Frequent Itemset Mining in Large Databases by using Artificial Bee Colony (ABC) Algorithm

Ch.Sreenu babu, Mahendra M, Ch. Prathima

Assst. Professor, Dept of IT,

Big data & Analytics Research Center, Sreevidyanikethan Engineering College,Tirupathi

Abstract

Artificial Bee Colony (ABC) algorithm is one of the most recently introduced swarm-based algorithms. ABC simulates the intelligent foraging behaviour of a honeybee swarm. In our proposed frequent mining technique, the input database is portioned by FPL (Frequent Pattern List). By exploiting the FPL sub databases the frequent itemsets are mined by Artificial Bee Colony (ABC) algorithm. Thus, the proposed technique successfully mines the frequent itemsets via FPL and ABC techniques. The proposed technique will be compared with the existing frequent itemsets mining techniques and ABC performance is compared with the conventional ABC in terms of benchmark functions.

Keywords: Frequent itemsets, Frequent Pattern List (FPL), Itemset Mining, Artificial Bee Colony (ABC),

Introduction

Data mining has recently attracted considerable attention from database practitioners and researchers because it has been applied to many fields such as market strategy, financial forecasts and decision support [5]. Many algorithms have been proposed to obtain useful and invaluable information from huge databases. Association rule mining has many important applications in our life. An association rule is of the form X => Y. And each rule has two measurements: support and confidence [6]. The association rule mining problem is to find rules that satisfy user specified minimum support and minimum confidence. It mainly includes two steps: first, find all frequent patterns; second, generate association rules through frequent patterns [4] .Frequent pattern mining plays an essential role[10] in many data mining tasks and applications, such as mining association rules, correlations, sequential patterns, episodes, multi-dimensional patterns, max-patterns and frequent closed patterns, partial periodicity, emerging patterns, classification and clustering [1,9]

Mining frequent patterns in transaction databases has been studied popularly in data mining research. Most of the algorithms used today typically employ sophisticated in-memory data structures, where the data is stored into and retrieved from flat files. The integration of data mining with database systems is an emergent trend in database research and development area. This is particularly driven by explosion of the data amount stored in databases such as Data Warehouses during recent years, and database systems provide powerful mechanisms for accessing, filtering, and indexing data, as well as SQL parallelization [2,11] .Frequent pattern mining often generates a very large number of frequent patterns and rules, which reduces not only the efficiency but also the effectiveness of mining since users have to shift through a large number of mined rules to find useful ones [3,12].

The candidate generation-and-test methodology, called the Apriori technique, was the first technique to compute frequent patterns (FP), based on the Apriori principle (i.e., the anti-monotone property)[16]. However, the requirements of multiple database scans and a large amount of memory for handling the candidate patterns restrict the efficient use of the

Apriori- based algorithms [17]. Introduction of tree structure-based frequent pattern mining effectively gets over the problems of Apriori algorithms. The FP-growth mining technique proposed by Han et al. has been found to be an efficient algorithm when using a tree structure called FP-tree [17-20]. However, the construction of such an FP-tree requires two database scans and prior knowledge about the support threshold. Researchers have generally focused on the frequent pattern mining, as it is complex and the search space needed for finding all frequent itemsets is huge[15]. A number of efficient algorithms have been proposed in the last few years to make this search fast and accurate [14].

Proposed Frequent Itemset Mining Technique

Here, we have proposed a frequent itemset mining technique by using the FPL and ABC algorithm. To mine the frequent itemset, the input database is divided into several sub databases by the FPL. The each item node of the FPL can be regarded as a sub-database of the original whole database. By using the database partitioned results from the FPL, the most frequent itemsets are mined by the ABC algorithm. The mined frequent itemsets result from the ABC algorithm produce most frequent itemsets of the whole database. Our proposed method comprised of two phases namely, (i) Frequent Pattern List and (ii) Frequent Itemset mining by ABC.

Frequent Pattern List (FPL)

In our proposed method an FPL is used to the partitioned the database into several sub database. One important property is the database-partitioning property. That is, each item node of the FPL can be regarded as a sub-database of the original whole database. In this way, FPL can partition the database into a set of sub-databases.

Transaction Signatures

The FPL is a linear list of item nodes, with each item node corresponding to one frequent item in the database. The transactions in the databases are transformed into bit strings called the transaction signatures, with 1-bits representing the existence of frequent items in the transactions, and 0-bits representing the absence of frequent items. The transaction signatures are then assigned to the item nodes according to the positions of their last 1-bits.

Before the FPL construction process the given input database D, with the size of $M \times N$ is once scanned with minimum support threshold value α. From the result of this one time scanning process, we can obtain the database D' which contains only frequent items that support values are not less than the minimum support threshold value α. By using the database D' the FPL is constructed by following the steps which are given below,

FPL Construction

Step 1: Scan the database D'

Step 2: For each transaction T in D', frequent items f are selected.

Step 3: Selected frequent items f are sorted in descending order.

Step 4: Afterwards, the transaction signature for each item node is built from MSB to LSB.

Step 5: At each item node, if its corresponding item is contained in T, set the bit to 1 otherwise set the bit to 0.

That is, the bits in the transaction signature indicate the occurrence or absence of the frequent items in T . We keep the bits from MSB to the last (rightmost) 1 bit, and trim all the trailing 0 bits. The transaction signature for T is then stored in the item node of the FPL that corresponds to the last (rightmost) 1 bit of the transaction signature.

Hierarchical Partitioning

The hierarchical partitioning in FPL is performed when the input database become larger and larger. The hierarchical partitioning is performed on the global FPL last sub database i.e. last item node. To mining frequent itemsets the last sub database is put into the memory. Therefore, we do not have to put the FPL into memory as a whole; rather, each time we put just one item node into memory.

The last sub database fit into the memory means we can't spilt the database in second level or otherwise the (first-level) last sub-database still outfits the memory, we again regard the last item node of this (first-level) local FPL as a (second-level) sub-database and construct its corresponding (second-level) local FPL. Again only the last item node of this second-level local FPL has to be put into memory. This process can be continued until the current local FPL can fit in memory. During the FRL hierarchical partitioning three basic operations are performed on the sub databases namely,

Bit Counting: It accumulates the number of 1-bits in the transaction signatures to count the supports of the items.

Signature Trimming: It trims the last 1-bits of the transaction signatures and then trims the trailing 0-bits.

Signature Migration: It moves the trimmed signatures from the last item node to other nodes according to the positions of the last 1-bits in the trimmed signatures

After the hierarchical partitioning the frequent itemsets are mined from the sub databases by exploiting the ABC algorithm which is explained in the following section.

Frequent Itemset Mining

In frequent itemset mining the database D' sub databases SD' are taken and find the most frequent itemsets by using the artificial bee colony algorithm (ABC).

Artificial Bee Colony algorithm (ABC)

In ABC algorithm there are four phases are performed to find the optimal solutions. In the ABC algorithm, each food source represents a possible solution to the problem under consideration, and the nectar amount of a food source represents the quality of the solution. The ABC algorithm assumes that there is only one employed bee for every food source, i.e., the number of food sources is same as the number of employed bees. The employed bee of an abandoned food source becomes a scout bee, and as soon as it finds a new food source, it becomes an employed bee again. The ABC algorithm is an iterative algorithm. It starts by associating all employed bees with randomly generated food sources (solution). Then, every employed bee moves to a new food source in the neighborhood of its currently associated food source and evaluates its nectar amount (objective value) during iterations. When the employed bees complete the process, they share the nectar information of the food sources with the onlooker bees, and the number of onlooker bees to be sent to the food source found by the employed bee is proportional to the nectar amount of that food source.

Here, we have introduced an ABC algorithm by making the adaptiveness in their food source selection in the onlooker bee phase. In ABC algorithm, food sources are selected in onlooker bee phase by accomplishing the fractional velocity operation. This food source selection operation by fractional velocity is incorporated into the standard ABC algorithm to make the adaptiveness as well as to improve the optimization performance. The procedure that followed in the ABC algorithm is given below,

Initialization phase

Step 1: Initialize parameters, including the number of food sources or the number of employed bees, number of onlooker bees and number of scout bees. Here we generate the food sources which have the different combinations items and their corresponding support S_k and confidence C_k values. The generated food source as,

$$F_k = ((a,b),(c,d),\cdots(i,j)) \quad (1)$$

In Equ. (2), (i, j) is the index value of the items, $F_k, where\, 1 \le k \le N_p$, N_p is the population size.

Step 2: Calculate the objective value of each food source and then determine the best food resource. The objective value is computed by,

$$O_k = C_k \times \log(S_k \times (bl) + 1) \quad (2)$$

In Equ. (2), bl is the length of best food source.

Employed bee phase

Step 3: For every employed bee, generate a new food source.

$$z_{n,k} = y_{n,k} + \phi_{n,k}\left(y_{n,k} - y_{l,k}\right) \quad (3)$$

Where, l and k is a random selected index, ϕ is randomly produced number in the range [-1, 1] and $z_{n,k}$ is the new value of the k^{th} position.

Step 4: Calculate the objective value for every new food sources and find the best food source.

Onlooker bee phase

Step 5: In onlooker bee phase, from the fourth iteration the food source is selected by using the formula,

$$z_{t+1} = \alpha z_t + \frac{1}{2}\alpha z_{t-1} + \frac{1}{6}\alpha(1-\alpha)z_{t-2} + \frac{1}{24}\alpha(1-\alpha)(2-\alpha)z_{t-3} + \varphi_{n,k}(y_{n,k} - y_{l,k}) \quad (4)$$

In Equ. (4), $y_{n,k}$ - old position

$y_{l,k}$ - new position

Step 6: Calculate the number of onlooker bees to be sent to the food source.

Step 7: For every onlooker bee, generate the new food sources using equation (3).

Scout bee phase

Step 8: Initialize scout bees with random solutions and update the best food sources.

Step 9: Determine the worst employed bees and replace them with the scout bees if the scout bees are better.

Step 10: If a stopping criterion is met, then the best food source is obtained with its objective value.

Among the food sources obtained from these three phases, select one best food source having minimum fitness value, which is denoted as B and the remaining food sources are given to the next iteration. This process is repeated until it reaches the maximum number of iterations I . The obtained best food sources are the most frequent itemsets.

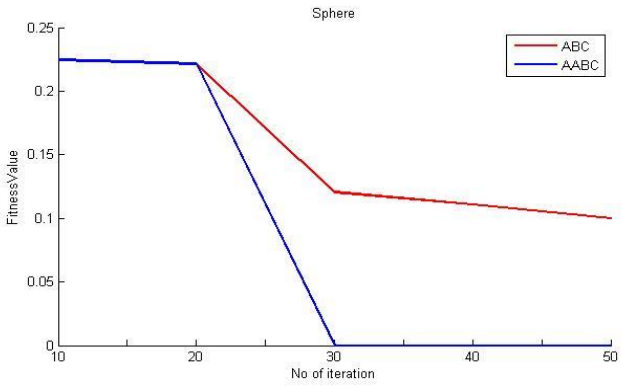
Experimental Results and Discussion

Data Set: The data are gathered over three non-consecutive periods.. The total number of receipts being collected equals 88,163. Totally, 5,133 customers have bought at least one product in the shop during the data collection period. Each record in the dataset contains details regarding the date of purchase, which is represented as variable 'date', the receipt number as 'receipt nr', the article number as 'article nr', the number of items purchased as 'amount', the article price in as 'price', and the customer number as 'customer nr'. But, we have not considered all this information in our data stream, instead of we used only the transaction ID and the name of the items to be purchased.

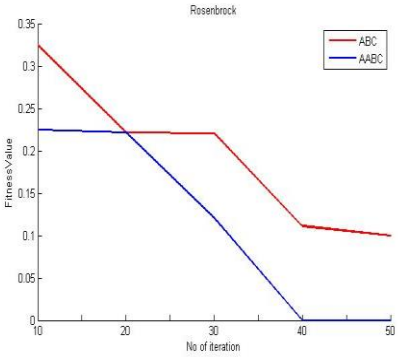
In our proposed technique, the frequent itemsets were mined by FPL and ABC algorithm. Initially the dataset frequent itemsets are mined and that database is divided into several sub databases based on the memory storage size by the FPL. The FPL is constructed for every sub database. Afterward, the most frequent itemsets were mined from the sub databases by using the ABC algorithm.

To accomplish the performance analysis process, we have performed the 10 rounds of experiments by ABC and FP methods. Moreover, our proposed ABC method performance is evaluated by using the standard functions like sphere and rosenbrock. These standard functions are computed by the Equations which is described as,

$$f_0(x) = \sum_{i=1}^{n} x_i^2 \qquad (5)$$



(i)

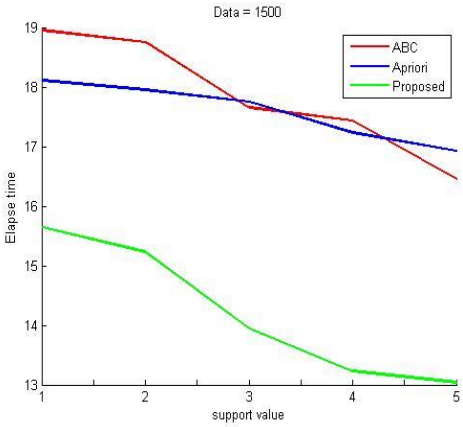


(ii)

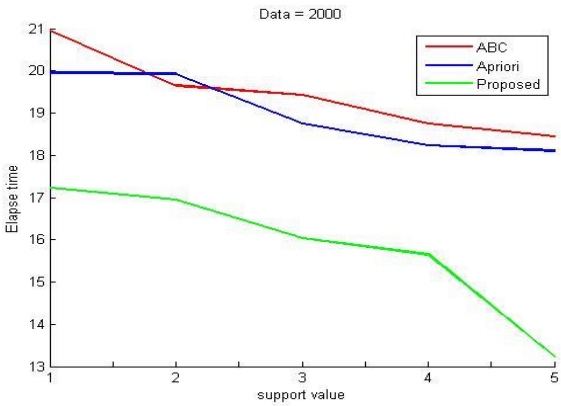
Figure 2: Performance of FP and ABC methods with (i) sphere and (ii) rosenbrock

As can be seen from Fig. 2, our proposed ABC method has obtained the more accurate food sources which having minimum fitness value than the ABC method. In Fig. 2(i) and (ii) shows that our proposed ABC method has obtain accurate fitness values in the two standard functions. The high performance result shows that our ABC method can able to find the more accurate frequent itemsets.

Moreover, our proposed FPL-ABC based frequent itemset mining technique performance is compared with the existing mining techniques. The results of our proposed and existing mining algorithm results in terms of run time are given in Fig. 3.



(i)



(ii)

Figure 3: Performance of our proposed FPL-ABC and existing frequent itemset mining technique results by varying dataset size (i) 1500 and (ii) 2000

As can be seen from Fig. 3, our proposed FPL-ABC technique mines the frequent itemsets from the large database with minimum run time than the existing apriori frequent itemset mining and ABC technique. When increasing the dataset size the run time of our proposed FPL-ABC is increased but this high level not higher than the existing techniques. The database partitioning by the FPL method speed up the process and also mining result based on the

Conclusion

In this paper, a frequent itemset mining technique with FPL and ABC algorithms was proposed to find the high frequent itemsets from the database. In this proposed methodology, the large database is portioned into sub databases which are based on the memory storage size. Afterward, the frequent itemsets are mined from the every sub database by exploiting the ABC algorithm. Our proposed frequent itemset mining technique has successfully mine the frequent itemsets by the FPL-ABC algorithm. All these processes have improved the performance of the proposed frequent itemset mining technique. The frequent itemset mining results have shown that the proposed technique with FPL-ABC has achieved high performance results with low run time. Thus, our proposed frequent itemset mining technique has offered better performance in mine the frequent itemsets from the large database.

In this paper, a frequent itemset mining technique with FPL and ABC algorithms was proposed to find the high frequent itemsets from the database. In this proposed methodology, the large database is portioned into sub databases which are based on the memory storage size. Afterward, the frequent itemsets are mined from the every sub database by exploiting the ABC algorithm. Our proposed frequent itemset mining technique has successfully mine the frequent itemsets by the FPL-ABC algorithm. All these processes have improved the performance of the proposed frequent itemset mining technique. The frequent itemset mining results have shown that the proposed technique with FPL-ABC has achieved high performance results with low run time. Thus, our proposed frequent itemset mining technique has offered better performance in mine the frequent itemsets from the large database.

References

1. Thabet Slimani and Amor Lazzez, "Efficient Analysis of Pattern and Association Rule Mining Approaches", International Journal of Information Technology and Computer Science (IJITCS), Vol.6, No.3, pp.70-81, 2014
2. Gouda, K. and Zaki,M.J. GenMax : An Efficient Algorithm for Mining Maximal Frequent Itemsets', Data Mining and Knowledge Discovery, 2005, 11: 1-20
3. Suman, Anuradha, Gowtham and Ramakrishna, "A Frequent Pattern Mining Algorithm Based on FP - Tree Structure and Apriori Algorithm", International Journal of Engineering Research and Applications (IJERA), Vol. 2, No. 1, pp. 114-116, 2012
4. Sridevi and Ramaraj, "Finding Frequent Patterns Based On Quantitative Binary Attributes Using FP-Growth Algorithm", International Journal of Engineering Research and Applications, Vol. 3, No. 6, pp. 829-834, 2013
5. C. K.-S. Leung, Q. I. Khan, Z. Li and T. Hoque, "CanTree: acanonical-order tree for incremental frequent-pattern mining," Knowledge and Information Systems, Vol. 11, No. 3, pp. 287-311, 2007
6. S.K. Tanbeer, C.F. Ahmed, B-S. Jeong and Y-K. Lee, "Efficient single-pass frequent pattern mining using a prefix-tree," Information Sciences, Vol. 179, No. 5, pp. 559-83, 2009
7. Pragya Agarwal, Madan Lal Yadav and Nupur Anand, "Study on Apriori Algorithm and its Application in Grocery Store", International Journal of Computer Applications, Vol. 74, No.14, pp. 1-8, 2013
8. G. Grahne and J. Zhu, "Fast algorithms for frequent itemset mining using FP-Trees," IEEE Transaction on Knowledge and Data Engineering, Vol. 17, No. 10, pp. 1347-62, 2005
9. Uday Kiran and Krishna Reddy, "Novel techniques to reduce search space in multiple minimum supports-based frequent pattern mining algorithms", In Proceedings of the 14th International Conference on Extending Database Technology, pp. 11-20, 2011
10. Jyoti Jadhav, Lata Ragha and Vijay Katka, "incremental Frequent Pattern Mining", International Journal of Engineering and Advanced Technology (IJEAT), Vol. 1, No. 6, pp. 223-228, 2012
11. Tanbeer, S.K, Ahmed C.F and Byeong-Soo Jeong, "parallel and distributed algorithm for frequent pattern mining in large databases", In Proceedings of the 11th IEEE International Conference on High Performance Computing and Communications, Seoul, pp. 407-414, 2009