MINING OF ASSOCOIATION RULES WITH HASH BASED PARTICLE SWARM OPTIMIZATION (HBPSO) ALGORITHM

T.Bharathi
Dept.of Computer Science
RVS Arts and Science College
Coimbatore, Tamilnadu, INDIA

Dr. A.Nithya
Dept. of Computer Science
RVS Arts and Science College
Coimbatore, Tamilnadu, INDIA

Abstract – In the current paper the Particle Swarm Optimization algorithm is being employed to extract the association rules from a large data base. The association rules are employed when a large database is present and there is a need to map the rules for an itemset that is present in large number of transactions. Generation of the candidate item set and evaluating the fitness, velocity help to find the solution to mine the association rules. A new method was proposed in the paper, namely, Hash Based Particle Swarm Optimization algorithm (HBPSO). This offers a more tight integration of the concurrently processed queries as it shares the memory data structures. In other words, the hash trees for candidate item set. By using the hash tree, one can generate the association rule in the end. To cut back on the time consumption and the cost of the computation, the method overlooked the fitness value and velocity. To precisely evaluate the performance of the proposed method, wide spread simulation study has been done.

Index Terms: Association rule mining,Hash based Particle swarm optimization algorithm, fitness, velocity

I. Introduction

The process of condensing useful data or observations from bigger data archives like relational database, data warehouses, XML repository etc is termed as Data Mining. Again data mining acts as one of the key processes of Knowledge Discovery in Database (KDD). Many a times it is considered as another term for Knowledge Discovery in Database. The preprocessing methods like, data cleaning, integration, selection and transformation, which are employed on the intended data prior to the data mining techniques. Data mining is a very vital step in the KDD, in which different algorithms are employed to produce hidden knowledge. The various data mining functions are clustering, classification, prediction, and link analysis (associations). Mining of the association rules is perhaps the most significant application of the data mining. Association rules were first introduced in the year 1993 [Agrawal1993]. They were employed to find the correlation among the item sets in a database. The basis for such correlation is the co-occurrence of the data items and not on the innate properties of the data itself.

Only when we get a acceptable result, one can present the knowledge or else the process has to be run again till the acceptable result is obtained. In traditional methods, to find out the items those are bought together repeatedly in a supermarket, one needs data like customer ID, items bought, transaction time, prices, number of each items and so on. But with data mining, only items purchased is needed. The process becomes more efficient when the data mining technique is applied as the data requirement gets smaller. As we apply different data mining techniques on the data source, we get different knowledge as the mining result. The result is then analyzed by rules like domain knowledge or concepts. If the results are not as expected, then process is repeated till we obtain the right result. Either the mining is changed or the user modifies his needs after the evaluation of the result. This makes the data mining easier and simpler to understand.

The following paper is having several segments for the ease of understanding. Section 2 has the problem description along with methods to evaluate the support. Section 3 discusses the various existing association rule mining algorithms. Section 4 contains the method to construct a hash tree. Section 5 has the proposed technique to discover the best association rules. Results and further analysis is explained in section 6. Lastly, the conclusion based on the graphs is discussed.

II PROBLEM DESCRIPTION

Agrawal first came up with the formal statement of association rule mining problem in [Agrawal et al. 1993]. Consider I=I1, I2, · · ·, where it is a set of n distinct attributes. Let T be transaction, which has a set of items such that $T \subseteq I$, DB be a database with different transaction records Ts. An association rule is an allusion in the form of $X \Rightarrow Y$, where $X, Y \subset I$ are sets of items called itemsets, and $X \cap Y = \emptyset$. X is called antecedent while Y is called consequent, the rule means that X implies Y. Support and confidence are the two key measures for association rules. We generally have large database. Customers are concerned with repeatedly purchased products that are generally the threshold values of support and confidence that are predefined by users to drop those rules that are not so interesting or useful. These are nothing but the minimal support and minimal confidence respectively. Also additional constraints can be defined by the users. Support of the rule is described as the percentage/fraction of records that contain $X \cup Y$ to the total number of records in the database. During the scanning process, the count for each item is increased by one every time the item is encountered in different transaction T in database DB. It implies that the support count does not take the quantity of the item into consideration. Consider that a buyer buys three numbers of product P1 but we only increase the support count number of {P1} by one, i.e. if a transaction contains a item then the support count of this item is increased by one. Support is calculated by the following equation:

Support(X=>Y) = Support count of (X,Y) / Total number of transaction in DB -------- Eq(1)

In an association rule, Support of an item is of statistical significance. If the support of an item is 0.2%, it implies that that 0.2 percent of the transaction contains purchasing of this item. The seller will not pay much attention to such items that are not bought frequently and hence require a high support for interesting association rules.
Before the mining process, users can specify the minimum support as a threshold, which means they are only interested in certain association rules that are generated from those itemsets whose supports exceed that threshold. However, sometimes even the itemsets are not so frequent as defined by the threshold, the association rules generated from them are still important. Confidence of an association rule is defined as the percentage/fraction of the number of transactions that contain X&Y to the total number of records that contain X, where if the percentage exceeds the threshold of confidence an interesting association rule X⇒Y can be generated.

Confidence(X=>Y) = Support(X,Y)/ Support(X) ---- Eq(2)

To measure the strength of the association rules, one uses the confidence. Consider that the confidence of the association rule X⇒Y is 90%, it implies that 90% of the transactions that contain both X & Y is previously defined by the users. From a given database, association rule mining help us to discover those rules that appeases the already predefined minimum support and confidence.
In general, an association rules mining algorithm contains the following steps:

- The set of candidate k-itemsets is generated by 1-extensions of the large (k -1)- itemsets generated in the previous iteration.
- Supports for the candidate k-itemsets are generated by a scan of the database.
- Itemsets that do not have the minimum support are eliminated and the remaining itemsets are called large k-itemsets.
- This process is repeated until no more large itemsets are found.
- The generation of association rules from those large itemsets with the constraints of minimal confidence

Let us consider that one of the large itemsets is Lk, Lk= {I1, I2, · · ·, Ik−1,Ik}. The association rules is generated as follows:The first rule is {I1, I2, · · ·, Ik−1} ⇒ {Ik}. it can be classified as interesting or not by investigating the confidence of this rule.Further rules are generated be eliminating the last items in the antecedent and inserting it to the consequent. Again confidence is employed to check for the interestingness.he above process is repeated till the antecedent becomes empty.

III EXISTING ASSOCIATION RULEMINING ALGORITHM

The Particle Swarm Optimization algorithm (PSO) is one of the key association rule mining algorithms. It is used as a substitute to the existing complex non-linear optimization problem that is based on the stochastic search algorithm. In 1995, Dr. Kennedy and Dr. Eberhart, introduced the PSO algorithm for the first time. The algorithm is stimulated by the replication of the social behavior of animals such as bird flocking, fish schooling etc. From then on, this algorithm is being used as a potent way to give solution to the optimization problems in a wide variety of applications. The basis of this algorithm is the elemental communication amongst the groups in order to divide the information when one group goes out in search of food or place or migration etc where the others in the group will not be aware of the information. As soon a way is found out, rest of the group will follow. But, the PSO cannot be applied in all situations and needs enhancement.
PSO is a Member of Swarm Intelligence
The composite behavior of decentralized, self-organized systems is the basis for the Swarm intelligence (SI), which might be again innate or phony. Ant colonies, fish schooling, bird flocking, bee swarming etc are examples of natural behavior. Multi robot systems, certain programs for tackling optimization and data analysis problems are illustrations of artificial swarm intelligence. The Particle Swarm Optimization (PSO) and Ant Colony Optimization (ACO) are the most fruitful swarm intelligence techniques. PSO algorithm encapsulates a member of the Swarm Intelligence as the individual particle flies through the multidimensional space and adjusts its position in every step with its own experience and that of peers toward an optimum solution by the entire swarm.
PSO algorithm:
In PSO, a particle is nothing but the individual member of the population and the entire population is named as the swarm. A population is randomly initialized and each particle traverses in the searching space in a random manner. It recollects the best previous positions of itself and its neighbors. The good positions are communicated with the particles of the swarm and they adjust their position and velocity in prompt from the best position of all particles. After all particles have moved, the next process begins. In the end, all the particles fly towards better positions over the searching process until the swarm reaches close to the optimum of the fitness function.
The simplicity of the implementation along with the capability to quickly cover a good solution is the reason for the popularity of the PSO method. The algorithm does not need the optimization of the gradient information and employs only the primitive mathematical operators. When compared to the other algorithms, PSO is quick, less expensive and more proficient. Besides, PSO requires only few parameters to be adjusted. All this makes PSO an apt choice to solve the optimization problems. PSO is very apt to find the solutions for the non-linear, non-convex, continuous, discrete, integer variable type problems. There are only three steps involved in a PSO algorithm that rerun again till met with certain stopping conditions. The steps are:
Fitness of each particle is calculated.
Individual global best fitness and positions are updated.

Velocity and position of each particle are updated.
Step 1 and step 2 are much easier. By giving the candidate solution to the objective function, fitness value is calculated. The newly calculated individual and global best fitness are compared with the earlier values and subsequently the best values replace the older ones if needed. The third step contributes towards the optimization ability of the PSO algorithm. The velocity of each particle in the swarm is updated using the following equation:

V(t+1)=V(t)+ C1.R1.(Pb-X(t))+C2.R2.(Pg-X(t)) ---- Eq(3)

X (t+1) =C1.R1. Pb+ C2.R2.Pg ---- Eq(4)

In which, R1 and R2 are either 0 or 1. C1 and C2 are either 0 or 1. Pbis the local best particle value andPg is the global best particle value.X(t) is the tth particle.

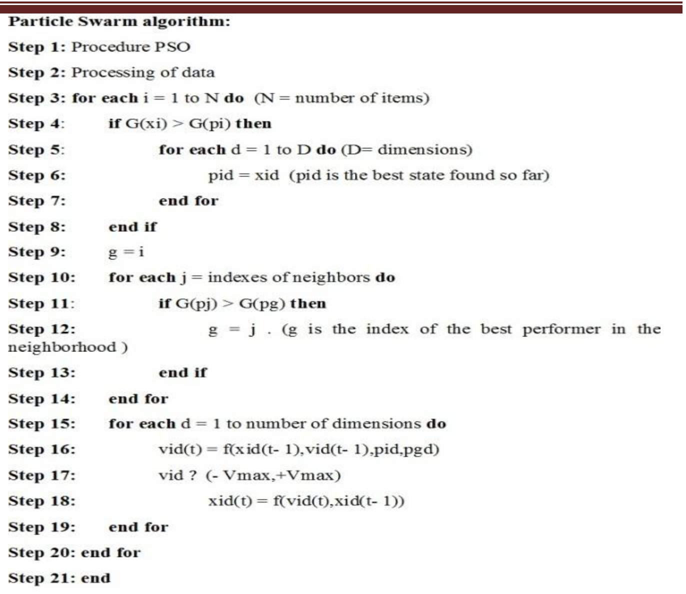
The above step is repeated till met with certain stopping conditions. Certain stopping conditions are - a preset number of iterations of the PSO algorithm, a number of iterations since the last update of the global best candidate solution, or a predefined target fitness value.

Comparison of ‘gbest’ to ‘lbest’

There are two main distinctions between the ‘global best’ PSO and the ‘local best’ PSO. One being that gbest PSO due to its larger particle interconnectivity, its convergence is faster than the lbest PSO. Secondly, the diversity of the lbest PSO is large due to which there are less chances of it being trapped in local minima.

PSO Algorithm Parameters
Certain parameters of the PSO algorithm influence its performance. In general, certain parameters will cast a huge impact on the proficiency of the PSO algorithm and on the other hand, certain parameters may not have any impact. Swarm size or number of particles, number of iterations, velocity components, and acceleration coefficients are some of the basic PSO parameters. Also, inertia weight, velocity clamping, and velocity constriction influence the PSO.
Swarm size

The total number of particles n in a swarm is called as the Swarm size or population size. A larger part of the search space to be covered is created by a big swarm per iteration. This will result in the reduction in the number of iterations that in turn results in a good optimization result as a result of large number of particles. But, at the same time, the computational complexity per iteration increases and also results in more time consumption.
Iteration numbers
To determine the number of iterations to have a good result again is dependent on the problem. If the number of iterations is very less, it might result in the premature stopping of the search process, while if the number of iterations is more, then it will result in the complexity of the computational process and consume more time as well.



```
Particle Swarm algorithm:
Step 1: Procedure PSO
Step 2: Processing of data
Step 3: for each i = 1 to N do  (N = number of items)
Step 4:      if G(xi) > G(pi) then
Step 5:           for each d = 1 to D do (D= dimensions)
Step 6:                pid = xid  (pid is the best state found so far)
Step 7:           end for
Step 8:      end if
Step 9:      g = i
Step 10:     for each j = indexes of neighbors do
Step 11:          if G(pj) > G(pg) then
Step 12:               g = j . (g is the index of the best performer in the
neighborhood )
Step 13:          end if
Step 14:     end for
Step 15:     for each d = 1 to number of dimensions do
Step 16:          vid(t) = f(xid(t- 1),vid(t- 1),pid,pgd)
Step 17:          vid ? (- Vmax,+Vmax)
Step 18:               xid(t) = f(vid(t),xid(t- 1))
Step 19:     end for
Step 20: end for
Step 21: end
```

Fig.1 PSO algorithm

Benefits of the PSO algorithm:

- It is a derivative-free algorithm Because of its ease to implement; it finds its application both in the area of scientific research and engineering problems.
- When compared to other optimization techniques, PSO has few parameters and also the effect of these on the solutions is less.
- It has a very simple easy calculation process as compared to others.
- In this method, less time is required to calculate the optimum value of the problem that also assures the convergence.
- As compared to the other optimization techniques, PSO does not depend much on the initial points.
- It is conceptually very simple.

Drawbacks of the PSO algorithm:

- PSO algorithm suffers from the partial optimism, which degrades the regulation of its speed and direction.
- Problems with non-coordinate system (for instance, in the energy field) exit.

IV PROPOSED SYSTEM

There are three sections in the proposed system. The data are extracted from the database that is converted into binary format in the first step. Then that data is used to find the transaction of a particular product in an easier manner. The second step involves the calculation of the IR value and generation of the candidate set on the basis of the IR value. Then the hash tree will be generated using the candidate set in the third step. Using the binary values, the support value will be calculated. Nodes that do not satisfy the minimum support property would be erased. Finally, the association rules with high support values are mined from the reduced Hash tree.

PHASE I:

Data processing:

The necessary data are condensed from the existing database. Information like customer id, product id that are associated with the purchase by a customer and the time id are condensed from large databases.

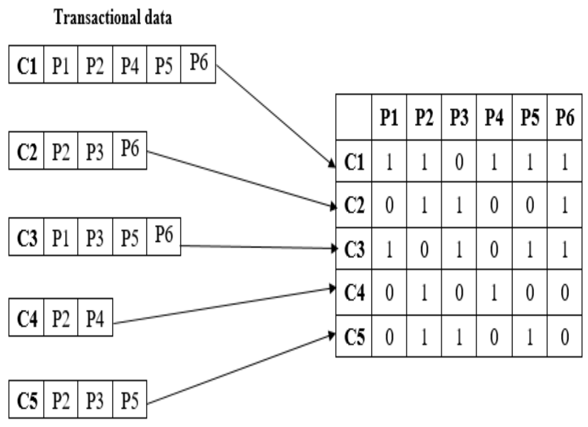
Binary transformation:

In this step, the data from the first step is transformed into binary format i.e. 0 or 1 for storage purposes. This in turn hastens the database scanning operation and results in quick calculation of the support value. The following figure depicts the binary conversion process.Let us consider that there are five customers, namely, C1 to C5 in the transactional database shown in figure 2. All the transactions are transformed into binary form and stored.In the above example, as there are six different products, hence six columns will exist. Consider C3, as he / she has purchased the products P1, P3, P5 and P6. Therefore, for C3, the rows under P1, P3, P5 and P6 will have the value "1" and P2 and P4 as "0".

PHASE II:

IR Calculation:

To produce association rules that are even more meaningful IR value has to be calculated.



	P1	P2	P3	P4	P5	P6
C1	1	1	0	1	1	1
C2	0	1	1	0	0	1
C3	1	0	1	0	1	1
C4	0	1	0	1	0	0
C5	0	1	1	0	1	0

Fig.2 binary conversion

Besides, by calculating the IR value to determine the rule length, the search efficiency is improved. By doing so, association rules that are meaningless are circumvented. According to the following function, the IR value after the binary conversion is determined using the given equation 5:

IR=[log(x*Num_trans(x))+log(y*Num_trans(y))][Num_trans(x,y)/Total_trans)] -----Eq(5)

In which, x is the length of the item set and Num_trans(x) is the number of transactions that contain x products. 'y' is the length of the item set and Num_trans(y) is the number of transactions containing y products. Num_trans(x,y) is the number of transactions purchasing x to y products. Total_trans denotes the number of total transaction in the transactional database.

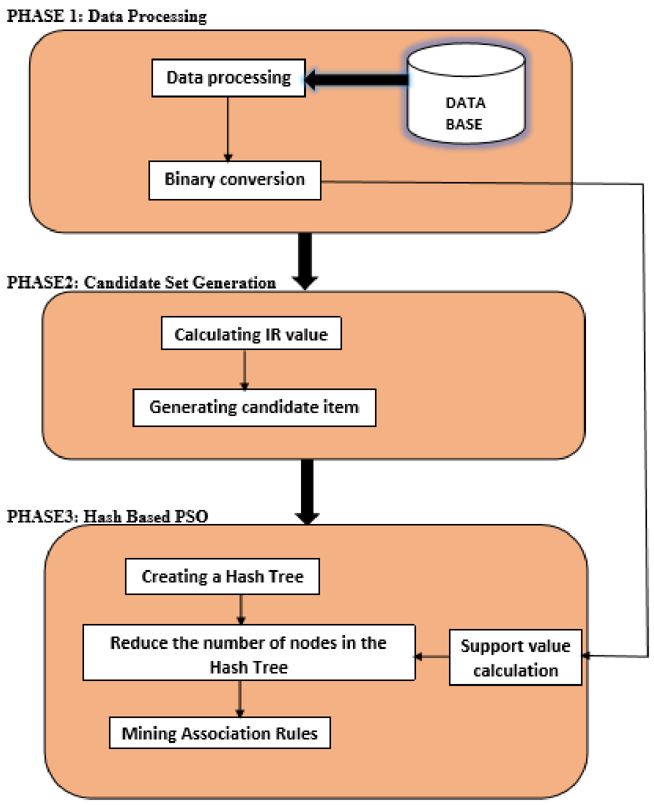


Fig.3 Proposed System

Generating candidate item set:

Candidate set is generated based on the IR value. For instance, if IR = 3.278, then the length of the item set is 3, i.e. each item set has 3 products. Therefore, a candidate set like {(1,2,3),(1,3,4),(1,4,6),(2,3,4),(2,5,6),(3,5,6)…..} is created. Based on this, a hash tree has to be generated.

PHASE III:

Creating a hash tree:

Employing the candidate set, one has to create the hash tree using the following hash function:

Hash (key) =product_key%N -----Eq(6)

In which, N is the length of the item set.If N=3, then the possibility of remainder is 0, 1, 2. So each node in the tree has 3 children node with the flag values of 0, 1,2. None of the leaf node contains flag values. Leaf nodes will have only the item set. If N=4, then the possibility of children node is 4 with the flag values 0,1,2,3. Consider the following candidate set.
{(9,3,6),(1,4,6),(11,31,47),(6,8,12),(6,12,17),(11,15,41),(11,15,34)…..}
The first item set is (9,3,6). Now we have to apply the hash function Hash(key) on each item set.
Hash (9)=9%3=0;
Hash (3)=3%3=0;
Hash (6) =6%3=0;
Since N=3, the possibility of children node is 0, 1, 2. Initially, the head of the hash contains the children with flag values 0, 1, 2.

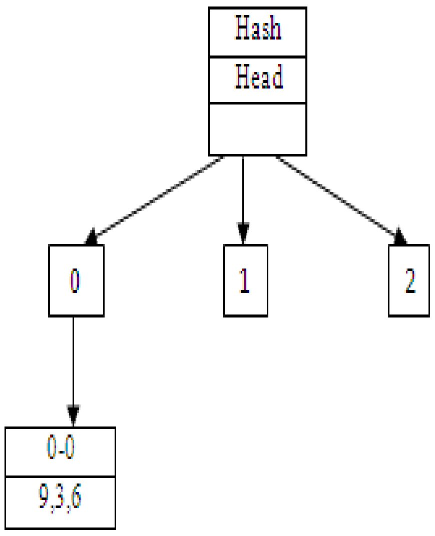


Fig.4Hash tree for (9, 3, 6)

Similarly, the hash tree will be created for the candidate set. A sample hash tree is depicted in figure 5.

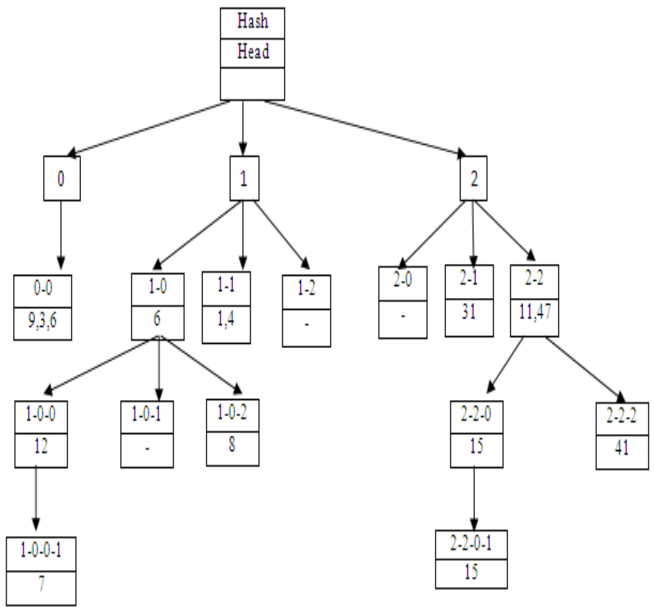
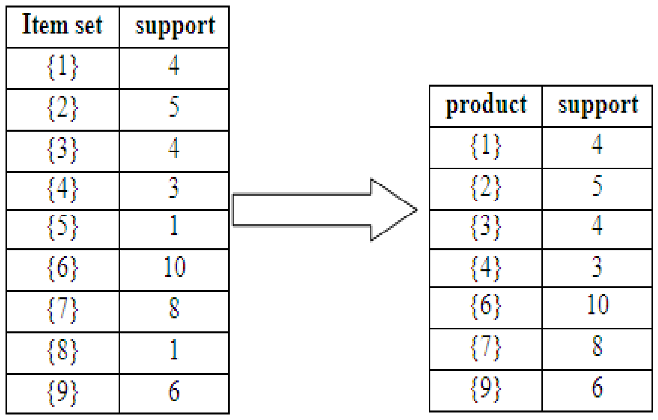


Fig.5 Sample hash tree

Reducing the nodes in hash tree:
Now the support value will be calculated for initial association rule k (u=>v) using the following equation 3

Support (u=>v) =Num_trans(u,v)/Total_trans. -----Eq(7)

The item set with least value will be erased from the hash tree. The process continues till the reach of the threshold value. The support value for each product is determined. Here the rule is A=>A. Let us consider the following products with their corresponding support values. From the hash tree, the products having the least support value would be erased. In this instance, the least value is 1. Hence, the product having 1 as the support value will be eliminated from the hash tree.

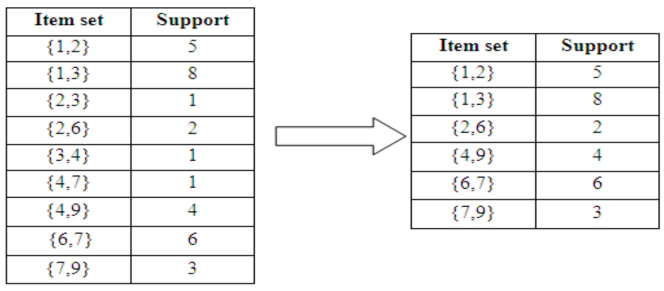


Item set	support
{1}	4
{2}	5
{3}	4
{4}	3
{5}	1
{6}	10
{7}	8
{8}	1
{9}	6

product	support
{1}	4
{2}	5
{3}	4
{4}	3
{6}	10
{7}	8
{9}	6

Fig.6 One dimensional

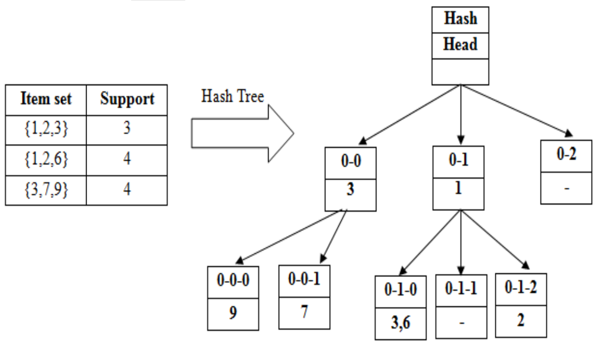
Next the item set will be moved to the two, three dimensions and depending on the support value the hash tree will be further reduced which is shown in figure 8.



Item set	Support
{1,2}	5
{1,3}	8
{2,3}	1
{2,6}	2
{3,4}	1
{4,7}	1
{4,9}	4
{6,7}	6
{7,9}	3

Item set	Support
{1,2}	5
{1,3}	8
{2,6}	2
{4,9}	4
{6,7}	6
{7,9}	3

Fig.7 Two dimensional



Item set	Support
{1,2,3}	3
{1,2,6}	4
{3,7,9}	4

Fig.8 Three dimensional

Mining Association rules:
On the basis of the minimum support value, the association rule will be created after erasing the nodes from the hash tree. From the hash tree, those item sets that do not have the minimum threshold value would be erased. If the support of the rule E=>F and E=>D is high, then we can generate a rule with the measures which is E=> {D, F}. Likewise, the rules can be generated from the reduced hash tree. From the above final hash tree we can create the following rules.

{3}=>{9} and {3}=>{7} , so the rule is {3}=>{9,7};
{1}=>{3,6} and {1}=>{2}, so the rule is {1}=>{3,6,2};

The possibility of buying the product set {3,9,7} and {1,3,6,2} is high. The hash tree omits the useless comparison of transaction from the beginning. It simplifies the generation of association rules.

V RESULT ANALYSIS

Sample input/output:
Input:

Products=>	p1= Bread
	P2= Milk
	P3= Jam
	P4= Cheese
	P5= Nuts
	P6= Butter

Transactional history of 6 customers:
C1= {p1, p2, p4}
C2= {p3, p1, p6}
C3= {p1, p2, p3, p4, p5, p6}
C4= {p2, p4, p6}
C5= {p1, p2, p3}
C6= {p1, p2, p3, p6}

Binary conversion:

	P1	P2	P3	P4	P5	P6
C1	**1**	**1**	**0**	**1**	**0**	**0**
C2	**1**	**0**	**1**	**0**	**0**	**1**
C3	**1**	**1**	**1**	**1**	**1**	**1**
C4	**0**	**1**	**0**	**1**	**0**	**1**
C5	**1**	**1**	**1**	**0**	**0**	**0**
C6	**1**	**1**	**1**	**0**	**0**	**1**

IR value Calculation:

IR=[log(x*Num_trans(x))+log(y*Num_trans(y))]
[Num_trans(x,y)/Total_trans)]

{p1, p2} = [log (1*5) + log (1*5)] * 4/6= 0.932
{p1, p3} = [log 5 + log 4] * 4/6 = 0.867
{p1, p4} = [log 5 + log 3] * 2/6 = 0.392
{p1, p5} = [log 5 + log 1] * 1/6 = 0.116
{p1, p6} = [log 5 + log 4] * 3/6 = 0.651
{p2, p3}= [log 5 + log 4] * 3/6 = 0.651
{p2, p4}= [log 5 + log 3] * 3/6 = 0.588
{p2, p5}= [log 5 + log 1] * 1/6 = 0.116
{p2, p6}= [log 5 + log 4] * 3/6 = 0.651
{p3, p4}= [log 4 + log 3] * 1/6 = 0.179
{p3, p5}= [log 4 + log 1] * 1/6 = 0.100
{p3, p6}= [log 4 + log 4] * 3/6 = 0.602
{p4, p5}= [log 3 + log 1] * 1/6 = 0.079
{p4, p6}= [log 3 + log 4] * 2/6 = 0.3597
{p5, p6}= [log 1 + log 4] * 1/6 = 0.100

IR value will be round off and the maximum value is 1. So the length is 1 for next IR value. Item sets which are having IR value 1 are given below.

{p1, p2}, {p1, p3},{p1, p6},{p2, p3},{p2, p4},{p2, p6},{p3, p6}
The IR value for length 1 is calculated as follows:

{p1, p2}=> p3= [log (2 * 4) + log (1 * 4)] * 3/6= 0.753
{p1, p2}=> p4= [log 8 + log 3] * 2/6= 0.460
{p1, p2}=> p5= [log 8 + log 1] * 1/6= 0.151
{p1, p2}=> p6= [log 8 + log 4] * 2/6= 0.502
{p1, p3}=> p4= [log 8 + log 3] * 1/6= 0.230
{p1, p3}=> p5= [log 8 + log 1] * 1/6= 0.151
{p1, p3}=> p6= [log 8 + log 4] * 3/6= 0.753
{p1, p6}=> p4= [log 6 + log 3] * 1/6= 0.209
{p1, p6}=> p5= [log 6 + log 1] * 1/6= 0.129
{p2, p3}=> p4= [log 6 + log 3] * 1/6= 0.290
{p2, p3}=> p5= [log 6 + log 1] * 1/6= 0.129
{p2, p3}=> p6= [log 6 + log 4] * 2/6= 0.460
{p2, p4}=> p5= [log 6 + log 1] * 1/6= 0.129
{p2, p4}=> p6= [log 6 + log 4] * 2/6= 0.460
{p2, p6}=> p5= [log 6 + log 1] * 1/6= 0.129
{p3, p6}=> p4= [log 6 + log 3] * 1/6= 0.290
{p3, p6}=> p5= [log 6 + log 1] * 1/6= 0.129

Generation of candidate item set:
From the above IR values top most value will be taken for forming the candidate item set. Here we are taking the values 0.753, 0.502, 0.460 and 0.290. The formed candidate set is given below:

{(p1, p2, p3), (p1, p2, p4), (p1, p2, p6), (p1, p3, p6), (p2, p3, p4), (p2, p3, p6), (p2, p4, p6), (p3, p6, p4)}

Hash Tree Creation:
Find the hash value of each product using the hash function,

Hash (product) = product_num % size of item set

Hash (p1) = 1% 3 = 1
Hash (p2) = 2% 3 = 2
Hash (p3) = 3% 3 = 0
Hash (p4) = 4% 3 = 1
Hash (p5) = 5% 3 = 2
Hash (p6) = 6% 3 = 0

The hash tree contains maximum of three leaves 0, 1, 2 for each parent. For the candidate item set, hash tree is created based on the hash value. Hash is shown in the figure 9.

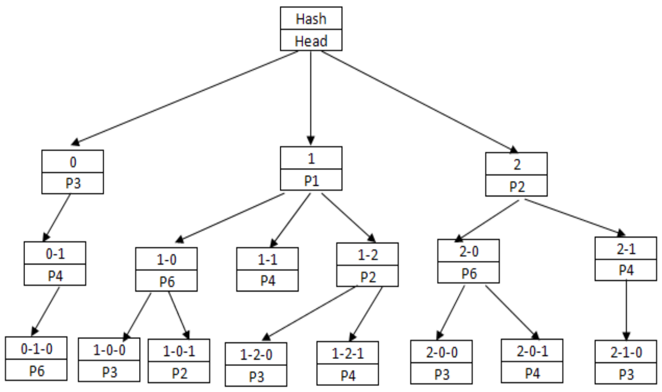


Fig.9 Hash tree for the generated candidate set

Support value Calculation:

Support (u=>v) =Num_trans(u,v)/Total_trans

Support (p1, p2, p3) = 3/6 = 0.5
Support (p1, p2, p4) = 2/6 =0.33
Support (p1, p2, p6) = 2/6 =0.33
Support (p1, p3, p6) = 3/6 = 0.5
Support (p2, p3, p4) = 1/6 =0.16
Support (p2, p3, p6) = 2/6 =0.33
Support (p2, p4, p6) = 2/6 =0.33
Support (p3, p6, p4) = 1/6 =0.16

Reducing the Hash tree:
Item set with minimum support value is removed from the hash tree. Here {p2, p3, p4} and {p3, p6, p4} are removed and the final hash tree is shown in figure 10.

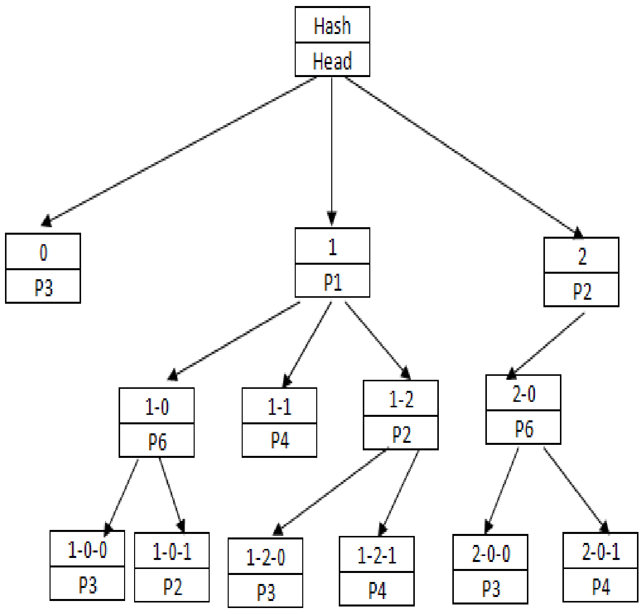


Fig.10 Reduced Hash tree

Mining Association rules:
From the reduced hash tree, possibility of buying the following item sets is high.

{(p1, p2, p3), (p1, p2, p4), (p1, p2, p6), (p1, p3, p6), (p2, p3, p6), (p2, p4, p6)}

Rules for the item sets:
{Bread, Jam} => {Butter}
{Bread, Milk} => {Jam}
{Bread, Milk} => {Cheese}
{Bread, Milk} => {Butter}
{Milk, Jam} => {Butter}
{Milk, Nuts} => {Butter}

Bread, Milk} => {Jam, Cheese, Butter}

DB size Vs Time - Hash tree generation:
The proposed system consumes some time to create the hash tree. The time taken for generation of the hash tree is directly proportional to the size of the database. Here, the instance of food mart 2015 is considered. There are 20 items in every food mart 2015record. A total of 40 repetitions were conducted to get the final results as the results differ for each repetition. The database size determines the time taken for each transaction to create a hash.

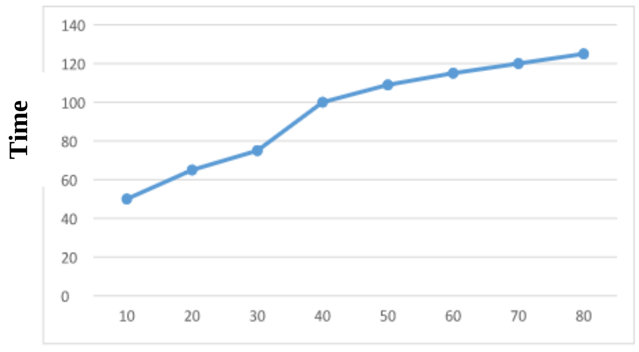
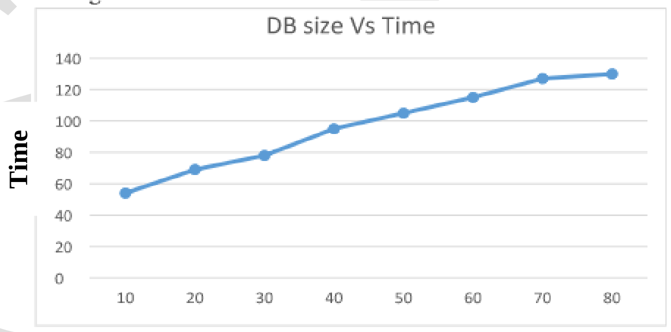


Fig.11Gra Database size ıeration

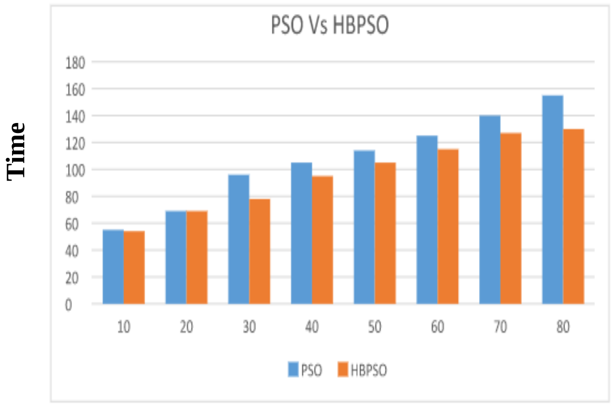
DB size Vs Time – Mining Association rules:

The association rules will be extracted from the tree, once the hash tree is ready. The rule size governs the time taken for extracting the rule. As the rule is directly generated from the hash tree, HBPSO took lesser time as compared to the PSO. The following graph depicts the total time (tree creation + mining association rule) required for mining the rules for different sizes of databases.



Database size
Fig.12 Graph of Association rule generation

PSO Vs HBPSO:



Database size
Fig.13 Comparison of HBPSO & PSO

HBPSO as compared to the already existing PSO algorithms performs better as it consumes lesser time although the difference is negligible. For the same database, the following graph depicts the comparison between HBPSO and PSO.

VI CONCLUSION

In the current paper, the mining of the association rules for items in a large database was studied. A new method was proposed to find the candidate item set and based on that to create a hash tree. It was given the name of Hash Based PSO as it uses the common hash tree structure for all the concurrently executed queries. Based on the experiments, it was found that the HBPSO is more proficient than the previously proposed method. It requires a smaller amount of memory and scales better with respect to the number of queries. The time needed is less as it overlooks the velocity and fitness value that consume much time to be calculated. Hence, the complexity and the cost is lessened. To precisely evaluate the performance of the proposed method, wide spread simulation study has been done.

VII REFERENCE

[1] R.J. Kuoa,∗, C.M. Chao b, Y.T. Chiuc "Application of particle swarm optimization to association rule mining", Elsevier Applied Soft Computing 11 (2011) 326–336.
[2] Saurav Mallik, Anirban Mukhopadhyay, "RANWAR: Rank-Based Weighted Association Rule
Mining from Gene Expression and Methylation Data", IEEE Transactions on NanoBioscience.
[3]Azadeh Soltani and M.-R. Akbarzadeh-T.,"Confabulation-Inspired Association Rule Miningfor Rare and Frequent Itemsets" IEEE Transactions On Neural Networks And Learning Systems.
[4]Tarinder Singh; Manoj Sethi, "Sandwich-Apriori: A combine approach of Apriori and Reverse-Apriori",2015 Annual IEEE India Conference (INDICON) Year: 2015 Pages: 1 - 4.
[5] APRIORI Algorithm "www3.cs.stonybrook.edu/~cse634/lecture_notes/07apriori.pdf".
[6]Rakesh Agrawal and amakrishnan Srikant, "Fast Algorithms for Mining Association Rules", IBM Almaden Research Center.
[7]F. Bodon. A Fast Apriori Implementation. Proc. 1st IEEE ICDM Workshop on Frequent Item Set Mining Implementations (FIMI 2003, Melbourne, FL). CEUR Workshop Proceedings 90, Aachen, Germany 2003. http://www.ceur-ws.org/Vol-90/
[8]C. Borgelt. Efficient Implementations of Apriori and Eclat. Proc. 1st IEEE ICDM Workshop on Frequent Item Set Mining Implementations (FIMI 2003, Melbourne, FL). CEUR Workshop Proceedings 90, Aachen, Germany 2003.http://www.ceur-ws.org/Vol-90/
[9]Li Min; Wang Chunyan; Yan Yuguang,"The Research of FP-Growth Method Based on Apriori Algorithm in MDSS", Digital Manufacturing and Automation (ICDMA), 2010 International Conference on Year: 2010, Volume: 2 Pages: 770 - 773.
[10]Christian Borgelt Otto-von-Guericke-University of Magdeburg, Magdeburg, "An implementation of the FP-growth algorithm", Proceeding OSDM '05 Proceedings of the 1st international workshop on open source data mining: frequent pattern mining implementations Pages 1-5.
[11]Wei Zhang ; Sch. of Software, Yunnan Univ., Kunming, China ; Hongzhi Liao ; Na Zhao, "Research on the FP Growth Algorithm about Association Rule Mining", Business and Information Management, 2008. ISBIM '08. International Seminar on (Volume:1), Pages:315 - 318.
[12]Jiawei Han,Jian Pei and Yiwen Yin, "Mining frequent patterns without candidate generation",ProceedingSIGMOD '00 Proceedings of the 2000 ACM SIGMOD international conference on Management of data Pages 1-12.
[13]Supriya Singh, "DFP-Growth: An Efficient Algorithm for Frequent Patterns in Dynamic Data Mining", International Journal of Application or Innovation in Engineering & Management (IJAIEM),Volume 3, Issue 4, April 2014.
[14]Wei Song, Wenbo Liu, Jinhong Li, "Dynamic FP-Tree Pruning for Concurrent Frequent Itemsets Mining", Computational Intelligence and Intelligent Systems Volume 316 of the series Communications in Computer and Information Science pp 111-120.
[15]Keshav Lodhi, Dr. R. C. Jain, Prof. Deepak Sain, "Dynamic Fp-growth Tree Mining Approach with Projection Technique", International Journal of Engineering Research & Technology, Vol. 3 - Issue 1 (January - 2014).
[16]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.
[17]Bilal Alataş , Erhan Akin, "An efficient genetic algorithm for automated mining of both positive and negative quantitative association rules", Soft Computing February 2006, Volume 10, Issue 3, pp 230-237.
[18]H. Toivonen, “Samplin, Large Databases for Association Rules”, VLDB, India, (1996) September 3-6.
[19]Bilal Alatas,Erhan Akin,Ali Karci, "MODENAR: Multi-objective differential evolution algorithm for mining numeric association rules", Applied Soft Computing Volume 8, Issue 1, January 2008, Pages 646–656.
[20]Suh-Ying Wur,Yungho Leu "An Effective Boolean Algorithm for Mining Association Rules in Large Databases", Proceeding DASFAA '99 Proceedings of the Sixth International Conference on Database Systems for Advanced Applications Pages 179-186.
[21]Mahmoud Mostafa El-Sherbiny,"Particle swarm inspired optimization algorithm without velocity equation" Egyptian Informatics Journal,31 March 2011.
[22]Kennedy J, Eberhart RC. Particle swarm optimization. In: Proc. IEEE international conference on neural networks (Perth, Australia), vol. IV. IEEE Service Center: Piscataway, NJ; 1995. p.1942–8.
[23] Kennedy J. The particle swarm: social adaptation of knowledge. In: Proc. IEEE international conference on evolutionary computation (Indianapolis, Indiana). IEEE Service Center: Piscataway,NJ; 1997, p. 303–8.
[24] Liu Bo, Wang Ling, Jin Yi-Hui, Tang Fang, Huang De-Xian. Directing orbits of chaotic systems by particle swarm optimization. Chaos, Solitons Fractals 2006;29:454–61.
[25] Tasgetiren M, Liang Yun-Chia, Sevkli Mehmet, Gencyilmaz Gunes. A particle swarm optimization algorithm for makespan and total flowtime minimization in the permutation flowshop sequencing problem. Eur J Oper Res 2007;1930–47.
[26] Wang Hui, Liu Yong, Wu Zhijian, Sun Hui, Zeng Sanyou, Kang Lishan. An improved Particle Swarm Optimization with adaptive jumps. IEEE World Congress Comput Intell 2008:392–7.
[27]Liu Jianhua, Fan Xiaoping, Qu Zhihua. An improved particle swarm optimization with mutation based on similarity. Third Int Conf Nat Comput 2007:824–8.
[28]Boeringer Daniel W, Werner Douglas H. Particle swarm optimizationversus genetic algorithms for phased array synthesis. IEEETrans Antenna Propagation 2004;52(3):771–9.
[29] Clerc M. The Swarm And The Queen: Towards A Deterministic And Adaptive Particle Swarm Optimization. Proc ICEC, Washington,DC 1999:1951–7.
[30]] Tasgetiren M, Liang Yun-Chia, Sevkli Mehmet, Gencyilmaz Gunes. A particle swarm optimization algorithm for makespan and total flowtime minimization in the permutation flowshop sequencing problem. Eur J Oper Res 2007;1930