, *Adaptation in Natural and Artificial Systems*. Ann Arbor: University of Michigan Press, 1975.

[8] K. F. Man, K. S. Tang, and S. Kwong, "Genetic Algorithms : Concepts and Applications," *IEEE Trans. Ind. Electron.*, vol. 43, no. 5, 1996.

[9] D. Beasley, D. R. Bull, and R. R. Martin, "An Overview of Genetic Algorithms : Part 1 , Fundamentals," *Univ. Comput.*, vol. 15, no. 2, pp. 58–69, 1993.

[10] M. Srinivas and L. M. Patnaik, "Genetic Algorithms: A Survey," *Computer (Long. Beach. Calif).*, vol. 27, no. 6, pp. 17–26, 1994.

[11] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 1989.

[12] Z. Michalewicz, *Genetic algorithms + data structures = evolution programs*. Berlin: Springer, 1992.

[13] R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules," in *20th International Conference on Very Large Data Bases*, 1994, pp. 487–499.

[14] D.-I. Lin and Z. M. Kedem, "Pincer-Search: A New Algorithm for Discovering the Maximal Frequent Set," in *6th International Conference on Extending Database Technology*.

[15] R. J. Bayardo, "Efficiently Mining Long Patterns from Databases," *ACM SIGMOD*, pp. 85–93, 1998.

[16] R. C. Agarwal, C. C. Aggarwal, and V. V. V. Prasad, "Depth first generation of long patterns," *Proc. sixth ACM SIGKDD Int. Conf. Knowl. Discov. data Min. - KDD '00*, vol. 2, pp. 108–118, 2000.

[17] D. Burdick, M. Calimlim, and J. Gehrke, "MAFIA: a maximal frequent itemset algorithm for transactional databases," *Proc. 17th Int. Conf. Data Eng.*, no. X, pp. 443–452.

[18] K. Gouda and M. J. Zaki, "GenMax : An Efficient Algorithm for Mining," *Data Min. Knowl. Discov.*, vol. 11, no. 3, pp. 223–242, 2005.

[19] B. Alataş and E. Akin, "An efficient genetic algorithm for automated mining of both positive and negative quantitative association rules," *Soft Comput.*, vol. 10, no. 3, pp. 230–237, Apr. 2005.

[20] W. Dou, J. Hu, K. Hirasawa, and G. Wu, "Quick response data mining model using genetic algorithm," *2008 SICE Annu. Conf.*, pp. 1214–1219, Aug. 2008.

[21] A. Salleb-aouissi, C. Vrain, C. Nortet, X. Kong, and D. Cassard, "QuantMiner for Mining Quantitative Association Rules," *Mach. Learn. Res.*, vol. 14, no. 1, pp. 3153–3157, 2013.

[22] A. Salleb-aouissi, C. Vrain, and C. Nortet, "QuantMiner : A Genetic Algorithm for Mining Quantitative Association Rules," in *20th International Joint Conference on Artificial Intelligence*, 2007, pp. 1035–1040.

[23] J. Huang, Y. Che-tsung, and C. Fu, "A Genetic Algorithm Based Searching of Maximal Frequent Itemsets," in *International conference on artificial intelligence*, 2004.