MOCANAR: A MULTI-OBJECTIVE CUCKOO SEARCH ALGORITHM FOR NUMERIC ASSOCIATION RULE DISCOVERY

Irene Kahvazadeh and Mohammad Saniee Abadeh

Faculty of Electrical and Computer Engineering,
Tarbiat Modares University, Tehran, Iran
i.kahvazadeh@modares.ac.ir
saniee@modares.ac.ir

ABSTRACT

Extracting association rules from numeric features involves searching a very large search space. To deal with this problem, in this paper a meta-heuristic algorithm is used that we have called MOCANAR. The MOCANAR is a Pareto based multi-objective cuckoo search algorithm which extracts high quality association rules from numeric datasets. The support, confidence, interestingness and comprehensibility are the objectives that have been considered in the MOCANAR. The MOCANAR extracts rules incrementally, in which, in each run of the algorithm, a small number of high quality rules are made. In this paper, a comprehensive taxonomy of meta-heuristic algorithm have been presented. Using this taxonomy, we have decided to use a Cuckoo Search algorithm because this algorithm is one of the most matured algorithms and also, it is simple to use and easy to comprehend. In addition, until now, to our knowledge this method has not been used as a multi-objective algorithm and has not been used in the association rule mining area. To demonstrate the merit and associated benefits of the proposed methodology, the methodology has been applied to a number of datasets and high quality results in terms of the objectives were extracted.

KEYWORDS

Numeric Association Rule, Cuckoo Search, Multi-Objective Algorithm

1. INTRODUCTION

Association rule mining methods are one of the most used methods to extract relationships among features of a dataset; They were introduced in [1]. An association rule, denoted by X→Y is defined with two parts, antecedent part (X) and consequent part (Y) and both of them contain an item set. There are many impressive methods to obtain association rules in various applications [2-4], although most of them require values of the features to be discrete. For this reason these techniques discretize the numeric features but this causes a loss of information. If we want to discover association rules from continuous features, we should deal with a large search space since when the features are continuous, the number of the association rules can be

Jan Zizka et al. (Eds) : ICAITA, SAI, CDKP, Signal, NCO - 2015
pp. 99–113, 2015. © CS & IT-CSCP 2015 DOI : 10.5121/csit.2015.51509

discovered are numerous. To solve the problem of large search space, one of the best suggestions is to use meta-heuristic algorithms.

The meta-heuristics are divided into two categories according to our knowledge, biological and Bio_Inspired Algorithms that are illustrated in Figure 1. The meta-heuristic methods are usually based on a physical phenomenon or based on the biological methods such as Simulated Annealing as suggested in [5], Gravitational Search Algorithm [6], Magnetic Optimization Algorithm [7], External Optimization [8] and Harmony Search [9]. We divide the Bio_Inspired meta-heuristics into two categories, evolutionary methods that use Darwin's theory directly, such as Genetic Algorithm that is suggested in [10], Genetic Programming [11], Evolution Strategy [12], Evolutionary Programming [13] and so on. Swarm intelligence based methods are other Bio_Inspired meta-heuristic algorithms that are mainly inspired by lives of living organisms. Swarm intelligence based methods also can be divided into Stigmergic based and imitation based categories. The Stigmergic based methods use an environmental memory to establish communication indirectly. The pheromone table in Ant Colony Optimization (ACO) is an example of environmental memory. ACO is suggested in [14], Honeybee Hive Optimization [15] and Termite Colony Optimization [16] are examples of the Stigmergic based methods. The imitation based methods have not any shared environmental memory and the communication between them is directly. All individuals in imitation based methods have a local memory and a global memory. The individuals are desired to the local best and global best positions. Particle Swarm Optimization [17], Imperialist Competitive Algorithm [18], Firefly Algorithm [19], Shuffled Frog-Leaping [20], Cat Swarm Optimization [21], Fruit Fly Optimization [22], Bacterial Foraging Optimization [23], Artificial Fish Swarm Algorithm [24], Bat Algorithm [25], Lion Pride Optimizer [26], Krill Herd Algorithm [27], Hunting Search [28] and Cuckoo Optimization Algorithm [29] are some examples of these methods. Mentioned taxonomy [30], The taxonomy is demonstrated in Figure 1.

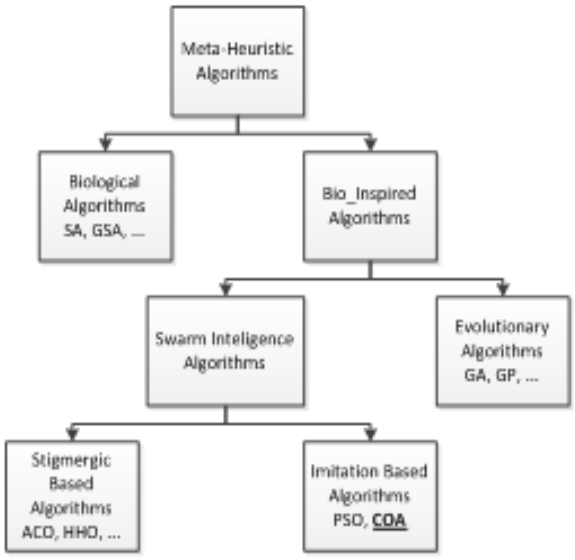

Figure 1. Meta-Heuristic Algorithms Categories [30]

If we use older algorithms, they may not have the capabilities of new algorithms; but however, they have been used in various applications and their performance is guaranteed. Choosing very new algorithms can be tricky, because they have not been used extensively and they might have unknown drawbacks. Many meta-heuristic algorithms have been used to discover association rules like [31-36] but in extracting of the numeric association rules, we have use Cuckoo Search [29] because this algorithm is one of the most matured algorithms and also, it is a simple and understandable algorithm. In addition, until now this method has not been used as a multi-objective algorithm and has not been used in the association rule mining area.

The paper is organized as follows: In the next section we describe preliminaries and in Section 3, the proposed method is explained. Section 4 contains experimental results and discussion and the last section concludes the paper.

2. PRELIMINARIES

This section consists of two subsections. In the first one, multi-objectivity concepts are described. Multi-objective approaches are divided into three categories and are explained separately. Also our objectives that are considered for numeric association rule mining are described in this subsection. In the second subsection, we discuss about the cuckoos life and review the studies that are inspired from their life.

2.1. Multi-Objectivity

Usually, the multi-objective problems are solved with one of the three multi-objective approaches: aggregation based approach, population based approach and the Pareto based approach. These approaches have their advantages and disadvantages. We studied these approaches in the following.

Aggregation based approach: in this approach, all of the objectives are combined into one objective which is done using mathematical operators like subtracting, multiplying and so on. An example is offered in (1) for this approach that uses the sum operator and weights for objectives.

$$f(x)= w\_1*f\_1\ (x)+w\_2*f\_2\ (x)+\cdots+w\_k*f\_k\ (x)$$

$$\text{where } x\in X\_f \tag{1}$$

w_iis the weight of ith objective, k is the number of objectives and X_fis the search space. This approach is one of the easiest approaches but the weights must be properly defined. One of the disadvantages of this approach is that the approach cannot discover the concave parts of the Pareto front [37]. Nonlinear aggregation functions do not have this restriction. One of the studies that used this approach is [38].

Population based approach: in this approach, the population is divided into k (that is the number of objectives) sub-populations. Each sub-population is improved with regards to one objective and finally after termination of the algorithm the sub-populations are aggregated in one solution to the k-objective problem. Since this approach is easy to use, it is well-known among the researchers, [39] is one of the studies that use a population based approach to solve multi-objective problems.

Pareto based approach: It rarely happens to have a unique solution that is optimal in terms of all objectives. So instead of looking for a unique solution that is optimal, we should trade-off between the objectives. Pareto optimality definition says that a solution is a Pareto optimal, if there exists no feasible solution in X_f which would improve some objective without causing a simultaneous deterioration in at least one other objective. [40-43] studies have used Pareto based approach. Also in this study we use a Pareto based approach that considers four objectives: support, confidence, interestingness and comprehensibility. These objectives are important in association rule mining area. The objectives are defined as follows: The support of an item set X, denoted by S(X), is the ratio of the number of records (|R_X |) that contains the item set X to the total number of records (|D|). S(X) is defined by (2). The support of an association rule is denoted by S(X→Y) and is the ratio of the number of records containing both X and Y (|R_X∪ R_Y |), to the total number of records, |D|. If the support of an association rule is 20%, this means that 20% of the analyzed records contain X∪Y. S(X→Y) is defined by (3).

$$S(X)=|R\_X|/|D| \tag{2}$$

$$S(X\rightarrow Y)=|R\_X\cup R\_Y|/|D| \tag{3}$$

The confidence of an association rule indicates the degree of correlation between X and Y in the dataset. The confidence of an association rule denoted by C(X→Y) is the ratio of the number of records that contain X ∪ Y to the number of records that contain X. If we say an association rule has a confidence of 80%, it means that 80% of the records containing X also contain Y. The confidence of an association rule is defined by (4).

$$C(X\rightarrow Y)=S(X\rightarrow Y)/S(X) = |R\_X\cup R\_Y|/|R\_X| \tag{4}$$

In addition to support and confidence measures, two other measures are used to mine high quality association rules. If the number of conditions involved in the antecedent part is less than the number of conditions in the consequent part, the rule is more comprehensible [44]. The comprehensibility is computed by (5).

$$Comp.= \log(1+|R\_X|)/\log(1+|R\_X\cup R\_Y|) \tag{5}$$

Interestingness measure refers to finding rules that are interesting or useful, not just all possible rules. In some approaches, to find interestingness the entire dataset is divided based on each feature presented in the consequent part. Since different numbers of features can present in the consequent part and because they are not predefined, this approach may not be feasible for association rule mining. So, a new expression is defined in [45] which uses the support count of the antecedent and the consequent parts of the rules. This expression is shown in (6).

$$Inter.=|R\_X\cup R\_Y|/|R\_X| *|R\_X\cup R\_Y|/|R\_Y| *(1-|R\_X\cup R\_Y|/|D|) \tag{6}$$

The equation contains three parts. The first expression describes probability of generating the rule based on the antecedent part. The second expression shows the probability based on the consequent part, and the last one (1-|R_X∪ R_Y |/|D|) describes the probability of not generating the rule based on the whole dataset.

2.2. Cuckoo's Life

Some of birds are known as Brood Parasites. These birds instead of having to build their own nest, lay eggs in the nests of other birds and so the owner of the nest takes care of the Brood Parasites eggs. The cuckoo is one of the most famous Brood Parasite birds and is an expert in deception of the other birds. The female cuckoo destroys one of the other bird's eggs and replaces her egg. Every bird differs in the color and pattern of its egg but the cuckoos have an uncanny talent for mimicry. This talent is one of the mysteries of nature. Of course, some of host birds know the stranger egg and they destroy it or they leave their nest forever. In fact, the cuckoos boosting their mimic power and the host birds boosting their identification power and these effort and fight are an incessant matter. Various types of algorithms in various applications have been proposed which are inspired by the cuckoo's life like [46-50] but our method is very similar to [29]. Details of our method will be explained in Section 3. In the paper, we suggest the multi-objective version of cuckoo search in the numeric association rule mining context for first time.

3. PROPOSED METHOD

In this section our proposed method is explained. This section consists of two subsections. In the first one, representation of the numeric association rules with cuckoo search algorithm is demonstrated and in the next one, MOCANAR is explained in detail.

3.1. Representation of Problem

In this paper, the cuckoos are represented with 2D array for association rule mining problem that is illustrated in Figure 2. The number of columns of the array is equal to n that shows the number of features in dataset and the number of rows is equal to 3 that first one shows location of each feature in current association rule, the second row shows the lower bound of the feature value and the third row shows the upper bound of the feature value in the current association rule. If the value of a cell in first row is set to 0, the related feature is not present in the association rule and if a cell in first row contains 1, it means that the related feature is in the antecedent part of the association rule and the value 2 in the cell shows the related feature is in the consequent part of the current association rule. For example, the Figure 2 shows the following association rule:

if (LL3<F3<UL3 and LLn<Fn<ULn)

then (LL2<F2<UL2)

	F1	F2	F3		Fn
Location of Feature	0	2	1	. . .	1
Lower Bound of Feature	LL1	LL2	LL3	. . .	LL n
Uper Bound of Feature	UL1	UL2	UL3	. . .	UL n

Figure 2. Representation of the Association Rules with Cuckoos.

3.2. MOCANAR: Multi-objective Cuckoo Search for Numeric Association Rule Mining

MOCANAR is a multi-objective cuckoo search algorithm that extracts high quality association rules from numeric datasets. The support, confidence, interesting and comprehensibility are the objectives that are considered in the MOCANAR. The MOCANAR extracts rules incrementally in which, in the each increment, low numbers of high quality rules are made. The number of increments is determined by NumOfIncrement parameter. To generate the low number of the high quality rules in each increment, an iterative loop is repeated NumGeneration (which is another input parameters) times. During the execution of these iterations that are called generations, the initial random association rules are improved in evolutionary way. Our chosen meta-heuristic method is the cuckoo search that intelligently improves the rules in generations. Each generation consists of two 'for-do' cycles. In the first one, since the convergence of the algorithm is very fast, NumOfRndCuckoo (another input parameter) numbers of random cuckoos are generated and are directed toward the best cuckoo by using the levy flight policy and so are replaced with worst cuckoos in the population. In the second 'for-do' cycle, each cuckoo in the population generates an egg by using the levy flights and so pa percent of the generated eggs are eliminated. Pseudo code of the MOCANAR is illustrated in Figure 3 and is explained in detail below. In the pseudo code, FinalNonDominateds keeps the non-dominated association rules from last increment. When the increments are finished, the FinalNonDominateds contains the final non-dominated association rules which will be shown to the user. The non-dominated association rules that are achieved in generations are stored in Non_Dominateds. DataArray stores the dataset. The increments are started from line 6. In line 8, the population is initialized by the InitializePopulation function. In this function, PopulationSize is the parameters that specifies the number of cuckoos of the population, Per0 specifies the possibility of placing a value of zero, Per1 specifies the possibility of placing a value of 1 and Per2 specifies the possibility of placing a value of 2 in the first row of association rules for each features that illustrated in Figure 2. In line 9, CheckConditionAndFixfunction checks two defined conditions for association rules: the first one, there should be at least one feature in the antecedent part of the rule and at least one feature in the consequent part of the rule; the second condition says that the range of lower bound and upper bound of the normalized features should not be greater that F_interval(that is another input parameter). Because the rules should not be too general, the F_interval parameter is used. The statements that are in the 'for-do' statement in line 10, are executed for NumGeneration (that is one of the input parameters) times. EvaluateObjectives function in the pseudo code calculates our objectives. 'for-do' statement in line 12 is executed NumOfRndCuckoo (that shows number of random cuckoos in each generation) times. High value for NumOfRndCuckoo increases the exploration ability of the algorithm and low value increases the exploitation ability of the algorithm. GetBestCuckooWithTournament function, determines the best cuckoo with Pareto policy in terms of our objectives, in which NumOfTourn numbers. of the cuckoos in population are selected randomly and so non-dominated cuckoos in terms of our objectives are removed. One of the non-dominated rules is returned by GetBestCuckooWithTournament randomly. In line 15, the generated random cuckoos are directed to the selected best rule by levy flight policy. The Levy flight essentially provides a random walk while the random step length is specified by a Levy distribution that is shown in (1)

$$\text{Levy} \sim u=t^{(-\lambda)} \quad ,1<\lambda<3 \qquad (1)$$

The levy distribution has an infinite variance with an infinite mean. The implementation of this distribution to directing the cuckoos toward best cuckoo in detail is shown in Figure 4 for the readers that want to implement it. This directed rule is replaced with worst rule in the population. In lines 18-21, each of the cuckoos in population are directed to best cuckoo by using levy flight policy; the best cuckoo is selected with GetBestCuckooWithTournament function. The directed cuckoos are known as cuckoo eggs. These eggs should be checked in terms of the aforementioned conditions and should be evaluated in terms of our objectivities. DoChoosing function chooses PopulationSize numbers of the cuckoos in population and eggs in terms of our objectives and put them in the new population. This function is explained later with pseudo code. Current population is merged with last non-dominated rules by using Mergefunction and so the duplicated rules are deleted from them and later, the non-dominated rules are selected from them by using Pareto policy. This non-dominated rule set is related to generations and is different with the increment's non-dominated rule set. The generation's non-dominated rules are accumulated in increment's non-dominated rule set at the end of each increment. Then the Per0,Per1 and Per2 parameters are changed randomly to investigation of the other spaces of the search space in each increment. Changing those parameters helps to have different rules in each increment. Finally, the duplicated rules in FinalNonDominateds rule set are eliminated. In the Figure 4, SourceCuckoo is the cuckoo that should be directed toward TargetCuckoo (best cuckoo). NumOfAttributes parameter shows the number of dimensions of the cuckoo (the number of data features) and P_(Mut)parameter specifies the probability of mutation on each dimension of the cuckoo. w_1,w_2 and w_3 are the step sizes of cuckoo rule in each row of Figure 2 respectively which should be related to the scales of the problem of interests. In most cases, we can use 1 value for them. SourceCuckoo.rule in pseudo code refers to the 2D array shown in Figure 2. The rows of the array in each dimension are directed toward the TargetCuckoo. After this process, mutation operation is applied to each dimension by probability of P_(Mut). the resulted cuckoos are known as new eggs that should be checked in terms of aforementioned two conditions In line 24 of Figure 3, we have two populations (CuckooEggs,Population) and both of them have PopulationSize numbers of eggs and cuckoos respectively, and The DoChoosing function tries to select the PopulationSize numbers of them for the new population, since the size of population in the algorithm is constant. The Obj-share in the DoChoosing pseudo code shows the share of each objectives in new population. If its value is equal to 25 it means that 25 places in population are reserved per each objective. First, the eggs population is sorted with respect to support and so Pa percent of eggs in that population are deleted according to cuckoo search policies. Then two populations are merged in a population called tempRules; from here, the eggs are known as cuckoos. The tempRules are sorted with respect to support and so the Obj-share number of cuckoos are elected to be placed in the new population. The elected cuckoos are eliminated from tempRules.Also, this process is applied to confidence, interestingness and comprehensibility objectives. Finally, the new population is returned to main method. Here it can be said that we use the concepts of the Population based multi-objective approach.

```
Function Mocanar returns a set of association rules
Input: PopulationSize, pa, P_Mut, NumOfTourn, MaxGeneration, Per0, Per1, Per2,
                    SupPercent, NumOfIncriment, NumOfRndCuckoo, w1, w2, w3
Output: Non-Dominated Association Rules
EndNonDominateds ← Ø
DataArray ← ReadFromDataSet(Address)
For runs = 0 to NumOfIncriment Do
    Non_Dominateds ← Ø
    Population ← InitializePopulation(PopulationSize, Per0, Per1, Per2)
    CheckConditionAndFix(Population, P_interval)
    For Generation = 0 to MaxGeneration Do
        EvaluateSupport(Population)
        EvaluateConfidence(Population)
        For RndCuckoo = 0 to NumOfRndCuckoo Do
            BestCuckoo ← GetBestCuckoo(SupPercent )
            NewRndCuckoo = GenerateRandomCuckoo( )
             Cuckoo ← GetNewCuckooByLevyFlights(NewRndCuckoo, BestCuckoo, P_Mut, w1, w2, w3)
            Replace Cuckoo With Worst Cuckoo
        End For RndCuckoo
        For i = 0 to PopulationSize Do
                BestCuckoo ← GetBestCuckooWithTournoment(NumOfTourn, SupPercent)
                populationCucko = population[i]
                 CuckooEggs[i] ← GetNewCuckooByLevyFlights(populationCucko, BestCuckoo, P_Mut)
            End For i
             EvaluateSupport(CuckooEggs)
             EvaluateConfidence(CuckooEggs)
             CheckConditionAndFix(CuckooEggs, P_interval)
            Population ← DoChoosing(CuckooEggs, Population, pa, SupPercent)
             tempRules ← Merge(Non_Dominateds, Population)
             tempRules ← DelDuplicatedRules(tempRules)
             Non_Dominateds ← DetermineNonDominateds(tempRules)
        End For Generation
         EndNonDominateds ← Non_Dominateds
         ChangeParametersRandomly(Per0, Per1, Per2)
    End For Runs
    EndNonDominateds ← DelDuplicatedRules(EndNonDominateds)
CalculateObjectives(EndNonDominateds)
Print Objectives
```

Figure 3. Pseudo Code of the MOCANAR

```
Function GetNewCuckooByLevyFlights returns Directed Cuckoos toward Best Cuckoos by Levy Flight
Input: SourceCuckoo, TargetCuckoo, P_Mut , NumOfAttributes, w1, w2, w3
Output: New Cuckoos
λ =3/2
sigma1 =(gamma (1+ λ)* sin (PI* λ /2)/ (gamma ((1+ λ)/2)* λ *(2^ ((λ -1)/2)))
sigma2= sigma1 ^ (1- λ)
For i=0 to NumOfAttributes Do
        u[i]= random()*sigma2
        v[i]= random()
        step[i]=u[i]/( abs(v[i]) ^ (1/ λ));
        stepsize1[i]= w1 * step[i] * ( TargetCuckoo.rule[0][i] - SourceCuckoo.rule[0][i])
        stepsize2[i]= w2 * step[i] * (TargetCuckoo.rule[1][i] - SourceCuckoo.rule[1][i])
        stepsize3[i]= w3 * step[i] * (TargetCuckoo.rule[2][i] - SourceCuckoo.rule[2][i])
        SourceCuckoo.rule[0][i]= round(SourceCuckoo.rule[0][i] + stepsize1[i]* random())
        SourceCuckoo.rule[1][i]= SourceCuckoo.rule[1][i] + (stepsize2[i]* random())
        SourceCuckoo.rule[2][i]= SourceCuckoo.rule[2][i] + (stepsize3[i]* random()
End For i
doMutation(SourceCuckoo, P_m)
CheckConditionAndFix(SourceCuckoo)
EvaluateObjectives(SourceCuckoo)
return SourceCuckoo
```

Figure 4: Directing the Cuckoo toward Best Cuckoo by Levy Function

```
FunctionDoChoosing returns PopulationSize Number of Rules
Input: CuckooEggs, Population, Pa, populationSize
Output: NewPopulation
Obj-share= 1/4 of Population_size
sort eggs by support and so eliminate pa percent of the eggs
tempRules←Merge(CuckooEggs,Population)
sort the tempRules by Support
Population←get the Obj-share numbers of the rules from top of sorted rules and so delete them from tempRules
sort the tempRules by Confidence
Population←get the Obj-share numbers of the rules from top of sorted rules and so delete them from tempRules
sort the tempRules by Interesting
Population←get the Obj-share numbers of the rules from top of sorted rules and so delete them from tempRules
sort the tempRules by Comprehensibility
Population←get the Obj-share numbers of the rules from top of sorted rules and so delete them from tempRules
return NewPopulation
```

Figure 5: Pseudo Code of the Dochoosing Function

4. EXPERIMENTAL RESULTS AND DISCUSSION

We assess our proposed method in three public domain datasets: Basketball, Body fat and Quake. These datasets are available from the Bilkent University Function Approximation Repository[51]. Characteristics of the datasets are shown in Table 1 in which, second column shows the number of records in each dataset and third column shows the number of features for each dataset.

Table 1. Datasets Characteristics

Dataset	Number of Records	Number of Features
Basketball	96	5
Body fat	225	18
Quake	2178	4

All of the parameters of MOCANAR are described in the Section3. The used parameters values for each dataset are shown in Table 2. First row in Table 2 shows the name of datasets, the second row shows the size of MOCANAR population for each dataset. A high value for this parameter causes the algorithm to explore more of the search space but on the other side, it is time consuming. The third row shows the number of generations in each increment of algorithm. A high value for this parameter in addition to cause further explore in the search space, leads the algorithm to a better convergence. The pa parameter shows percentage of the cuckoo eggs that are scheduled to be eliminated in each generation. A low value for pa causes the algorithm less attention to previous generation cuckoos. The NumOfRndCuckoo parameter shows the number of the random cuckoos in each generation. A high value for this parameter increases the search space of the algorithm in comparing to its exploitation ability. The NumOfIncrement specifies the number of increments in the algorithm. A high value for this parameter increases the number of final non-dominated association rules. The NumOfTourn parameter shows the number of cuckoos that should be selected in tournament selection when the algorithm finds the best cuckoo. The P_Mut parameter determines the probability of the mutation after the eggs are generated. Increasing the value of this parameter causes increasing in the exploration ability of the algorithm and also, causes the algorithm to avoid local optimums.

The F_interval parameter specifies the maximum ranges between the Lower Limit (LL) and Upper Limit (UL) of the features. A high value for this parameter causes the generated association rules to be more general. In this study its value is set to 0.5 * (max value of feature - min value of feature). w_1,w_2 and ⟦ w⟧_3 that are used in GetNewCuckooByLevyFlights function, specify the length of steps in three rows of cuckoo (illustrated in Figure 2) to move toward the better position. High values for these parameters causes increasing in steps length and this larger steps leads to a faster convergence. In this paper, w_1,w_2 andw_3 values are equal to 1. Per0,Per1 and Per2 parameters are initialized randomly in which sum of thier values is equal to 1. The values of these parameters are changed in each increment randomly to produce different association rules in the increments. The proposed method is run 10 times and the results are averaged. In the following the results are shown and compared with other studies.

Table 2. Parameters Values for each Dataset

Dataset	Basketball	Quake	Body fat
$PopulationSize$	300	300	500
$NumGeneration$	300	300	250
pa	0.3	0.2	0.3
$NumOfRndCuckoo$	1	2	1
$NumOfIncrement$	40	50	50
$NumOfTourn$	30	50	100
P_{Mut}	0.05	0.2	0.1

In Tables 3, 4, 5, 7 and 8, the results obtained from our method are compared with results from Alatas and Akin[52],Alatas and Akin [53]and Minaei, Barmaki, and Nasiri[45]. Comparison in terms of the extracted rules count is shown in the Table 3. Increasing in the number of the extracted rules causes the increasing in the discovered knowledge but on the other side it decreases the interpretability of results. In Table 4, comparisons in terms of confidence are shown. The results show that in most cases, MOCANAR yields better results. Table 5 shows that MOCANAR has got best results in terms of the support measure. Because the algorithm runs many increments and in each increment tries to generate low number of high quality rules,

MOCANAR has better support and confidence compared to other studies. Unfortunately, these increments are a little time consuming. The spent times for each dataset in 15 runs are shown in Table 6.

Table 3. Comparison in terms of Number of Association Rules

Dataset	Alatas [52]	RPSO [53]	MOGAR [45]	MOCANAR
Basketball	33.8	34.2	50	55.4
Body fat	44.2	46.4	84	47.2
Quake	43.8	46.4	44.87	28.2

Table 4. Comparison in terms of Confidence

Dataset	Alatas [52]	RPSO [53]	MOGAR [45]	MOCANAR
Basketball	0.60	0.60	0.83	0.82
Body fat	0.59	0.61	0.85	0.91
Quake	0.62	0.63	0.82	0.84

Table 5. Comparison in terms of Support

Dataset	Alatas [52]	RPSO [53]	MOGAR [45]	MOCANAR
Basketball	32.21	36.44	36.69	66.1
Body fat	63.29	65.22	65.26	79.57
Quake	38.74	38.74	36.96	51.22

Table 6. Spent Time to 15 runs

Dataset	Basketball	Body fat	Quake
Times	7m and 33 s	25m and 2s	11m and 21s

The average of the extracted rules length average in 10 runs is shown in Table 7. To compute Table 7, the average of the extracted rules length in each run is calculated and after completion of the runs, the average of the averages is calculated. In Table 8, the coverage values of the four algorithms on each dataset are shown. It shows that also in terms of coverage, our method is better than others. Finally, the results of the MOCANAR and the MOGAR algorithms are compared in terms of the interestingness and comprehensibility measures in Table 9. Because our support values are high, interestingness of the rules that are extracted with our method is low.

Table 7. Comparison in terms of Size

Dataset	Alatas [52]	RPSO [53]	MOGAR [45]	MOCANAR
Basketball	100.0	100.0	100.0	100.0
Body fat	84.12	86.11	93.52	99.
Quake	87.6	87.92	91.07	99.26

Table 8. Comparison in terms of Coverage

Dataset	Alatas [52]	RPSO [53]	MOGAR [45]	MOCANAR
Basketball	100.0	100.0	100.0	100.0
Body fat	84.12	86.11	93.52	99.
Quake	87.6	87.92	91.07	99.26

Table 9. Comparison in terms of Interestingness and Comprehensibly

Dataset	Interestingness		Comprehensibility	
	MOGAR	MOCANAR	MOGAR	MOCANAR
Basketball	0.53	0.38	0.72	0.92
Body fat	0.56	0.41	0.80	0.85
Quake	0.46	0.34	0.68	0.95

5. CONCLUSION

Because the extraction of association rules from numeric features has a very large search space, MOCANAR is suggested in the paper. Having high support, confidence, interesting and comprehensibility measures are the objectives that were considered in the MOCANAR. The rules were extracted incrementally in which, in the each increment of the algorithm, low numbers of high quality rules were made. Also in this paper, a comprehensive taxonomy of meta-heuristic algorithm was presented. Using this taxonomy, we decided to use Cuckoo Search algorithm because this algorithm is one of the most matured algorithms and also, it is a simple and understandable algorithm. In addition, until now this method was not used as a multi-objective algorithm and was not used in the association rule mining area. We demonstrate with our results that our method has high quality results in terms of our four objectives.

REFERENCES

[1] R. Agrawal, T. Imieliński, and A. Swami, "Mining association rules between sets of items in large databases," in ACM SIGMOD Record, 1993, pp. 207-216.

[2] K. H. C. Lee, Y. Tse, G. Ho, K. Choy, and H. K. Chan, "Fuzzy association rule mining for fashion product development," Industrial Management & Data Systems, vol. 115, 2015.

[3] Ö. M. Soysal, "Association rule mining with mostly associated sequential patterns," Expert Systems with Applications, vol. 42, pp. 2582-2592, 2015.

[4] V. K. Ravi, P. Rahul, and S. K. Anand, "Designing an Expert System Using Association Rule Mining for Instant Business Intelligence," Middle-East Journal of Scientific Research, vol. 23, pp. 88-93, 2015.

[5] S. Kirkpatrick and M. Vecchi, "Optimization by simmulated annealing," science, vol. 220, pp. 671-680, 1983.

[6] E. Rashedi, H. Nezamabadi-Pour, and S. Saryazdi, "GSA: a gravitational search algorithm," Information Sciences, vol. 179, pp. 2232-2248, 2009.

[7] N. Tayarani and M. Akbarzadeh-T, "Magnetic optimization algorithms a new synthesis," in Evolutionary Computation, 2008. CEC 2008.(IEEE World Congress on Computational Intelligence). IEEE Congress on, 2008, pp. 2659-2664.

[8] S. Boettcher and A. G. Percus, "Extremal optimization: Methods derived from co-evolution," arXiv preprint math/9904056, 1999.

[9] Z. W. Geem, J. H. Kim, and G. Loganathan, "A new heuristic optimization algorithm: harmony search," Simulation, vol. 76, pp. 60-68, 2001.
[10] A. Eraser, "Simulation of genetic systems by automatic digital computers. I. Introduction," Australian Journal of Biological Sciences, vol. 10, pp. 484-491, 1957.
[11] J. R. Koza and P. James, "Rice, Genetic programming (videotape): the movie," ed: MIT Press, Cambridge, MA, 1992.
[12] I. Rechenberg, "Cybernetic solution path of an experimental problem," 1965.
[13] L. J. Fogel, A. J. Owens, and M. J. Walsh, "Artificial intelligence through simulated evolution," 1966.
[14] M. Dorigo, "Optimization, learning and natural algorithms," Ph. D. Thesis, Politecnico di Milano, Italy, 1992.
[15] M. S. Abadeh, J. Habibi, and E. Soroush, "Induction of Fuzzy Classification systems via evolutionary ACO-based algorithms," computer, vol. 35, p. 37, 2008.
[16] R. Hedayatzadeh, F. A. Salmassi, M. Keshtgari, R. Akbari, and K. Ziarati, "Termite colony optimization: A novel approach for optimizing continuous problems," in Electrical Engineering (ICEE), 2010 18th Iranian Conference on, 2010, pp. 553-558.
[17] R. C. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in Proceedings of the sixth international symposium on micro machine and human science, 1995, pp. 39-43.
[18] E. Atashpaz-Gargari and C. Lucas, "Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition," in Evolutionary computation, 2007. CEC 2007. IEEE Congress on, 2007, pp. 4661-4667.
[19] X.-S. Yang, "Firefly algorithm, stochastic test functions and design optimisation," International Journal of Bio-Inspired Computation, vol. 2, pp. 78-84, 2010.
[20] M. M. Eusuff and K. E. Lansey, "Optimization of water distribution network design using the shuffled frog leaping algorithm," Journal of Water Resources Planning and Management, vol. 129, pp. 210-225, 2003.
[21] S.-C. Chu, P.-W. Tsai, and J.-S. Pan, "Cat swarm optimization," in PRICAI 2006: Trends in Artificial Intelligence, ed: Springer, 2006, pp. 854-858.
[22] W. Pan, "A new evolutionary computation approach: Fruit Fly Optimization Algorithm," in 2011 Conference of Digital Technology and Innovation Management, Taipei. Program code on the website http://www.oitecshop.byethost16.com/FOA. html, 2011.
[23] K. M. Passino, "Biomimicry of bacterial foraging for distributed optimization and control," Control Systems, IEEE, vol. 22, pp. 52-67, 2002.
[24] X.-l. Li and J.-x. Qian, "Studies on Artificial Fish Swarm Optimization Algorithm based on Decomposition and Coordination Techniques [J]," Journal of Circuits and Systems, vol. 1, pp. 1-6, 2003.
[25] X.-S. Yang, "A new metaheuristic bat-inspired algorithm," in Nature inspired cooperative strategies for optimization (NICSO 2010), ed: Springer, 2010, pp. 65-74.
[26] B. Wang, X. Jin, and B. Cheng, "Lion pride optimizer: An optimization algorithm inspired by lion pride behavior," Science China Information Sciences, vol. 55, pp. 2369-2389, 2012.
[27] A. H. Gandomi and A. H. Alavi, "Krill herd: a new bio-inspired optimization algorithm," Communications in Nonlinear Science and Numerical Simulation, vol. 17, pp. 4831-4845, 2012.
[28] R. Oftadeh, M. Mahjoob, and M. Shariatpanahi, "A novel meta-heuristic optimization algorithm inspired by group hunting of animals: Hunting search," Computers & Mathematics with Applications, vol. 60, pp. 2087-2098, 2010.
[29] X.-S. Yang and S. Deb, "Cuckoo search via Lévy flights," in Nature & Biologically Inspired Computing, 2009. NaBIC 2009. World Congress on, 2009, pp. 210-214.
[30] M. S. Abadeh, Evolutionary Algorithms and Biological Algorithms, 2012.
[31] Y. Chen, F. Li, and J. Fan, "Mining association rules in big data with NGEP," Cluster Computing, pp. 1-9, 2015.
[32] A. M. Palacios, J. L. Palacios, L. Sánchez, and J. Alcalá-Fdez, "Genetic learning of the membership functions for mining fuzzy association rules from low quality data," Information Sciences, vol. 295, pp. 358-378, 2015.

[33] F. Jiang, "Study on Adaptive Genetic Simulated Annealing Algorithm in Association Rules Mining," in Applied Mechanics and Materials, 2015, pp. 77-82.
[34] R. Kuo and C. Shih, "Association rule mining through the ant colony system for National Health Insurance Research Database in Taiwan," Computers & Mathematics with Applications, vol. 54, pp. 1303-1318, 2007.
[35] R. Kuo, S. Lin, and C. Shih, "Mining association rules through integration of clustering analysis and ant colony system for health insurance database in Taiwan," Expert Systems with Applications, vol. 33, pp. 794-808, 2007.
[36] R. J. Kuo, C. M. Chao, and Y. Chiu, "Application of particle swarm optimization to association rule mining," Applied soft computing, vol. 11, pp. 326-336, 2011.
[37] A. Abraham and L. Jain, Evolutionary multiobjective optimization: Springer, 2005.
[38] M. S. Abadeh, J. Habibi, M. Daneshi, M. Jalali, and M. Khezrzadeh, "Intrusion detection using a hybridization of evolutionary fuzzy systems and artificial immune systems," in Evolutionary Computation, 2007. CEC 2007. IEEE Congress on, 2007, pp. 3547-3553.
[39] J. D. Schaffer, "Multiple objective optimization with vector evaluated genetic algorithms," in Proceedings of the 1st International Conference on Genetic Algorithms, Pittsburgh, PA, USA, July 1985, 1985, pp. 93-100.
[40] Y. Li, X. Li, and J. N. Gupta, "Solving the multi-objective flowline manufacturing cell scheduling problem by hybrid harmony search," Expert Systems with Applications, vol. 42, pp. 1409-1417, 2015.
[41] J. Rudy and D. Żelazny, "Solving multi-objective job shop problem using nature-based algorithms: new Pareto approximation features," An International Journal of Optimization and Control: Theories & Applications (IJOCTA), vol. 5, pp. 1-11, 2014.
[42] K. Gao, P. Suganthan, Q. Pan, T. Chua, T. Cai, and C. Chong, "Pareto-based grouping discrete harmony search algorithm for multi-objective flexible job shop scheduling," Information Sciences, vol. 289, pp. 76-90, 2014.
[43] J. Gómez, C. Gil, R. Baños, A. L. Márquez, F. G. Montoya, and M. Montoya, "A Pareto-based multi-objective evolutionary algorithm for automatic rule generation in network intrusion detection systems," Soft Computing, vol. 17, pp. 255-263, 2013.
[44] P. P. Wakabi-Waiswa and V. Baryamureeba, "Extraction of interesting association rules using genetic algorithms," International Journal of Computing and ICT Research, vol. 2, pp. 26-33, 2008.
[45] B. Minaei-Bidgoli, R. Barmaki, and M. Nasiri, "Mining numerical association rules via multi-objective genetic algorithms," Information Sciences, vol. 233, pp. 15-24, 2013.
[46] S. Walton, O. Hassan, K. Morgan, and M. Brown, "Modified cuckoo search: a new gradient free optimisation algorithm," Chaos, Solitons & Fractals, vol. 44, pp. 710-718, 2011.
[47] N. Bacanin, "Implementation and performance of an object-oriented software system for cuckoo search algorithm," International Journal of Mathematics and Computers in Simulation, vol. 6, pp. 185-193, 2012.
[48] R. Rajabioun, "Cuckoo optimization algorithm," Applied soft computing, vol. 11, pp. 5508-5518, 2011.
[49] A. H. Gandomi, X.-S. Yang, and A. H. Alavi, "Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems," Engineering with computers, vol. 29, pp. 17-35, 2013.
[50] X.-S. Yang and S. Deb, "Cuckoo search: recent advances and applications," Neural Computing and Applications, vol. 24, pp. 169-174, 2014.
[51] H. A. Guvenir and I. Uysal, "Bilkent university function approximation repository," ed, 2000.
[52] B. Alataş and E. Akin, "An efficient genetic algorithm for automated mining of both positive and negative quantitative association rules," Soft Computing, vol. 10, pp. 230-237, 2006.
[53] B. Alatas and E. Akin, "Rough particle swarm optimization and its applications in data mining," Soft Computing, vol. 12, pp. 1205-1218, 2008.

AUTHORS

Irene Kahvazadeh was born on September 1991. She passed her bachelor between 2009 to 2012 years. From February 2013, she is student in master degree at Tarbiat Modares University.

Mohammad Saniee Abadeh received his B.S. degree in Computer Engineering from Isfahan University of Technology, Isfahan, Iran, in 2001, the M.S. degree in Artificial Intelligence from Iran University of Science and Technology, Tehran, Iran, in 2003 and his Ph.D. degree in Artificial Intelligence at the Department of Computer Engineering in Sharif University of Technology, Tehran, Iran in February 2008. Dr. Saniee Abadeh is currently a faculty member at the Faculty of Electrical and Computer Engineering at Tarbiat Modares University. His research has focused on developing advanced meta-heuristic algorithms for data mining and knowledge discovery purposes. His interests include data mining, bio-inspired computing, computational intelligence, evolutionary algorithms, fuzzy genetic systems and memetic algorithms.