2024, 21(5): 1743-1753
https://dx.doi.org/10.21123/bsj.2023.8018
P-ISSN: 2078-8665 - E-ISSN: 2411-7986

Baghdad Science Journal

Modified Meerkat Clan Algorithm for Association Rules Mining

***Mohamad Ab. Saleh*[1]*, *Ahmed T. Sadiq*[2]**

[1]Banking and Financial Department, College of Economic and Administration, Al Iraqia University, Baghdad, Iraq.
[2]Computer Science Department, University of Technology, Baghdad, Iraq.
*Corresponding Author.

Received 29/10/2022, Revised 05/05/2023, Accepted 07/05/2023, Published Online First 20/10/2023, Published 01/05/2024

Abstract

Association Rules Mining (ARM) forms one of the important data mining techniques. The classical methods that were previously worked on by researchers have become ineffective to deal with the steady growth of databases, which prompted us to use the mining process for association rules based on metahuristic, and in our work all the correct rules will extracted, and mining is not limited to high-quality rules. Swarm intelligence based is one of these methods. In this paper, Modified Meerkat Clan for Association Rules Mining (MCC-ARM) has been proposed. Basically, the proposed algorithm depends on Meerkat Clan Algorithm (MCA). The greatest benefit is the diversity of candidate solutions in MCA. In our work the rules will represented using two methods which are borrowed from the genetic algorithm; in the first one each group of rules refers to object in society which is called Pittsburgh; while the second one each rule refers to an object in society which is called Michigan. The proposed algorithm aims to inspect for the maximum possible number of correct association rules. The so-called algorithm follows the approach of defining the effective search area, which depends on a main random mechanism to lead the algorithm in extracting alternative rules and avoiding total solutions from being guided by the same rule, and this led to a great deal of diversity. In addition, the MCC-ARM uses condensation method in the adjacency search process to prevent the algorithm from falling into the local mode. In order to prove their efficiency, it should be applied on four reliable datasets (i.e. Zoo, German Credit, Primary Tumor and Chess). The enhancement brought about by the proposed algorithm has obtained two crucial factors, namely on the number of correct rules and quality fitness value.

Keywords: Association Rules Mining, Bees Algorithm, Genetic Association Rules, Meerkat Clan Algorithm, Vicinity Space.

Introduction

Association rules mining has been a promising research area since it first emerged Agrawal et al [1]. Nowadays, one of the most promising areas for pattern discovery is mining through the preceding rules.

The problem of the Association Rules Mining (ARM) may be formulated mathematically as follows: T= {t1, t2, t3, …, tm} represents a collection of the transactions, and A= {a1, a2, …,an} represents a group of the items. An implication of the association rule takes the form $X \rightarrow Y$, where X & Y represent sets of the items, $Y \subset A$, $X \subset A$, and $X \cap Y = \phi$.

The set A' is a collection of transactions such that $A' \subseteq A$. The usefulness of the association rule is defined using two parameters. The first one is support of a rule, while the 2nd represents the confidence of a rule. The support of a rule represents redundancy of targeted association rules that are discovered in the database. The determination operation of association rule supporting, e.g., X →Y, is calculated by dividing the transactions supporting number by total number of transactions, e.g., sup (X→Y) = sup (XUY)/|T|. The association rule confidence refers to the probability that the existence of Y transaction is on the condition of the X transaction. The rule confidence of X→Y can be represented by conf (X→Y) = sup (XUY)/sup(X). The confidence level of a rule refers to the association rules' strength. The idea of mining problem for the association rules can be expressed by elicitation of all the rules from a specific database with support ≥ Minimum support (Minsup) and confidence ≥ Minimum confidence (Minconf), where the values of the Minconf and Minsup are fixed by a user.

Basically, the idea of mining the association rules for a specific database is based on a collection of procedures introduced by Agrawal *et al.* [1], Srikant and Agrawal [2] through two important steps which are:

1- Specifying all the sets of ingredients that carry the supports value greater than, or equal to, the Minsup criteria, which is already specified by the user.

2- Producing all the rules that meet the Minconf criteria.

Anyway, the classical methods have become ineffective in dealing with the database inflation; in addition to that, the process of best rules electing in a single run of the relative algorithm represents exceptional importance. Furthermore, the collection of the useful rule gets the attention of the user, but not the whole rules related to the huge amount of database led to the mining process for association rules depending on metaheuristic algorithms. If we focus on the classical algorithms, we will notice that they search on the high-quality rules but not all the intact rules[3,4].

In the last decade, swarm intelligence algorithms have become a promising scientific research area, and they refer to metaheuristic algorithms that trigger their ideas from nature. We can define Swarm Intelligence (SI) as the group behaviour which is found in nature with two important characteristics that are the self-regulation and uncenterlized systems. Examples of such groups in natural systems comprise the hawks hunting, ant colonies, bacterial growth, fish schooling, beehives, bird flocking, and microbial intelligence where a collection of agents are interacting with each other and with the environment in order to reach smart solutions for some of the complicated problems[5] such as work distribution and replacement among agents in the population, food searching for animals, nest building for birds, … etc[6].

Many complicated problems are solved using a swarm algorithm where it is characterized by being inspired by the societies in nature, and it has the ability to produce solutions that are strong, swift, and inexpensive[7,8]. A number of swarm algorithms have emerged and been implemented in a successful way in many life domains. Some examples of swarm intelligence algorithms are the optimization procedures for each Particle Swarm[9], Ant Colony[10], Bees Swarm[11] in addition to the Cuckoo Search algorithm[12].

In the presented work, a new approach depending on Meerkat Clan Algorithm is suggested for mining association rules for extracting the largest numbers of the valid rules with the high values of confidence and support[13,14]. The remainder of the present work has been arranged in the following manner. Section2 discusses studies that are related to the present research. Brief explanation of the Meerkat Clan algorithm and its concepts is offered in Section3. Modified Meerkat Clan algorithm for Association Rules Mining (MMC-ARM) will be presented in Section 4. Whereas Section 5 describes experimental setup and results. A conclusion with the summary is provided in Section6.

Materials and Methods

Related Works

Mata et al. are among the firsts researches who proposed the concept of association rules mining depending on metaheuristic algorithm, where the algorithm is relied upon the genetic algorithm (GA) and called the Genetic Association Rules (GENAR)[15]. The preceding algorithm goes ahead with finding the numerical association rules, relying on values distribution of quantitative features, and preventing the agents from the repetition of solution (rule) using a unique methodology. The same previous researchers presented another research with a new algorithm that is entitled Genetic

Association Rules (GAR), where it is characterized by finding recurrent item sets in the condition of giving k-item set for each agent within the system Mata et al[16]. Another algorithm is presented for association rules mining where the important feature of it is the overcoming of non-concourse by using mutation rate with the ability of adaptation to avoid the abnormal differences at the 1st generation Guo & Zhou[17]. Yan et al [18], proposed another idea for ARM depending on the genetic algorithm that is entitled ARMGA. This algorithm produces a collection of high fitness rules depending on the relative confidence. The disadvantage of the algorithm is that it does not take into consideration the rules with unfixed length, minimum confidence, and minimum support restrictions.

All of the previous algorithms are characterized by essential operations limits since the produced solutions may be unacceptable although the antecedent and consequent parts own the same item, and even worse, there is no practical solution to dealing with this problem. Mohammed et al.[19] presented an algorithm based upon Bees Swarm Optimization metaheuristic (BSO), which is called MBSO-ARM that are used for mining a collection of association rules with premium quality. This algorithm is differentiated by the useful rules number, fitness value, and the enhancement in computational time in comparison with the older version, which is BSO-ARM.

Meerkat Clan Algorithm (MCA)

The "Meerkat Clan Algorithm" meta-heuristic can be represented as population-based search algorithm that has been inspired by food searching behaviours of the meerkat clan in the desert that was first proposed in[20]. This algorithm is used to get the preferred solution by using the effective method for solving the optimization problem. It is based on a group of three basic social behaviour elements. The sentry is the first social behaviour of the meerkat clan, which means choosing one or more as a guard or observer while others are hunting or playing, and that is to inform them in the event of any dangerous emergency. In case of danger, the sentry will alert the group by barking, which will lead others to flee to the burrows until the danger has passed. The second social behaviour is foraging which indicates the usual activity carried out by social animals in order to feed individually while maintaining audio and visual communication with each other. In addition to that and in a systematic manner within the framework of the foraging process, the meerkat clan take a different path every day and leave the area that they visited for at least a week in order to provide an opportunity for the region to renew its food supplies. The final social behaviour is the babysitter, where the support is provided by a number of meerkats clans as a babysitter while the rest of the group are on the trip of the foraging process. Algorithm1 describes general processes of the MCA meta-heuristic[20].

Algorithm 1: The general MCA.

a. Initialization: randomly creating a clan of the individuals and setting other parameters' clan size, care size, foraging size, and worst foraging and care rates.

b. Computing clan fitness.

c. Choosing the optimal one as the 'sentry.'

d. Dividing clan to 2 groups (care and foraging).

e. Generating the foraging group neighbours.

f. Chosing the worst members in foraging group and swapping with optimal ones in care group.

g. Dropping the worst members in care group and randomly generating another individual.

h. Replacing the optimal member in the foraging with the sentry if it's optimal.

The pseudocode for an algorithm is illustrated below:

```
Input: Clan size ($n$), foraging size ($m$), care size ($c$),
worst foraging rate ($Fr$), worst care rate ($Cr$),
neighbour solution ($k$).
Output: Best Solution.
Begin
    Generate random clan of solutions clan ($n$)
    Compute fitness for clan solutions
    Sentry= best solution of clan ($n$)
    Divide the clan into two groups (foraging &
care).
    While not terminating condition Do
        For $i$= 1 to $m$
```

```
            Generate neighbours from foraging set
            Foraging (i) = best one from k neighbour
        End for
        Swap the worst for Fr solution in foraging group with best ones solution in care group.
        Drop the worst Cr solution from care group and generate ones solution randomly.
        Select the best one of foraging call it best-f or g
        If best-f or g ≤ Sentry then Sentry = best-f or g
        End if
    End while
End
```

MMC-ARM Algorithm

In this section, we'll look over initialization, rule representation, the search area determination approach, neighbourhood search, fitness function, which are all important elements of the algorithm.

Rule Representation

To represent rules, there are two methods that can be used; the first one is called Pittsburgh, while the second is called Michigan. These methods are borrowed from genetic algorithms that may be generalized to the rest of algorithms that are based upon the population[21]:

- Pittsburgh Method

In this method, every member of society denotes a group of the rules.

- Michigan Method

In this method, every member of society denotes one rule.

The Michigan method is what we will utilize in our work as a way to represent rules. The encoding technique used is a mix of two famous methods that are binary and integer encoding, which is used in[22]. Depending on the binary encoding method, the rule which acts as a solution is denoted with using a vector S of n items. For example, if S [i] = 1, this item will be represented in the rule, whereas when the value of S [i] = 0, the item won't be represented in the rule. Furthermore, S indicates the solution rule, which is represented depending on integer encoding with k+1 items. The k number refers to the size of the solution rule, while the extra one to k represents the separator tag between antecedent and consequent parts of the rule. Regarding vector S, if S[i] corresponds to a certain value, then this value will appear at the rule's i^{-th} position. Depending on the encoding mechanism that has been suggested by[23], this solution has been represented by a vector S, which contains n items that represent the whole number of them in the data-set. The items' locations of S can be defined as follows:

1. The item i will not be presented in the solution if the S[i] value equals zero.

2. The item i will be presented in the antecedent part of the solution if the value of S[i] is equal to one.

3. The item i will be presented in the consequent part of the solution if the value of S[i] equals two.

Example1: Let T = {t1, t2, t7} be a collection of items

Suppose we have a set of items called T, where T = {t1, t2, t7}. If S1= {1, 0, 1, 0, 0, 0, 2} then V1 will represent the rule R1 such that: t1, t3 → t7. In addition to that, if S2= {1, 2, 0, 0, 0, 1, 0} then V2 will represent rule R2 such that: t1, t6 → t2.

The fitness function can be computed more easily because of the separating process between the antecedent and consequent of the preceding representation.

The Primitive Solution

In our algorithm MMC-ARM, we avoided the pure random process of selecting the primitive solution since such randomness can affect some rules that cause to uncover of the starting data that may lead to a number of invalid rules. In addition to that, using a non-random configuration can enhance the solution quality and reduce the runtime, as we can see in[24]. Anyway, the process of non-random configuration needs search space information which may not be available in some situations. In order to overcome the preceding problem, the following approach is proposed:

First of all, a primitive solution P of size n (n = total number of data set items) will be selected, as we can see in the following steps:

The first step: is placing in a random way in P the two items that are 1 and 2, whereas the whole remaining positions of P are getting the zero value.

The second step: is the choosing of the initial solution (S_indc) using the replacement process over n times for the items of the primitive solution P. After that, the solutions fitness function that is generated from this replacement process is evaluated, and depending on that, the first (S_indc) will emerge with high fitness quality solution.

Search Area Determination Strategy
The strategy for search area determination that was used in our work was first proposed in[19]. It starts by adding one value to two bits of the initial solution S_indc, where one of them is being chosen randomly. The preceding process is reiterated for n times which represents the length of S_indc. Algorithm 2 explains the so-called strategy.

Algorithm 2: Determination of Search Space
Input: S_indc, n // initial solution, n items in S_indc
Outputs: SoS: Array [1...n][1...n]

```
Begin
    r← random integer of 0 ≤ r≤ n
    For i=0 to n
        Copy (S_indc, SoS[i])
        For j =0 to n do
            If j= i Then
                SoS[j] = SoS[j] +1
                If SoS[j] > 2
                    SoS[j] =0
                End if
            End if
            SoS[r] = SoS [r] +1
            If SoS[r] > 2 Then
                SoS[r] =0
            EndIf
        EndFor
    EndFor
 Return SoS
End.
```

Example 1: Considering S_indc = {1, 0, 1, 0, 0, 2, 0}
1) add a value of 1 to 1st bit and to a different bit that has been randomly chosen in S_indc:
S1= {$\underline{2}$, 0, 1, 0, $\underline{1}$, 2, 0}
2) add a value of 1 to 2nd bit and to a different bit that is randomly chosen in S_indc:
S 2= {1, $\underline{1}$, 1, $\underline{1}$, 0, 2, 0}

The Vicinity Space
The process of calculating the vicinity space is done through the addition or subtraction of a value of one from one bit of the solution S that is selected randomly, where this process is done for each solution S[25,26]. In addition to that, there is an adaptive value (Adp) which is used to specify if the mathematical operation is plus or minus that is applied to the targeted bit as we can see in algorithm 3. The target of using the adaptive rate is the unfavorable cases that result from the use of fixed Adp value, as we can see in the following two cases:

First case: when the value of Adp is too small, the majority of solutions will possess a fitness function with a small value. The previous phenomenon is caused by the big increase in the volume of solutions (rules) which will lead to rules (solutions) with weak support, and over time, most solutions will become not useful unfortunately.

Second case: when the value of Adp is large, this may lead to getting a small number of solutions that are closed to the optimal one, which will be deleted instead of approaching in the evolution process. The foregoing will lead to the great possibility of falling into the trap of the optimal local solution. This situation happens because the work is done frequently on a small number of solutions with high support within the same population. Anyway, there is the possibility of a situation in which the values of a particular site are omitted for solution in the algorithm's early stage.

Depending on the above, the Adp is calculated using the following equation:

$$Adp = (i / Maxlter) \wedge EXP^{-1} \text{ .... } 1$$

The variable i represents the existing iteration, and the variable MaxIter represents the topmost value of iterations. Depending on Eq 1 the Adp value will be either 0 or 1, where it grows with the growing of the variable i; therefore, the magnitude of the laws is avoided over time. In addition to that, it prevents the algorithm from falling into the local optimum trap by preventing the process from reducing some elements in the early compensation stages.

Algorithm3: Neighbourhood Space Search
Inputs: Sol, K, n // M Foraging Meerkat, S represents solution assigning to a bee, number, n represents number of the items in S.
Outputs: SoN: Array [1... M*n][1...n] // Space of Neighborhood.

```
Begin
    a← 0
    While a< M
```

```
                For j =0 to n
                    Copy (Sol, SoN [a])
                    r← random where $0 \le r \le 1$
                    j ← integer random where
                    $0 \le h \le n$
                    If r > Add_Sub Then
                        SoN[h] = SoN[h] +1
                        If NeighborhoodSpace [h]>2
                            SoN[h] =0
                        EndIf
                    Else
                        SoN[h] = SoN[h] - 1
                        If SoN[h] <0
                            Neighborhood _Space [h] =0
                        EndIf
                    EndIf
                EndFor
                a← a+ 1
            EndWhile
    Return SoN
End.
```

Fitness Function

This function is the practical expression of the primary goal that is standing behind the association rules mining, which represents the detection of all the association rules that possess confidence and support values not less than the value of the threshold. As we can see in the fitness function below, the s, represent the solution, and there are two experimental parameters, a and b, where a+b=1 [22]:

Fitness(s) = axConfidence(s) +bxSupport(s)
if Confidence(s) $\ge$ Minconf & Support(s) $\ge$ Minsup
otherwise, Fitness= -1.

The Algorithm

Initially, generating the reference population (S_ref) (i.e., initial solutions), search area determination strategy will explore the best solution to find the best solution (called sentry). Then selecting the foraging solutions group (size m) and making an improvement on these solutions via neighbourhood operation, this step plays a big role in increasing the diversification of current solutions and implementing the exploration of more solutions. The best-generated neighbourhoods will be put in the best group, and the best one will be the best solution (sentry). Then replacing worst solutions in Foraging by best in care group, and replacing worst solutions in Care group by randomly ones. The above steps will be repeated till choosing the maximum iteration. The generated rules will be formed from the best group. The below algorithm represents the basic stages for generating association rules using the modified meerkat clan algorithm.

Algorithm 4: MMC-ARM

```
Input:
        n           Clan size;
        m           Foraging size;
        k           Neighbor solutions;
        TD          Transactional Dataset;
        Minsup      Minimum Support;
        Minconf     Minimum Confidence;
Output:
        Set of Association Rules;
Begin
    Generate random clan of solutions clan(n);
    Sentry = Best one from the generated clan;
    While not termination condition Do
        Generate Search Area from Sentry via Determination of Search Space;
        Evaluating every one of the solutions in Search Area;
        Choosing m solutions from the Search Area call Foraging group;
        For each solution (F) in Foraging group Do
            Generate (k) neighborhoods from solution F;
            Put the best neighborhood in Foraging group;
        End for
        Add the optimal solutions to the Best group;
        Sentry ← the optimal solution in Best group;
        Replace worst solutions in Foraging by Best in Care group;
        Replace worst solutions in Care group by randomly ones;
    EndWhile
    For every solution R in the Best group, Do
        Generating rule from R
    EndFor
End.
```

Results and Discussion

Four standard data-sets were selected to confirm the efficiency of the proposed algorithm; these datasets were downloaded from http://www.philippe-fournier-viger.com/spmf/index.php?link=datasets.php and https://dtai.cs.kuleuven.be/CP4IM/datasets. Table 1 shows the specifications regarding such a dataset.

Table 1. Dataset Specifications.

Data-set Name	Size of the Transactions	Size of the Item
Zoo	101	36
German Credit	1000	112
Primary Tumor	336	31
Chess	3196	75

Experiments were performed using the proposed method and modified bees' algorithm for association rules (MBSO-ARM) [19]. There are parameters for MMC-ARM and MBSO-ARM. Table 2 lists the parameter values for each method.

Table2. Parameters Values.

MMC-ARM		MBSO-ARM	
Parameter	**Value**	**Parameter**	**Value**
Meerkat Clan	20 – 30	Bees' No.	8 – 12
Foraging Size	12 – 16	A	0.6
Worst Ratio	0.2	B	0.4
Neighbors	3 – 5	Min-Sup	0.1
a	0.6	Min-Conf	0.7
b	0.4	Max. Iteration	150 - 200
Min-Sup	0.1	Average Execution No.	10 - 12
Min-Conf	0.7		
Max. Iteration	150 – 200		
Average Execution No.	10 – 12		

Table 3 shows the average fitness of MMC-ARM and MBSO-ARM and explain that the proposed MMC-ARM algorithm exceeds MBSO-ARM with reference to the fitness of solutions. The Total numbers of non-redundant rules as well as number of corrected rules are shown in Table 3 and Table 4 (illustrated in Figs 1 and 2).

Table3. No. of Total and Correct Rules where Meerkat No. = 24 (12 Foraging) and Bees No. = 8.

Dataset	MMC-ARM		MBSO-ARM	
	Total Rules	**Correct Rules**	**Total Rules**	**Correct Rules**
Zoo	620	608	582	558
Primary Tumor	567	558	511	498
German Credit	675	668	635	621
Chess	615	615	612	602

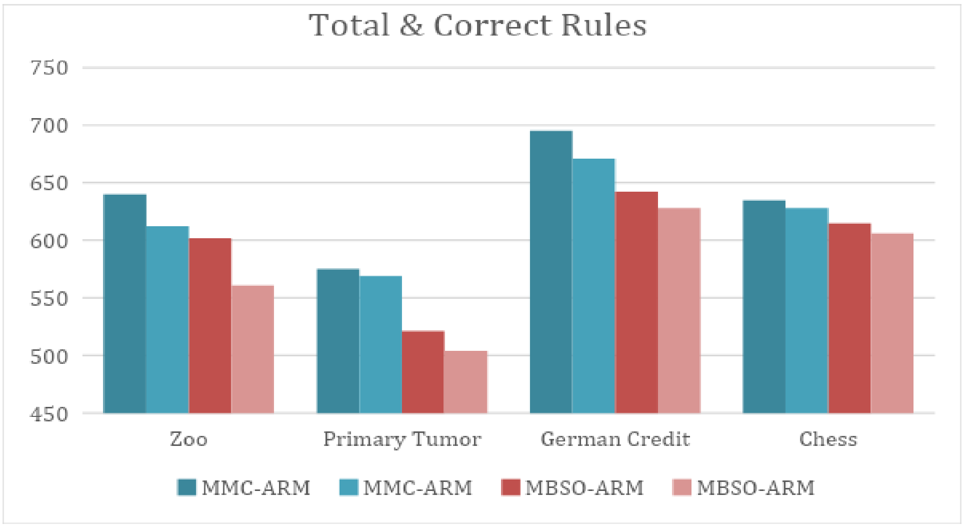

Figure 1. Chart for Total and Correct Rules where Meerkat No. =24 (12 Foraging) and Bees No.=8

Table 4. No. of Total and Correct Rules where Meerkat No. = 30 (16 Foraging) and Bees No. = 12.

Dataset	MMC-ARM		MBSO-ARM	
	Total Rules	Valid Rules	Total Rules	Valid Rules
Zoo	640	612	602	561
Primary Tumor	575	569	521	504
German Credit	695	671	642	628
Chess	635	628	615	606

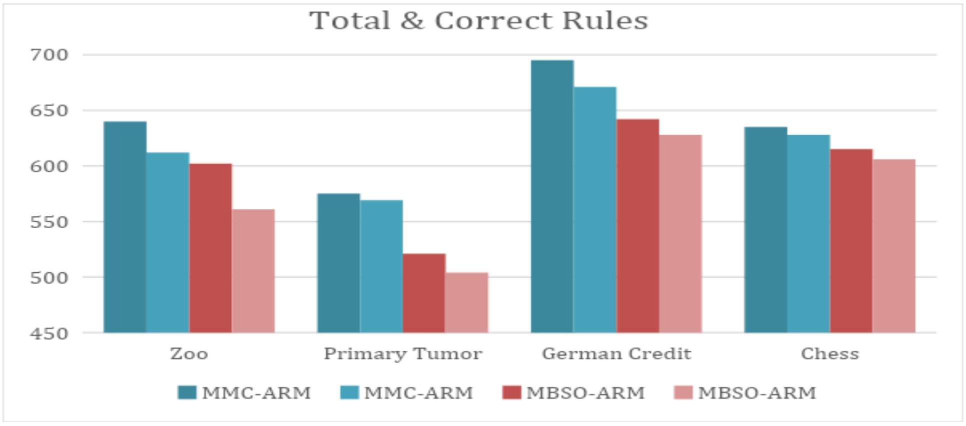

Figure 2. Chart for Total and Correct Rules where Meerkat No. =30 (16 Foraging) and Bees No.=12

Table 5 shows the execution time of the two methods. It is an indication of the fact that MMC-ARM needs more than seconds compared with MBSO-ARM; the ratio of time increasing is 3.5%, this is because of its diversification strategy.

Table 5. Average Execution Time (Sec.) where Meerkat No. = 30 (16 Foraging) and Bees No. = 12.

Dataset	MMC-ARM	MBSO-ARM
Zoo	69	63
Primary Tumor	246	239
German Credit	3295	3214
Chess	6418	6389

The obtained results may be explained as follows: The suggested algorithm has a high degree of diversification due to an effective search area determination approach which uses a basic randomization mechanism to aid the algorithm in producing alternative rules and avoiding group solutions from tending to the same rule. In other cases, the frequency of repeated rules climbed to more than 45% in the case when the search area was examined without such randomization. Furthermore, the MMC-ARM neighbourhood search operation employs the intensification method, preventing the algorithm from becoming stuck in a local mode. When MMC-ARM is investigating a search area, the algorithm's capacity to reach larger portions of the search space increases. The size of rules that are generated by the suggested algorithm is clearly larger compared with those generated by BSO-ARM of four datasets. Because of its diversification, the suggested MMC-ARM creates a higher number of total and correct rules in 4 datasets compared to MBSO-ARM. MMC-ARM diversification technique occasionally increases the risk of selecting a bad solution as reference solution, reducing the number of valid rules and the fitness. Also, because MMC-ARM utilizes a deterministic neighbourhood search with less randomization, the chances of the search around the current best are limited.

Conclusion

A modified Meerkat clan algorithm was proposed for finding the largest possible numbers of correct association rules. We applied it on four standard datasets as a practical application so as to prove their efficiency. The suggested algorithm gave good results compared to the MBSO-ARM in regards to the number of correct rules and quality fitness value, but there is about a 3.5% increase in the execution time. The good diversification of the Meerkat clan algorithm is the main reason for improved results (i.e., more correct association rules). As future work, using more efficient neighbour's generation techniques for more performance and less time.

Acknowledgment

The cooperation of the technical staff of computer laboratories at the University of Technology and the Al Iraqia University is appreciated.

Authors’ Declaration

- Conflicts of Interest: None.
- We hereby confirm that all the Figures and Tables in the manuscript are ours. Furthermore, any Figures and images, that are not ours, have been included with the necessary permission for re-publication, which is attached to the manuscript.
- Ethical Clearance: The project was approved by the local ethical committee in Al Iraqia University.

Authors’ Contribution Statement

This work was carried out in collaboration between all authors. M. Ab. S. analyzed the target algorithms then interpreted them, wrote a draft of the manuscript and finally established the revision and proofreading. A. T. S. created the general conception of the manuscript, designed the general structure and partial details and finally the data acquisition. All authors read and approved the final manuscript.

References

1. Agrawal R, Imielinski T, Swami A. Mining association rules between sets of items in large databases. ACM SIGMOD '93 Int Conf. 1993; 22(2): 207-216. https://doi.org/10.1145/170036.170072.
2. Agrawal R, Srikant R. Fast Algorithms for Mining Association Rules in Large Databases. VLDB '94: Proc 20th Int Conf Very Large Data Bases. 1994; 487-499. https://courses.cs.duke.edu/compsci516/spring16/Papers/AssociationRuleMining.pdf.
3. Chen S, Xi J, Chen Y, Zhao J. Association mining of near misses in hydropower engineering construction based on convolution neural network text classification. 1st ed. Springer. 2022; 406 p. https://doi.org/10.1155/2022/4851615.
4. Czubryt T, Leung C, Pazdor A. Q-VIPER: quantitative vertical bitwise algorithm to mine frequent patterns. Big Data Analytics and Knowledge Discovery: 24th Int Conf, DaWaK. 2022; 219-233. https://doi.org/10.1007/978-3-031-12670-3_19.

5. Sharma A, Shoval S, Sharma, A, Pandey J. Path planning for multiple targets interception by the swarm of UAVs based on swarm intelligence algorithms: A review. IETE Tech Rev. 2022; 39(.3): 675-697. https://doi.org/10.1080/02564602.2021.1894250.
6. Sadiku M, Musa S, Ajayi-Majebi A. A Primer on Multiple Intelligences. 1st ed. Springer. 2021; pp: 211-222. DOI: 10.1007/978-3-030-77584-1_17.
7. Kicska G, Kiss A. Comparing Swarm Intelligence Algorithms for Dimension Reduction in Machine Learning. Big Data Cogn Comput. 2021; 5(3), 36-51. https://doi.org/10.3390/bdcc5030036.
8. Sharma A, Sharma A, Pandey J, Ram M. Swarm Intelligence: Foundation, Principles, and Engineering Applications.1st ed. CRC Press: Boca Raton. 2022; 140 p. https://doi.org/10.1201/9781003090038.
9. Bas E, Egrioglu E, Kolemen E. Training simple recurrent deep artificial neural network for forecasting using particle swarm optimization. Springer. Granul. Comput. 2022; 7: 411-420. https://link.springer.com/article/10.1007/s41066-021-00274-2.
10. Zhao D, Liu L, Yu F, Heidari A, Wang M, Oliva D. Ant colony optimization with horizontal and vertical crossover search: Fundamental visions for multi-threshold image segmentation. Elsevier. Expert Syst Appl. 2021; 167: 317-332. https://doi.org/10.1016/j.eswa.2020.114122.
11. Ogren R, Kong S. Optimization of diesel fuel injection strategies through applications of cooperative particle swarm optimization and artificial bee colony algorithms. Int J Engine Res. 2020; 22(9): 3030-3041. https://doi.org/10.1177/1468087420954020.
12. Liu C, Wang J, Zhou L, Rezaeipanah, A. Solving the multi-objective problem of IoT service placement in fog computing using cuckoo search algorithm. Neural Process. Lett. 2022; 54: 1823-1854. https://link.springer.com/article/10.1007/s11063-021-10708-2.
13. Mirjalili S, Song Dong J, Lewis A, Sadiq A. Particle swarm optimization: theory, literature review, and application in airfoil design. Nature-inspired optimizers: Springer; Stud Comput Intell. 2020; 167-184. https://doi.org/10.1007/978-3-030-12127-3_10.
14. Kurtuluş E, Yıldız A, Sait S, Bureerat S. A novel hybrid Harris hawks-simulated annealing algorithm and RBF-based metamodel for design optimization of highway guardrails. Mater Test. 2020; 62(3): 251-260. https://doi.org/10.3139/120.111478.
15. Mata J, Alvarez J, Riquelme J. Mining numeric association rules with genetic algorithms, Proc Int Conf *Adapt* Nat Comp Algo. Springer. 2001; 264–267. https://doi.org/10.1007/978-3-7091-6230-9_65.
16. Mata J, Alvarez J, Riquelme J. An evolutionary algorithm to discover numeric association rules, Sac 02 Proc ACM Symp *Appl Comput.* Springer. 2002; 590–594. https://doi.org/10.1145/508791.508905.
17. Guo H, Zhou Y. An Algorithm for Mining Association Rules Based on Improved Genetic Algorithm and its Application, 3ed Int Conf Genetics Evol Comput. IEEE Access. 2009; 117–120. https://doi.org/10.1109/WGEC.2009.15.
18. Yan X, Zhang C, Zhang S. Genetic algorithm-based strategy for identifying association rules without specifying actual minimum support. Expert Syst Appl Elsevier. 2009 Mar 17; 36(2); 3066–3076. https://doi.org/10.1016/j.eswa.2008.01.028.
19. Mohammed R, Duaimi M, Sadiq A. Modified Bees Swarm Optimization Algorithm for Association Rules Mining. Iraqi J Sci. 2022 Jan 18; 58(1); 364-376. https://www.iasj.net/iasj/download/e1498ea57bb18ffd.
20. Al-Obaidi A, Abdullah H, Ahmed Z. Meerkat Clan Algorithm: A New Swarm Intelligence Algorithm, Indones. J Electr Eng Comput Sci. 2018 April 1; 10(1); 354-360. https://doi.org/10.11591/ijeecs.v10.i1.pp354-360.
21. Bishop J, Gallagher M, Browne W. Pittsburgh learning classifier systems for explainable reinforcement learning: comparing with XCS. Proc Genetics Evol Comput Conf. 2022 July 9; 323–331. https://doi.org/10.1145/3512290.3528767.
22. Yang N, Mehmood D. Multi-Objective Bee Swarm Optimization Algorithm with Minimum Manhattan Distance for Passive Power Filter Optimization Problems. Mathematics. MDPI. 2022 Jan 2; 10(1) 133-153. https://doi.org/10.3390/math10010133.
23. Djenouri Y, Drias H, Habbas Z, Mosteghanemi H. Bees swarm optimization for web association rule mining. Int Conf Web Intell *intelligent* Agen Technol. IEEE Access. 2012; 142–146. https://doi.org/10.1109/WI-AT.2012.148.
24. Surry P, Radcliffe N. Inoculation to initialize evolutionary search. Comput Sci. Springer, 1996, 169-285. https://doi.org/10.1007/BFb0032789.
25. Yasser Y, Sadiq A, AlHamdani W. Honeyword Generation Using a Proposed Discrete Salp Swarm Algorithm. Baghdad Sci J. 2022: 0357-0357. https://doi.org/10.21123/bsj.2022.6930.
26. Mazher A, Waleed J. Retina Based Glowworm Swarm Optimization for Random Cryptographic Key Generation. Baghdad Sci J. 2022; 19(1); 0179-0179. http://dx.doi.org/10.21123/bsj.2022.19.1.0179.

خوارزمية فصيلة ميركات المعدلة لتعدين القواعد الرابطة

محمد عبدالرضا صالح[1] ،احمد طارق صادق[2]

[1] قسم العلوم المالية و المصرفية, كلية الأدارة و الأقتصاد , الجامعة العراقية , بغداد , العراق.
[2] قسم علوم الحاسبات , الجامعة التكنولوجيا , بغداد , العراق.

الخلاصة

تشكل قواعد جمعيات التعدين (**ARM**) إحدى تقنيات التنقيب عن البيانات المهمة. أصبحت الأساليب الكلاسيكية التي عمل عليها الباحثون سابقًا غير فعالة للتعامل مع النمو المطرد لقواعد البيانات ، مما دفعنا إلى استخدام عملية التعدين لقواعد الارتباط القائمة على ميتاهيورستك، وفي عملنا سنستخرج جميع القواعد الصحيحة ، و التعدين لا يقتصر على القواعد عالية الجودة. ذكاء السرب يعتبر إحدى هذه الأساليب المعتمدة. في هذه الورقة، تم اقتراح تعديل فصيلة ميركات لتعدين قواعد الجمعية (**MCC-ARM**). تعتمد الخوارزمية المقترحة بشكل أساسي على خوارزمية فصيلة ميركات (**MCA**). أكبر فائدة هي تنوع الحلول المرشحة في **MCA**. سنقوم في عملنا بتمثيل القاعدة باستخدام طريقتين مستعارتين من الخوارزمية الجينية، في المجموعة الأولى، تشير كل مجموعة من القواعد إلى كائن في المجتمع و التي تدعى طريقة بيتسبرغ، بينما تشير القاعدة الثانية إلى كائن في المجتمع و التي تدعى طريقة ميشيغان. تهدف الخوارزمية المقترحة إلى فحص أكبر عدد ممكن من قواعد الارتباط الصحيحة. ما يسمى بخوارزمية تتبع نهج تحديد منطقة البحث الفعالة، والتي تعتمد على آلية عشوائية رئيسية لقيادة الخوارزمية في استخراج القواعد البديلة وتجنب الحلول الكلية من الاسترشاد بنفس القاعدة، وهذا أدى إلى قدر كبير من التنوع. بالإضافة إلى ذلك، يستخدم **MCC-ARM** طريقة التكثيف في عملية البحث المتجاورة لمنع الخوارزمية من الوقوع في الوضع المحلي. لإثبات كفاءتها، سنطبقها في أربع مجموعات بيانات موثوقة (مثل حديقة الحيوان، والائتمان الألماني، والورم الأساسي والشطرنج). لقد حصل التحسين الذي أحدثته الخوارزمية المقترحة على عاملين حاسمين ، وهما عدد القواعد الصحيحة وقيمة لياقة الجودة.

الكلمات المفتاحية: تعدين القواعد الرابطة ، خوارزمية النحل ، القواعد الرابطة الوراثية ، خوارزمية فصيلة الميركات ، مساحة الجوار.