A Hybrid GeneticMax Algorithm for Improving the Traditional Genetic Based Approach for Mining Maximal Frequent Item Sets

Mir Md. Jahangir Kabir† , Shuxiang Xu††, Byeong Ho Kang††† , and Zongyuan Zhao††††

School of Engineering and ICT, University of Tasmania, Australia

Summary

Mining Frequent item sets is one of the most useful data mining methods which discovers important relationships among attributes of data sets. Initially it was developed for market basket analysis, but these days it is used to solve any task where discovering hidden relationships among different attributes is required. Mining frequent item sets plays a vital role for generating association rules, finding correlations and many more interesting relationships among different sort of data. A major challenge in the frequent item set mining task is that it generates a huge number of frequent sub item sets from dense data sets. Researchers proposed mining maximal frequent item sets to overcome this problem. Maximal frequent item sets contain the information of an exponential number of frequent sub item sets since if an item set is frequent each of its sub item sets is also frequent. Very few studies have applied evolutionary algorithms to mine maximal frequent item sets using thorough experimental analysis. In a previous study, we showed the efficiency of using a genetic based approach named GeneticMax to find maximal frequent item sets. In this study we will introduce a new algorithm name, hybrid GeneticMax, which uses local search along with a genetic algorithm to mine maximal frequent item sets from large data sets. The purpose of using the genetic algorithm is that this algorithm based approach is robust and the existing genetic based method which is working fine for a specific problem can be improved by hybridizing it. Experiments are performed on different real world data sets as well as on a synthetic data set. Our new scheme compared favorably to existing GeneticMax under certain conditions which are being evaluated.

Key words:

Association rule mining, Maximal frequent item sets, Genetic Algorithm, Lexicographic tree, Data mining.

1. Introduction

Mining association rules is one of the important and essential issues in different data mining applications. This task discloses hidden and important relationships among frequently appearing item sets in large data sets. The problem of mining association rules can be divided into two steps. The first and most important step is mining frequent item sets and the next step is generating association rules based on frequent patterns. The overall performance of generating association rules depends on the efficiency of the first step since mining frequent item sets is the most time consuming task. A challenging issue for mining frequent patterns is that it often generates a large number of frequent sub items especially when users set the threshold value low [1]. Large frequent patterns contain a huge number of frequent sub patterns. Since if a pattern is frequent all of its sub patterns are also frequent. To overcome this problem researchers proposed mining maximal frequent item sets. An item set x is a maximal frequent item set if it is frequent i.e. if it satisfies user defined support value and no superset of this item x set is frequent.

A large number of research is devoted to mine maximal frequent items which have led to a variety of efficient algorithms for this task. Although MaxMiner, Apriori, breadth first search, MAFIA are the well-known methods for mining maximal frequent item sets but there exists a large number of different algorithms which act as alternatives. Very few studies showed impact of evolutionary algorithms on mining maximal frequent items through vast analysis of experimental results. For this study we propose a new algorithm which is based on a genetic algorithm for mining maximal frequent items. This approach will be applied to different real data sets and the experimental analysis will demonstrate the effect of evolutionary algorithms on mining maximal frequent items. The major advantage of hybrid evolutionary algorithm based approach is that it follows global optimization methods and performs well for large data sets. This research differs from existing studies [2] in the following facets: 1) It sorted out infrequent items from 1-item sets using local search; 2) hybrid GeneticMax avoids level by level searching and uses a lexicographic tree as a search space; 3) The principles of a genetic algorithm is used by this approach, which generates new chromosomes by

Manuscript received October 5, 2014
Manuscript revised October 20, 2014

applying genetic operators on parent chromosomes. If the generated chromosome is frequent then all the subsets of this chromosome are pruned and if not then all the supersets of this chromosome are pruned. Through this way the search space of the lexicographic tree becomes narrower and narrower. This approach dramatically reduces the required time for accessing large data sets for calculating the support value of unnecessary item sets for mining maximal frequent item sets.

This paper comprises the following sections: basic notation, problem definition, maximal frequent item sets and search space are defined formally in section two. Section three represents the previous research works. The proposed algorithm and methodology of search technique are described in section four. Experimental results are shown in section five. Finally, section six summarizes the whole research work.

2. Basic Notions

Frequent item set mining is a famous data mining method which was basically developed for analyzing market data. The main aim of this task is to find regularities of customers' shopping behavior in supermarkets, online shop and mailing orders of companies. Specifically, it tries to mine frequent item sets that are frequently bought together by the customers from large transactional data sets. These sets of associated items help the organization to make decisions in which bundles of item sets should be offered, which bundles of items are popular to the customers and need to be arranged on the same shelf, or which bundles of products should be bought by the industry frequently, which will benefit industries by selling those products and so on. These days mining frequent item sets play a vital role in different data mining tasks such as mining association rules, classification techniques, finding correlations among attributes of a data sets, clustering and many other interesting regularities among data.

Formal definition of frequent item set mining is as follows: Given Item base $B = \{i_1, i_2, \ldots\ldots, i_{n-1}, i_n\}$, which is a set of different items and data sets $D = \{d_1, d_2, \ldots\ldots, d_{m-1}, d_m\}$, where D is a transactional, or other type of data sets such as car, zoo, gaming data sets and so on. For zoo data sets, an item could be hair, feathers and so on. Top-left-square, bottom-right-square are the items for TIcTacToe game. So the item base represents the set of all items offered by the data sets. Any subset of an item base B is referred to by the term *item set*. For example, if we consider transactional data sets, each transaction in a data set D is an item set, which is bought together by a customer on any day. Transaction id (tid) may be used to enhance each transaction. Item base B can be represented by the union of all transactions i.e. $B = \cup_{i[1\ldots m]t_i}$.

The support value of an item set I is how many times this item set appeared in a data set. Let x is an item set. The support value of x is $supp(x) = \left|\frac{x}{D}; x \subseteq d_i, d_i \in D\right|$.

An item set is frequent if its support value satisfies a user defined support value, min_supp i.e. supp(x) ≥min_supp.

The main problem of mining frequent item sets is that it often generates a large number of item sets which satisfy min_supp threshold, especially for low min_supp value. To solve this problem the researcher proposed different restrictions on the set of frequent item sets. Mining maximal frequent item sets is one of the famous methods of those suggested proposals. An item set x is maximal in data set D, if x is frequent and there exists no superset y such that y⊇ x, is frequent in data set D.

As an illustration, figure 1 shows a small transaction data set containing 8 transactions of item base B = {a, b, c, d, e}.

a)Transactions	
1:	{a,b,d}
2:	{c,d,e}
3:	{a,c,d}
4:	{a,b,c,e}
5:	{b,c,e}
6:	{b,c,d}
7:	{d,e}
8:	{a,b,d,e}

b) Frequent item sets (min_supp = 2)			
0 item	1 item	2 items	3 items
{}: 8	{a}: 4 {b}: 5 {c}: 5 {d}: 6 {e}: 5	{a,b}: 3 {a,c}: 2 {a,d}: 3 {a,e}: 2 {b,c}: 3 {b,d}: 3 {c,d}: 3 {c,e}: 3 {d,e}: 3	{a,b,d}: 2 {a,b,e}: 2 {b,c,e}: 2

c) Maximal frequent item sets
{a,b,d}, {a,b,e}, {b,c,e}

Fig. 1 a) A simple transaction data set of 8 transactions containing 5 items, b) Frequent item sets based on a user defined threshold value, min_supp = 2 and c) maximal frequent item sets based on table b).

If the size of an item base B is α, then it will generate 2^{∞} candidate item sets. It is computationally infeasible to determine the support value of all the candidate item sets and filtering out the infrequent items since a small supermarket or industry generally offers thousands of various items.

To make the search techniques efficient, different concepts have been proposed by the researchers. One of the concepts which is still famous is Apriori property. The main theme of this property is that, all the supersets of infrequent item sets are not frequent [3]. For this study, we used this property to find maximal frequent item sets. The only difference is that Apriori considers level by level searching whereas hybrid GeneticMax generates frequent

item sets based on the property of a parent chromosome. To mine maximal frequent item sets, we are looking for a few solutions. The search space is that space which considers all feasible solutions. Lexicographic tree [4] is the search space for the hybrid GeneticMax algorithm. Abstract representation of large item sets is done by this tree and it considers the lexicographic ordering, defined in the following way.

1) Each node of a lexicographic tree represents an item set.
2) If $I = \{i_1, i_2, \dots, i_n\}$is an item set, where items $i_1, i_2,\dots, i_n$ follows lexicographic ordering i.e. $i_1 \leq i_2$. Here $\{i_1, i_2,..., i_{n-1}\}$ is the parent of item set I.
3) The root of the tree is an empty set.
4) The tree is the left most tree i.e. item sets are arranged from left to right.
5) The node which is close to the root has a higher support value than the node which is far from the root.
6) There is a non-linear line in the tree called "cut" which separates infrequent item sets from frequent ones. This cut is defined in the tree based on a user defined threshold value i.e. min_supp.
7) nodes above the cut are frequent item sets whereas nodes below the cut are infrequent item sets.

Fig. 2. Shows a lexicographic tree of four items.

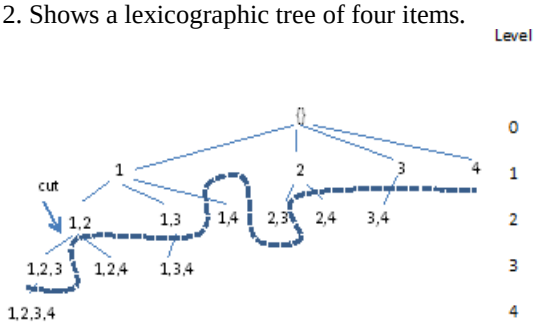

Fig. 2 Lexicographic tree of four items

3. Related Works

It is well known that the Apriori algorithm has a level by level searching mechanism and tests candidate item sets in a breadth fast manner. It determines the support value of all the candidate item sets at level k and filters out infrequent item sets and then it moves to its next (k+1) level. Since it considers candidate item sets at each level and frequently scans the data sets made, this algorithm is costly, especially for a long pattern [5].

A Bi-directional method is applied for the Pincer-Search algorithm [6] to traverse a lattice which follows both bottom-up and top-down. It applies a pruning method to find maximal frequent item sets, two properties are followed for this:

1) It prunes the subsets of frequent item sets
2) Supersets of infrequent item sets are also pruned.

The MaxMiner search method uses Breadth first traversal approaches. It uses a look-ahead method to prune the branches of a tree. To limit the number of passes MaxMiner uses breadth first methods for look-ahead, which considers superset pruning techniques, and performs better for depth first search approaches [4].

Depth first traversal is applied for the DepthProject algorithm on a lexicographic tree. Various superset pruning techniques are applied for this approach. Dynamic reordering methods are used for ordering child nodes. Infrequent items are trimmed out from each node's tail and through this way it reduces the size of the search space. For this approach, Post pruning methods are used to eliminate non-maximal frequent item sets [7].

Burdick, Calimlim, and Gehrke [8] proposed an algorithm named MAFIA which is an extended idea of DepthProject. MAFIA uses vertical bitmap representation which is similar to DepthPorject, and operations among the items are applied for counting the support value of an item set. For example, if there are 4 items in a tuple of a data set, then the vertical bitmap representation of that data set is as follows:

A	B	C	D
1	1	0	1
1	0	1	1
1	1	1	0
1	0	1	0
0	1	0	1

Fig. 3 Vertical bitmap representation.

Bit vectors of items A, B, C, D of Fig. 1. are 11110, 10101, 01110, 11001 respectively. Bitwise AND operation between the bit vectors is applied for counting the support value of item sets. In the above example, 10001 is the result of bitwise AND operation of bit vectors B and D which is 10101 & 11001 respectively. So the support value of the item set {B, D} is 2. If A is an item which we need to add with the previous item set {B,D}, bit wise AND operation will need to perform on previous result with the bit vector of A, which is 10001 && 11110 and the result is 10000. That is, the support value of the item set {A,B,D} is 1. Like MaxMiner algorithm, MAFIA also uses a depth first method along with an efficient pruning method.

GENMAX is a novel approach which is proposed by Gouda and Zaki [9] to find maximal item sets. A novel technique named Progressive Focusing is used by this approach. Local maximal frequent item sets (LMFI) are used by this technique for making comparisons with frequent item sets (FI). Through this way non maximal frequent item sets are identified which decrease subset testing phase. Vertical representation of a data set is used by GENMAX and instead of bit vector, for each item set transaction identifier set (TIS). Cardinality of each item

set's TIS is used to count the support value of an item set. Experimental analysis of this algorithm shows that GENMAX performs better than other existing algorithm on different data sets.

An efficient genetic algorithm is designed by Bilal Alataş and Erhan Akin [10] for mining association rules. This algorithm is used to mine both positive and negative quantitative association rules. Generally, frequent patterns are used for generating association rules. Their proposed method is different than other approaches. Without generating frequent patterns it mines association rules. A Uniform operator, a new genetic operator, is used in this method for ensuring genetic diversity.

In [11], a quick response data mining model has been proposed. This approach used a genetic algorithm and it gives lots of flexibilities to users. If a data set contains a large number of item sets, the higher relationship among those items can generate a long frequent pattern. These type of data sets generates a large number of candidate item sets and the Apriori algorithm takes huge amounts of time to mine frequent item sets from these data sets. This approach is more user concerned and scans the data sets for those item sets users are interested in, through this way it avoids to considering huge candidate item sets.

Hipp, Guntzer and Nakhaeizadeh [12] showed the performance analysis of Apriori and other existing famous algorithms of present day. Although it was invented long time ago but still Apriori is one of the famous algorithm and it performs better than other existing algorithms like Eclat, Partition, DIC and so on for large value of minsupp. On the otherhand, other algorithms perform better than Apriori for small value of minsupp. Finally, they concluded that no algorithms fundamentally beating each other. After analyzing, they showed that the run time behavior of all the algorithms are similar as it is expected.

For mining quantitative association rules researchers proposed an approach which is used to find "good" intervals in association rules. This approach is based on a genetic algorithm called QUANTMINER [13], [14]. They used a genetic algorithm to optimize support and confidence value in this system. Experiments performed on artificial and real life data sets and the analytical results of this experiment showed the usefulness of this system as an exploratory and interactive data mining tool.

4. Practical Implementation

As we mentioned above the core of this study is an evolutionary algorithm where each individual represents an item set. In the following sections we will undergo the general view of the hybrid GeneticMax algorithm, representation of each chromosome or individual, fitness function of each individual, generation of new individual using genetic operators and item sets enumeration process.

4.1 The Purpose of Using Genetic Algorithm

Genetic algorithm plays a vital role for this study which simulates the natural behavior of biological organisms [15]. Genetic algorithm based techniques are robust and can be used to solve wide range of problems including those which are hard to solve by other methods. Researchers concluded that, it is not guaranteed that GA always provide optimum solution to a problem rather it provides "acceptably good" solution to a problem which is solved by other method "quickly". Existing methods which are working well as a solution for a particular problem, improvement of those methods can be done by hybridizing with genetic algorithm [16].

4.2 Hybrid GeneticMax Algorithm

The hybrid GeneticMax algorithm is based on the theory of genetic algorithms. The structure of lexicographic tree is based on a user defined threshold value as defined in [2], this study will use this search space to find maximal frequent item sets. For this algorithm, data set D is the input and it returns maximal frequent item sets. In a brief, data set D contains a large number of transactions i.e. $D = \{t_1, t_2, \dots, t_{n-1}, t_n\}$ and each transaction contains items.

The form of transaction t_1 is as follows: $t1 = \{i_{11}, i_{12}, \dots, i_{1j-1}, i_{1j}\}$. Based on the presence or absence of an item i_{1k}, $k \in 1 \dots j$ is either 1 or 0.

Algorithm Hybrid GeneticMax
Step 1: Find infrequent items from 1-item sets and Initialize NFI array.
Step 2: Find maximal frequent item sets from k-item sets where k>1. 1. Set generation number NbGN = δ and nGN = 0 2. Generate initial population 3. **While** (nGN < NbGN) 4. Compute fitness value using fitness_function (individual) 5. **If** (fitness_function (individual) ≥ min_supp) 6. **If** subset of this individual is in FI array then replace it by the current individual 7. **Else** add individual in FI array 8. **Else If** superset of this individual is in NFI array then replace it by the current individual 9. **Else** add individual in NFI array 10 Select two parent individuals 11. Generate new individual, applying crossover and mutation operators on parent individuals 12 nGN++ 13 **end While** **end**

Fig. 4 Hybrid GeneticMax algorithm

Firstly, it will use local search to find infrequent items from 1-item sets and initialize NFI array. NFI array will contain infrequent item sets. Secondly, it will use the genetic algorithm based approach to find maximal frequent item sets from k item sets where k>1. The first step is to set a generation number by using the variable name (NbGN). Other parameters which should be used as an input along with given data sets D, are mutation rate (MR), crossover rate (CR), minimum support value (min_supp).
Each individual is frequent or infrequent depending on the fitness value of that individual. If the given individual is frequent based on fitness value of an individual, then it stores the individual in an array for further checking. The array of hybrid GeneticMax algorithm is classified into two groups. 1) array of frequent item sets name FI and 2) array of infrequent item sets name NFI. Finally, the member of FI array are the maximal frequent item sets. The basic structure of hybrid GeneticMax algorithm is shown in Fig. 4.

4.3 Representation of individuals

A transaction is an item set, which shows the presence or absence of items. An individual represents a transaction. A simple form of i^{th} individual is $individual_i$= attributes. The value of an attribute comes from an item set domain of a data set D. Real codification is used to represent individuals. An individual of hybrid GeneticMax algorithm is a k-item set i.e. k items are present in the individual, where k≥1. If the size of an item base B is n, it will generate 2^n different item sets.

$Item_1$	$Item_2$	….	$Item_{n-1}$	$Item_n$

Fig. 5 Representation of an individual for n- items

4.4 Generation of Population

At the initial stages of the hybrid GeneticMax algorithm, it needs an initial population to generate the next population. It considers the whole item set domain of a lexicographic tree for generating initial individuals. Random generation of the initial population means the algorithm starts from any node of the lexicographic tree and classifies this item set as either frequent or infrequent based on a user defined threshold value. For generating next individuals, the initial population will act as parent individuals. Genetic operators will apply on parent individuals to create new individuals.

4.5 Genetic Operators

Two essential operators named crossover and mutation are used to improve the quality of offspring. Parent individuals are selected randomly from the initial population. After random segmentation of parent individuals, a crossover operator is used to generate new individuals. A Mutation operator is performed in the segmented region and new two offspring are generated. Fig. 6. shows an example of the process of generating new offspring using genetic operators.

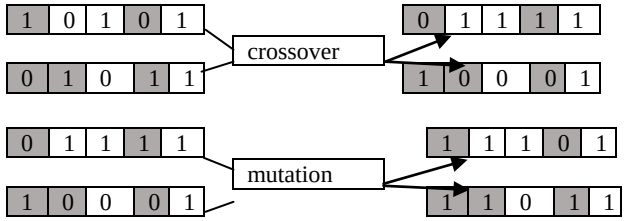

Fig. 6 Process of generating new offspring using genetic operators

4.6 Fitness function

The Lexicographic tree is classified into two areas based on a user defined threshold value, 1) frequent and 2) infrequent.
All the individuals have support values. An individual is fittest for the frequent area of a lexicographic tree whether the support value of this individual is greater or equal to the user defined threshold value. If the support value of an individual is greater than or equal to min_supp, then the fitness function will return a positive value for this individual otherwise it will return a negative value i.e.
If support (individual)≥min_supp **then** fitness (individual) = +1
Else fitness (individual) = -1.

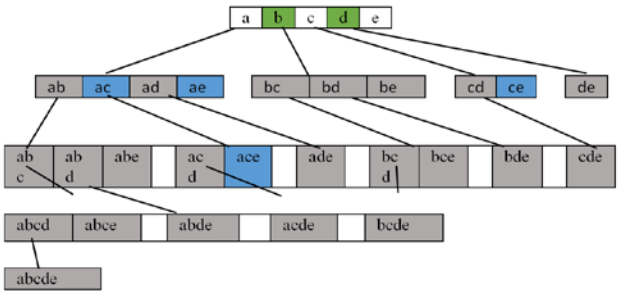

Fig. 7 Illustration of the hybrid GeneticMax approach to find maximal frequent item sets. Initially it sorted out infrequent items {b,d} from 1 – item sets {a},{b},{c},{d},{e} which is shown by green box. When it identify infrequent items then all the sub item sets of these items will be invalidate chromosomes which are shown by black box. After then traditional GeneticMax will apply to find maximal frequent item sets from item sets {ac}, {ae}, {ce} and {ace} which is shown by blue box.

4.7 Item Set Enumeration

If data tuples contain long item sets it generates huge candidate item sets which finally reduces the efficiency of a solution. A long item set enumerates combinatorial number of shorter, frequent sub item sets. For example, a data tuple contains 50 item sets, such as $\{i_1, i_2, i_3, \ldots, i_{50}\}$ which enumerate $\binom{50}{1}$ frequent 1-item sets: $(i_1, i_2, \ldots, i_{50})$, $\binom{50}{2}$ frequent 2-item sets: (i_1, i_2) , (i_1, i_3) , … , (i_1, i_{50}), (i_2, i_3), (i_2, i_4),…, (i_2, i_{50}) and so on.

Lemma 1: If the length of an item set is n, then it enumerates $2^n - 1$ frequent sub-item sets.

This is too huge for a computer to compute and store if the length of an item set is long. For each sub-item set, Apriori algorithm needs to be used to scan the data sets and calculate the support value of that item set which increases the computational time of the algorithm and decreases the efficiency of it. This algorithm works fine if the position of the solution is near to the root in lexicographic tree. On other hand if the position of the solution is far from the root then it needs to consider huge amount of candidate item sets for calculating support value to get reach to the solution nodes.

For this reason computational time increases as it considers huge amount of candidate item sets especially if the position of the solution is far from the root. To overcome this low efficiency of Apriori algorithm, GeneticMax uses global search mechanism which starts from any position of the lexicographic tree. If the generated individual is infrequent then all of its sub item sets are infrequent and those item sets are automatically pruned. Similarly if the generated individual is frequent then all of its sub item sets are frequent and those item sets are automatically pruned. Through the way, the search space of GeneticMax algorithm becomes narrower and narrower. Hybrid GeneticMax improves the traditional GeneticMax algorithm by introducing local search. Initially, it sorted out the infrequent items from 1- item sets. The effect of sorting out the infrequent items at initial stage is shown through Fig. 7.

5. Experimental Results

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 new GeneticMax. The program is written in C language. Four datasets including Plant Cell Signaling, Random Number #1, Synthetic, Zoo data sets were used to test the performance of the new GeneticMax approach. Different support values were applied on these datasets to check how many nodes have been tested, and the number of chromosomes have been generated to get the exact number of maximal frequent item sets, run times, and so on. Total execution time of the program is defined by the term run time. Three main features are embedded by the new approach: i) it sorts out infrequent items from 1- item sets, ii) there is a superset-subset relationship in both positive and negative boundaries in a lexicographic tree for pruning invalid chromosomes, and iii) the use of a genetic algorithm which uses a global search mechanism. The purpose of sorting out infrequent items from 1-item sets is that, it dramatically reduces the search space for finding the solution. Because if an item is infrequent all of its sub item sets are infrequent. The aim of this new approach is converging to a solution as fast as possible, especially if 1-item sets contain a reasonable amount of infrequent items and the solution resides in the deep level of the lexicographic tree instead of near the root. A full experiment of the new approach on the above mentioned data sets was conducted, demonstrating this approach's ability to yield solutions rapidly by accessing the data sets for a few numbers of nodes in a lexicographic tree.

As we see from the previous discussions, all the nodes in each level of a lexicographic tree are tested by Apriori algorithm and those nodes from a level which do not satisfy user defined support value are pruned. In traditional GeneticMax, if it generates an individual X in any level which satisfies a user defined support value, then all other subsets of X in any level will be automatically pruned. This mechanism is also true the other way around: if it generates an individual Y on any level which is infrequent i.e. which does not support user defined support value, then all the supersets of Y in any level of a lexicographic tree will be automatically pruned. The hybrid GeneticMax embeds all the features of the traditional GeneticMax algorithm including local search mechanism for finding infrequent item sets from 1- item sets of a large data set.

We tested the algorithm on different data sets. These data sets were taken from the University of California at Irvine (UCI) machine learning repository (http://archive.ics.uci.edu/ml/datasets.html) and data sets of the University of Regina (http://www2.cs.uregina.ca/~dbd/cs831/notes/itemsets/datasets.php). We have conducted several experiments on different data sets for evaluation purposes. From the experimental results as shown in Table 1, we seem to be able to conclude that the hybrid GeneticMax considers less number of nodes to get the solution whereas traditional GeneticMax gives the same solution by considering a large number of nodes. The above statement is true if there are a reasonable amount of infrequent items in 1- item sets. If 1-item sets do not contain any infrequent items, then the local search mechanism of hybrid GeneticMax will not

work and in this case hybrid GeneticMax will perform the same as the traditional GeneticMax algorithm.

In this paper, CPU time and I/O time are both included by run time. Figure 8-11 shows the run time behavior of traditional and hybrid GeneticMax. 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 [17]. For both algorithms, the same number of chromosomes are generated. But traditional GeneticMax accesses the data set for calculating the support value for large number of nodes, whereas hybrid GeneticMax considers a small number of nodes to get the solution. For this reason hybrid GeneticMax takes a smaller amount of time than traditional GeneticMax to converge to a solution.

Table 1: Number of nodes of a lexicographic tree of Plant Cell Signaling data set, are used for getting the solution for GeneticMax and Hybrid GeneticMax algorithms

minsupp (in %)	GeneticMax	Hybrid GeneticMax
0.95	13273	76
0.8	15761	163
0.6	15761	163
0.3	15761	294
0.2	16358	2086
0.15	20151	7537
0.1	22624	8956

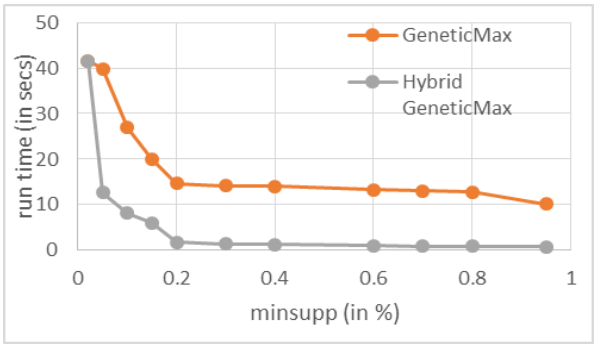

Fig. 8 Performance comparison of traditional GeneticMax versus Hybrid GeneticMax with different support values for Plant Cell Signaling data set

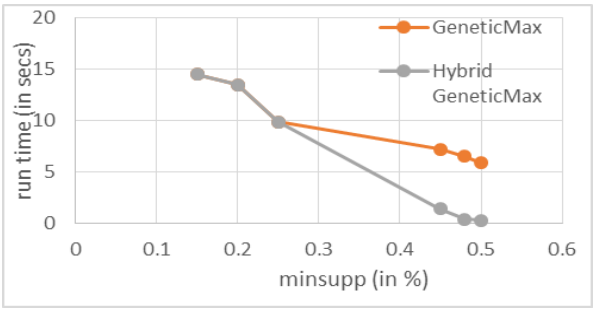

Fig. 9 Performance comparison of traditional GeneticMax versus Hybrid GeneticMax with different support values for Random Numbers #1 data set

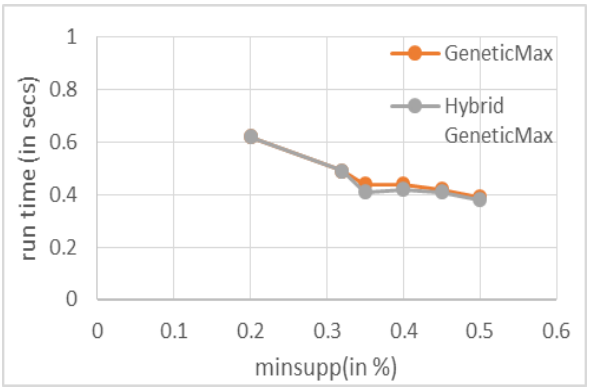

Fig. 10 Performance comparison of traditional GeneticMax versus Hybrid GeneticMax with different support values for Synthetic #3 data set

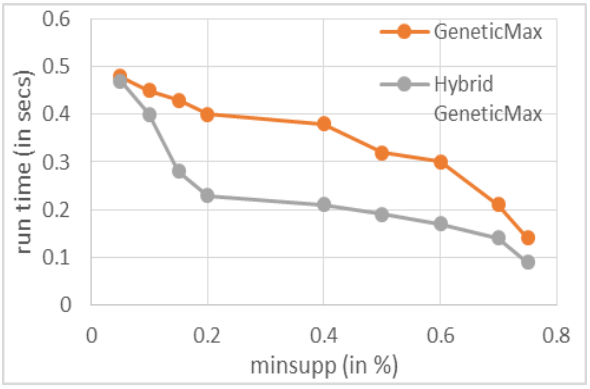

Fig. 11 Performance comparison of traditional GeneticMax versus Hybrid GeneticMax with different support values for Zoo data set

From the performance graph it can be concluded that hybrid GeneticMax outperforms the traditional GeneticMax algorithm especially if 1- item sets contain a reasonable amount of infrequent item sets. We tested both algorithms on various data sets for different support values and the experimental results show that the hybrid GeneticMax performs better than the traditional GeneticMax algorithm until it reaches a certain threshold

value of minsupp. After that 1 item sets do not contain any infrequent items, so the hybrid GeneticMax performs the same as the traditional GeneticMax i.e. both algorithms access the same number of nodes to get the solution and the computational time is the same as well.

6. Conclusions and Summary

In this paper we proposed a novel approach (named hybrid GeneticMax) based on a local search and a genetic algorithm to mine maximal frequent item sets in an efficient way. We have conducted thorough experiments on different real data sets for evaluating the performance of the traditional and hybrid GeneticMax algorithm. The experimental results demonstrate several advantages of our algorithm in comparison with the traditional GeneticMax algorithm.

1) It considers fewer item sets of large data sets to calculate support value to get the solution i.e. finding maximal frequent item sets.

2) It shows the power of using an evolutionary algorithm along with a local search mechanism for generating frequent item sets from lexicographic trees. Abstract representation of large data sets is done by a lexicographic tree based on a user defined support value, which is used as a search space for these experiments.

3) The experimental analysis of the hybrid GeneticMax shows the effect of a local search along with a global search mechanism, and it compared the results with the traditional GeneticMax algorithm. Firstly, it sorted out the infrequent items from 1- item sets using a local search mechanism and it used these infrequent item sets for further pruning methodologies. After this step, it used a global search mechanism i.e. using a genetic based approach it prunes all the subsets and supersets in both positive and negative boundary areas, which dramatically reduces search space and the cost of counting the support value of item sets.

4) The above advantages of the hybrid GeneticMax increases the searching speed and scalability of this algorithm which is shown through comparative analysis of traditional and the hybrid GeneticMax algorithm.

5) This approach outperformed the traditional GeneticMax algorithm, if there are reasonable amount of infrequent items in 1-item sets. For a certain threshold value, if there are no infrequent items in 1-item sets, then this approach performs similar to the traditional GeneticMax algorithm.

Applying genetic operators on parent chromosomes the newly generated chromosomes are called replicated chromosomes, if they are similar to the previous generated chromosomes. These replicated individuals are considered as invalid since they have been examined before. Identifying replicated individuals is a time consuming task. In future we will develop genetic operators in such a way that it will generate more valid chromosomes by reducing the generation of invalid chromosomes, which will increase its searching speed in converging to a solution.

References

[1] J. Han, H. Cheng, D. Xin, and X. Yan, “Frequent pattern mining: current status and future directions,” Data Min. Knowl. Discov., vol. 15, no. 1, pp. 55–86, Jan. 2007.

[2] 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.

[3] C. Borgelt, “Frequent item set mining,” Wiley Interdiscip. Rev. Data Min. Knowl. Discov., vol. 2, no. 6, pp. 437–456, Nov. 2012.

[4] R. J. Bayardo, “Efficiently Mining Long Patterns from Databases,” ACM SIGMOD, pp. 85–93, 1998.

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

[6] 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.

[7] 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.

[8] 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.

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

[10] 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.

[11] 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.

[12] 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.

[13] 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.

[14] 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.

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

[16] 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.

[17] 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.

Mir Md Jahangir Kabir is currently a doctoral student in School of Engineering and ICT, University of Tasmania, Australia. He received B.Sc. and M.Sc. degrees, from Rajshahi University of Engineering and Technology, Bangladesh and University of Stuttgart, Germany in 2004 and 2009, respectively. After working as a lecturer (from 2004), he is an assistant professor (from 2010) in the Dept. of Computer Science and Engineering, Rajshahi University of Engineering and Technology, Bangladesh. He received an Overseas Postgraduate Research Award from the Australian government in 2013 to research in PhD. His research interests include the theory and applications of Data Mining, Genetic Algorithm, Machine Learning and Artificial Intelligence.

Shuxiang Xu is currently a lecturer of School of Engineering and ICT, University of Tasmania, Australia. He received a Bachelor of Applied Mathematics from University of Electronic Science and Technology of China (1986), China, a Master of Applied Mathematics from Sichuan Normal University (1989), China, and a PhD in Computing from University of Western Sydney (2000), Australia. He received an Overseas Postgraduate Research Award from the Australian government in 1996 to research his Computing PhD. His current interests include the theory and applications of Artificial Neural Networks, Genetic Algorithms, and Data Mining.

Dr. Byeogn Ho Kang a computer scientist, is the Associate Professor of University of Tasmania, Australia. He received his Ph.D from the University of New South Wales, Sydney (1996), and has worked as a visiting researcher in the Advanced Research Lab. HITACHI, Japan (1995) and Kyung Hee University (2013) in South Korea. He has also worked in research and development projects with industries and research organizations, the Smart Internet Collaborative Research Centre, the US Air Force, Hyundai Steel, AutoEver Systems, LG-EDS, IPMS Technology and Asian Office of Aerospace Research Department, US in Japan. He has published more than 150 papers in journals and international conferences, editor of several books of proceedings, and guest editor for special issues of journals. He has participated in many conference organization committees, and he has been a chair of Australian AI Conference in 2006 and KM & Acquisition for Intelligent Systems (PKAW) in 2014, 2010, 2006. He is steering committee member of Pacific Rim International Conference on AI and Austrian AI conference. He is currently served as Vice President of Korean Academy of Scientists and Engineers in Australia.

Zongyuan Zhao is currently a PhD student in School of Engineering and ICT, University of Tasmania, Australia. He received a Bachelor of Computer Science and Technology from Beihang University, China (2010), and a Master of Computer Science and Technology from China Agriculture University, China (2012). His current interests include the theory and applications of Artificial Neural Networks and Data Mining.