Memetic Comp. (2015) 7:93–104
DOI 10.1007/s12293-014-0147-5

REGULAR RESEARCH PAPER

An effective multiobjective approach for hard partitional clustering

Jay Prakash · P. K. Singh

Received: 27 February 2014 / Accepted: 15 November 2014 / Published online: 6 January 2015

Abstract Clustering is an unsupervised classification method in the field of data mining. Many population based evolutionary and swarm intelligence optimization methods are proposed to optimize clustering solutions globally based on a single selected objective function which lead to produce a single best solution. In this sense, optimized solution is biased towards a single objective, hence it is not equally well to the data set having clusters of different geometrical properties. Thus, clustering having multiple objectives should be naturally optimized through multiobjective optimization methods for capturing different properties of the data set. To achieve this clustering goal, many multiobjective population based optimization methods, e.g., multiobjective genetic algorithm, mutiobjective particle swarm optimization (MOPSO), are proposed to obtain diverse tradeoff solutions in the pareto-front. As single directional diversity mechanism in particle swarm optimization converges prematurely to local optima, this paper presents a two-stage diversity mechanism in MOPSO to improve its exploratory capabilities by incorporating crossover operator of the genetic algorithm. External archive is used to store non-dominated solutions, which is further utilized to find one best solution having highest F-measure value at the end of the run. Two conceptually orthogonal internal measures SSE and connectedness are used to estimate the clustering quality. Results demonstrate effectiveness of the proposed method over its competitors MOPSO, non-dominated sorting genetic algorithm, and multiobjective artificial bee colony on seven real data sets from UCI machine learning repository.

Keywords Data clustering · Multiobjective optimization · Evolutionary and swarm intelligence · Mutiobjective particle swarm optimization

J. Prakash (✉) · P. K. Singh
Computational Intelligence and Data Mining Research Lab., ABV-Indian Institute of Information Technology and Management, Gwalior, India
e-mail: jayprakash.iiitm@gmail.com

P. K. Singh
e-mail: pksingh@iiitm.ac.in

1 Introduction

Clustering, an important task in data mining, has been approached by many disciplines because of its wide applications such as biology, information retrieval, business, medicine, social science, earth science. In a clustering problem, objects of the data set are partitioned into appropriate number of clusters based on some similarity function. Consequently, objects sharing same cluster are more similar in comparison to the objects in the distinct clusters [31]. Clustering quality is often measured by an internal validity criteria, which can be based on different features of the clusters such as compactness, isolation, and connectedness. Conventionally, clustering algorithms are broadly classified as partitional, hierarchical, and density based algorithms [15]. The hierarchical methods organize the objects as a hierarchical tree structure where each level represents partitions of the data set. These methods do not require initialization of solutions and prior knowledge about number of clusters. However, in these methods, setting stopping criteria is very difficult as well as objects assigned to a cluster can not move to another cluster at later stage [29]. Density based clustering methods [12] implement a key idea that each object in a cluster should not contain number of objects in the neighborhood (in a defined radius) less than a defined threshold value. Though these methods are usually well suited to identify outliers, they find it difficult to partition the data set with overlapping clusters.

On the other hand, Partitional clustering methods directly decompose the data set into number of clusters. Partitional clustering can be fuzzy or hard [6]. Hard partitional clustering creates non-overlapping clusters by assigning each object into one cluster. However, in case of fuzzy partitional clustering [2], each object resides into every cluster with some membership weight. It is more suited to the data sets having overlapping clusters. Many traditional/conventional clustering algorithms, e.g., K-Means [18], K-medoids [26] have been proposed to solve the hard partitional clustering problem. However, these algorithms suffer from many drawbacks, e.g., they stuck into local optima, the quality of partitions is dependent on initial solution [23]. Therefore, a large number of nature-inspired population based global search optimization methods, e.g., evolutionary methods, swarm intelligence methods, have been proposed to overcome these deficiencies and enhance the quality of clustering solutions [11,17].

However, most of the research proposals available in the literature partition data sets on a single objective function. In this sense, optimized solution based on a single objective function is biased towards a particular criterion. As clustering problems usually consist of multiple conflicting objectives, various researchers [14,15,28] suggest that quality of clustering solution should be evaluated by multiobjective optimization methods to reduce dominance of a particular objective on the results of clustering and it should be treated as a natural way to partition the data sets. Such methods obtain a set of non-dominated solutions known as pareto-front, where each solution is a trade-off of the conflicting objectives and no solution in this front is inferior to the other solutions. Finally, the decision maker picks a solution as the promising solution based on his/her requirement.

The multiobjective optimization methods simultaneously optimize a number of conflicting objectives. Over the years, many classical (non nature-inspired computing) multiobjective methods, e.g., weighted sum methods [24], Benson's method [10], are proposed to solve many real-world problems involving multiple conflicting objectives. Though these methods are easy to implement, they suffer from some inherent drawbacks, e.g., optimal solutions depend on the chosen initial solution, they tend to get stuck to a suboptimal solution, they are not efficient in handling problems having discrete search space, they can not be used efficiently on parallel machines [7]. To overcome these drawbacks, many multiobjective evolutionary algorithms [15,32] and multiobjective swarm intelligence methods [1,4] are proposed to obtain trade-off solutions in the pareto-front. In [4], the authors propose mutiobjective particle swarm optimization (MOPSO) to solve multiobjective problems using PSO [21] and apply it on several benchmark test functions. They show that it is highly competitive over its competitors NSGA II, microGA, and PAES. However, we observe that there is no information sharing mechanism among particles as a swarm (in group); each particle is influenced by its own experience pbest, known as cognitive component, and the leader of the swarm gbest, known as social component. Here, we present a novel method TSMPSO to improve the diversity mechanism (to build such an information sharing mechanism among particles) in MOPSO by introducing crossover operator of GA. Here, TSMPSO along with three competing algorithms MOPSO, elitist non-dominated sorting genetic Algorithm [9] (NSGA-II), and Multiobjective Artificial Bee Colony [1] (MABC) are applied to clustering application. Two conflicting clustering objectives SSE [30] and connectedness [16] are optimized to obtain non-dominated solutions on variety of real data sets having different cluster characteristics, e.g., size, shape, density. Here, external archive (which is dynamic but limited in size) stores the non-dominated solutions as the search progresses and is used to guide the particle's flight in swarm during iteration and further to find a best solution having highest F-measure value at the end of the run. We experiment with seven real data sets from UCI machine learning repository. The results show that TSMPSO is superior over the competing methods on all the data sets.

Rest of the paper is organized as follows. Section 2 presents a brief literature review. Section 3 presents the algorithmic background of competing methods. Our proposed method TSMPSO is detailed in Sect. 4. Experimental set up, results and discussion are included in Sect. 5. Finally, Sect. 6 concludes with future scopes in the field of data clustering.

2 Related work

A brief survey of the multiobjective evolutionary algorithms and multiobjective particle swarm intelligence methods and applications can be found in [33] and [27] respectively. As mentioned above, clustering usually consists of conflicting objectives; few researchers approach the problem in multiobjective framework to find the best clustering solutions corresponding to trade-off of the objectives. In this context, an evolutionary multiobjective optimization method [14] attempts to improve the accuracy of clustering using two conflicting objective functions overall deviation and connectedness when the number of clusters in the data set is known. It adopts PESA-II [5] for clustering by incorporating a specialized mutation and an initialization procedure. A well-known multiobjective hard clustering method, Multiobjective Clustering with automatic k-determination (MOCK) [15] consists of two main phases. In the first phase, the initial clustering phase, PESA-II [5] is used to optimize two conflicting objectives as mentioned in [14] to produce a set of trade-off solutions. For encoding, it uses a locus-based adjacency representation. This representation encodes clustering along with number of clusters. The initial population is filled up using two clustering algorithms minimum spanning tree

(MST) based algorithm and K-means. Half of the population is filled up using the MST-based algorithm. Remaining half is filled up by K-means algorithm for different number of clusters. It uses uniform crossover operator to provide diversity in search space and neighborhood-biased mutation operator to make the convergence faster. At the end, this phase generates a set of non-dominated solutions. In second phase, the model selection phase, the number of clusters is determined based on selecting a particular solution from the pareto-front. This particular solution is selected based on the shape of the pareto-front (knee) by the Gap statistics. In [16], four different pairs of clustering criteria are evaluated and analyzed in multiobjective framework with the same encoding, variation operators, initialization and parameter setting as in [15]. It has been shown that pair of objectives employed in MOCK acts as strongest combination. In [13], the corrected Rand based selection strategies are used to reduce a high number of partitions generated in multiobjective framework. These strategies significantly reduce the number of solutions (partitions) by maintaining diversity in search space and preserving quality solutions. We propose a novel method TSMPSO which outperforms its competitors for hard partitional clustering. Performance assessment is carried out in terms of classification accuracy, distribution of solutions in the obtained pareto-front, convergence of the obtained pareto-front with respect to the reference pareto-front, and number of solutions in the obtained pareto-front.

3 Algorithm background

In this section, we describe the MOPSO [4], NSGA II [9], and MABC [1] in brief.

3.1 MOPSO

PSO is a population-based metaheuristic search algorithm for single objective optimization where every particle in the swarm is a potential solution. Particle's flight is influenced by its personal best position and the global best position in the swarm [21]. The velocity (v) and position (x) of a particle is updated based on Eqs. 1 and 2, respectively.

$$v_{bd}(t+1) = v_{bd}(t) + c_1.r_{1d}(y_{bd}(t) - x_{bd}(t)) + c_2.r_{2d}(\widehat{y}_d(t) - x_{bd}(t)) \quad (1)$$

$$x_{bd}(t+1) = x_{bd}(t) + v_{bd}(t+1) \quad (2)$$

where d is dth dimension of a particle; b is bth particle in the swarm; y is local best of a particle ; $\widehat{y}$ is the global best in the swarm; t is the iteration; r_1 and r_2 are random numbers in the interval (0, 1), and c_1 and c_2 are positive acceleration constants. It requires following changes to solve a multiobjective optimization problem [27].

- Selection criteria of the personal best (pbest) for a particle and the global best (gbest) in the swarm.
- Strategy to retain all non-dominated solutions found during the run, e.g., external archive.
- Diversity mechanism in the search process to escape convergence to a single solution or sub-optimal solutions.

Algorithm 1 demonstrates a step by step procedure of MOPSO to solve a multiobjective problem. MOPSO changes PSO to deal with multiobjective optimization problems. When particle's pbest is dominated by particle's current position, it is replaced by particle's current position. If neither of them is dominated by each other, then we select one of them randomly. MOPSO uses external repository to store non-dominated solutions obtained in each iteration and later leader selected from this repository guides the solutions towards pareto-front by maintaining diversity. Concept of this archive is similar to the adaptive grid of Pareto archive evolutionary strategy (PAES) [22]. PSO is mainly suitable for multiobjective optimization because of the high speed of convergence as long as premature convergence does not occur. Since single information sharing mechanism in PSO among the particles may easily lead to convergence to the local optimum (i.e., premature convergence) in multiobjective optimization, appropriate promotion of diversity is an important issue.

Algorithm 1: procedure of MOPSO

```
1:  Initialize a swarm;
2:  Evaluate each particle in the swarm;
3:  Store a copy of non-dominated solutions in the external archive;
4:  Initialize particles pbest to its initial position.;
5:  while stopping criteria is not satisfied do
6:    Select leader (gbest) from external archive;
7:    for each particle do
8:      Update velocity and position as Eqs. (1) and (2);
9:      Evaluate each of the particle in the swarm;
10:     Update pbest of each particle;
11:   end for
12:   Update external archive by current non-dominated solutions;
13: end while
14: Generate result from external archive;
```

3.2 NSGA II

NSGA II [9] is a multiobjective optimization evolutionary algorithm based on elitism and diversity preservation mechanism. Initially, parent population PP_t (size N) is generated randomly based on a particular criteria. Then offspring population OP_t is generated by selecting parents from parents population PP_t using crowded binary tournament selection and other genetic operators. Further, the offspring population OP_t (size M) is combined with the parent population PP_t (size N) making the entire population EP_t of size $M+N$. Afterwards, non-dominated fronts and crowding distance (a measure of density of solutions in the neighborhood) of solutions are calculated on EP_t. Later, N solutions out of $M+N$ are promoted to the next generation based on better rank or higher crowding distance on the same rank. Consequently, at the end of the run, we obtain a set of non-dominated solutions on best pareto-front (solutions with the best rank) in the final population. We outline NSGAII in Algorithm 2.

Algorithm 2: NSGA II algorithm

```
1: Initialize chromosomes in the population;
2: while Termination criteria is not satisfied do
3:     Evaluate solutions in the population based on all objectives;
4:     Rank the solutions as well as evaluate their crowding distance;
5:     Generate offspring population $OP_t$ by selecting parents from
       parent population $PP_t$ using genetic operators;
6:     Evaluate solutions in the offspring population based on all
       objectives;
7:     Combine $OP_t$ and $PP_t$ to make $EP_t$;
8:     Rank the solutions in $EP_t$ as well as evaluate their crowding
       distance;
9:     Select new population $PP_{t+1}$ from top sorted solutions in $EP_t$
       based on better rank or higher crowding distance on same rank;
10: end while
11: Output: set of non-dominated solutions on the best pareto front
    in the final population;
```

NSGA II uses explicit diversity preservation mechanism crowding distance to diversify solutions on the non-dominated front and elitism to preserve already obtained non-dominated solutions to the next generation.

3.3 MABC

ABC algorithm is a swarm intelligence population based optimization algorithm introduced by Karboga [19] where potential solutions are food sources of honey bees. The algorithm design is inspired by intelligent food foraging behavior of real honey bees. The quality of solutions is evaluated based on values (nectar amount) of food source in terms of two objectives SSE and connectedness. In a natural bee swarm, there are three types of honey bees: employed bees, onlooker bees, and scout bees. The employed bees search new food sources, which dominate the food sources in their memory. The onlooker bees gather information from employed bees to select food sources which dominate larger number of food sources and further search for better quality food sources in neighborhood of the selected food sources. The scout bees discover food sources randomly for the exhausted food sources. Here, MABC [1] changes ABC to deal with multi-objective optimization problems. Three phases of MABC to search new food sources are described as follows:

– Employed bees phase: In employed bees phase, an employed bee searches for the food source in the neighborhood, which dominates the food source in her memory. The position of new food source with respect to i_{th} food source is obtained by following Eq. 3.

$$x'_{ij} = x_{ij} + w1 \star \phi_{ij}(x_{ij} - x_{pj}) \tag{3}$$

where p is a food source randomly selected from the archive ; N denotes number of food source; $j \in \{1, 2, \ldots, d\}$ is a randomly selected index; ϕ_{ij} is a random number between [0,1]; the coefficient w1 controls the importance of p in search of new food source. If the new obtained food source in the neighborhood is better than the food source in her memory, then the bee updates her position with new one. Here, the number of employed bees is equal to the number of food sources.

– Onlooker bees phase: In onlooker bees phase, all the employed bees provide information about the quality of new food sources to the onlooker bees. Then onlooker bees choose their food sources based on probability prob_i related to fitness of the obtained solutions in the employed bees phase. In MABC, prob_i of food source i is evaluated using Eq. 4.

$$\text{prob}_i = \frac{\text{fit}_i}{\sum_{i=1}^{N} \text{fit}_i} \tag{4}$$

where fit_i is the fitness value of food source i; $\text{fit}_i = \frac{\text{dom}(i)}{N}$; $\text{dom}(i)$ is number of food sources dominated by i; N represents the number of food sources. Each onlooker bee modifies the solution in her memory similar to the employed bee based on prob_i values. However, an another coefficient parameter w2 controls the importance of neighbour p which is selected randomly from the swarm. If the obtained solution dominates previous one, bee memorizes non-dominated solution and forget the previous one.

– Scout Bees Phase: In the previous two phases of ABC, it is found that the position of some food sources is not upgraded in predetermined number of trials. Such food sources are called abandoned solutions. At this point, scout bees search new solution x_i randomly as shown in Eq. 5 for each abandoned solution.

$$x_{ij} = x_{\min,d} + \text{rand}(0,1)(x_{\max,d} - x_{\min,d}) \tag{5}$$

where d is dimension of the data set.

Algorithm 3 demonstrates a step by step procedure of MABC [1] to solve a multiobjective problem.

Algorithm 3: procedure of MABC

```
1: Initialize food sources (solutions);
2: Evaluate each food source ;
3: Store a copy of non-dominated food sources in the external archive;
4: while stopping criteria is not satisfied do
5:   Employed Bees Phase for searching new food sources using Eq. 3;
6:   Calculate fitness probability of food sources using Eq. 4;
7:   Onlooker Bees Phase for searching new food sources based on fitness probability;
8:   Scout Bees Phase for re-generating abandoned solution using Eq. 5;
9:   Update external archive by current non-dominated solutions;
10: end while
11: Generate result from external archive;
```

MABC uses grid based approach to maintain non-dominated solutions in an external archive after performing three phases Of MABC.

4 Proposed methods

In this section, we present a description of the proposed algorithm TSMPSO.

4.1 Solution representation and initialization

We follow a centroid based representation to represent a particle (candidate solution) as described in [23]. Here, every candidate solution contains $k \times$ nvar dimensions, where k denotes the number of clusters, which is known a priori and nvar indicates the number of dimensions in the data points. Every dimension of a cluster is assigned a random number between maximum ($x_{\max}$) and minimum ($x_{\min}$) value of the corresponding dimension in the data set. The position(n, d) represents dth dimension of the nth cluster, and rand() is uniformly distributed random number in the range of [0,1]. Algorithm 4 describes steps of particle initialization.

Algorithm 4: particle initialization

```
1: $k \leftarrow$ number of clusters;
2: nvar $\leftarrow$ number of dimensions;
3: for $n = 1 : k$ do
4:   for $d = 1$ : nvar do
5:     position$(n, d) = x_{\min}(d) + \text{rand}() * (x_{\max}(d) - x_{\min}(d))$;
6:   end for
7: end for
```

Note that initial velocities of particles in the PSO are assigned to zero. When new solutions are generated, dimensions of the new solutions must satisfy the dimensional boundary condition of search space. It means that if the value of a dimension of a new solution crosses the value of upper bound of corresponding dimension in the search space then that value is replaced by the value of the upper bound. Similar condition is applied for the lower bound also. As the paper deals with the clustering problem where k is known a priori, a solution must contain k clusters. However, it is possible that a newly generated solution does not contain valid k clusters. Such solution is regarded as erroneous solution . Here, the erroneous solution is replaced by a randomly generated new solution as per the procedure described in Algorithm 4.

4.2 Fitness functions

Since single validity measure is not equally appropriate for the data set of different characteristics, simultaneous optimization of multiple objectives naturally improves the results. Therefore, we select a pair of conflicting objectives intra cluster variance (SSE) [30] and connectedness [15] based on comparative study shown in [16]. First objective SSE, shown in Eq. 6, is based on the idea that each object should belong to a nearest cluster centre and quality of solution should be evaluated in context of compactness of clusters. It is effective for the data set having well-separated and spherical clusters.

$$SSE(S) = \sum_{i=1}^{k} \sum_{\forall x_p \in c_i} \|x_p - m_i\|^2 \tag{6}$$

Here S is a solution, x_p indicates pth object of the data set, m_i implies centroid of the ith cluster, and c_i denotes the ith cluster. However, the second objective connectedness, shown in Eq. 7, is based on the idea that objects in the neighbourhood should belong to the same cluster. It is more compatible to identify clusters of arbitrary shapes [15].

$$\text{Connectedness}(S) = \sum_{p=1}^{N} \left(\sum_{j=1}^{T} x_{p,nn_p(j)} \right) \tag{7}$$

where S denotes a solution, p indicates pth object in data set, N indicates number of objects in the data set, $nn_p(j)$ indicates jth nearest neighbor of object p, and $x_{p,nn_p(j)}$ indicates penalty received with value (1/j) when jth nearest neighbor data point is not in the same cluster as p. Otherwise, penalty value received for that nearest neighbor data point is zero. In other words, for each data point p, amount of penalty is equal to sum of penalty received with value (1/j) due to each nearest neighbor j when p and j are not in the same cluster; T is a parameter which indicates the number of nearest neighbours to utilize; it can be seen that penalty value received gradually decreases for distant neighbor.

Consequently, by these two objective functions, homogeneity of the clusters is measured by SSE and separation between the clusters is measured implicitly by connectivity measure.

4.3 TSMPSO

TSMPSO algorithm is based on pareto based elitist approach [7]. For initialization of swarm POP (size NP), particle velocities are initialized to zero and personal best (pbest) of each particle is initialized to its initial position. Obtained non-dominated solutions in the swarm are preserved in external archive. It serves two purposes; first, non-dominated solutions are preserved as search progresses, and second, an appropriate global best (leader) particle is selected from this archive that guides the swarm during the search process. To select the leader and maintain the external archive, hypercubes [4] are created in the objective space where solutions in external archive are maintained as coordinate system. Location of these solutions in hypercubes are determined according to values of their objective functions. Further, fitness assignment of each hypercube containing more than one solution is performed. The assigned fitness of hypercube equals to the result of dividing any number $x > 1$ (Here, we selected $x = 10$ as recommended in [4]) by number of solutions in respective hypercube. This process lead to decrease the fitness of hypercube having more number of solutions. Roulette-wheel selection is performed to select the hypercube based on associated fitness. A leader is selected randomly from solutions residing in selected hypercube.

In first-stage of diversity mechanism in TSMPSO, changes in particle's velocity and position are performed on current swarm as per Eqs. 1 and 2 respectively to create a temporary swarm, say $POP1$. Then particle pbest is updated to its current position except if its pbest dominates its current position in order to provide better diversity to solutions [3]. Since particles are influenced by their pbest and the leader of swarm, no information sharing mechanism is available among all particles in the swarm except that each particle can obtain information from the leader. Therefore, in order to build such an information sharing mechanism among particles to improve diversity in search process, in second-stage a genetic crossover operator is applied on the swarm to build such an information sharing mechanism among individuals. A single point crossover is performed with some crossover probability P_c by selecting two parents randomly from the current swarm to generate offsprings population by exchanging segments beyond crossover point. To perform crossover, a random integer number is generated in the range of 1 to $k * \text{nvar} - 1$, where k denotes number of cluster centroids and nvar indicates the number of dimensions in a centroid. Here, we denote generated offsprings population as $POP2$. This additional diversity mechanism is helpful to enhance the exploration capability of MOPSO and leads to diversify solutions in the search space.

Now, the $POP1$ and $POP2$ are combined to create temporary population ($TEMP\_POP$). For $TEMP\_POP$, different pareto fronts are created based on dominance criteria [7]. New population is filled by starting the best non-dominated front followed by the second, and so on. When last-allowed front is considered, the remaining slots in the new population may be less than the solutions on this front. The crowding distance (CD) is calculated on last-allowed front. It is one of the niching method to calculate density of solution around an individual solution on a non-dominated front, which sums the normalized distance of two neighbour solutions $p - 1$ and $p + 1$ of a solution p in objective space along each of the objectives. The least crowded solution on the last-allowed front is given priority like NSGA II [9]. Finally, external archive is updated with respect to the non-dominated particles obtained in the new swarm. Existing solution in the archive is removed if it is dominated by any current non-dominate solution. Whenever external archive becomes full, extra solutions are removed based on giving priority to solutions residing in less populated regions [4]. Accordingly, places of solutions within hypercubes are redetermined. The pseudo code of the TSMPSO is shown in Algorithm 5 and the flowchart is presented in Fig. 1.

Algorithm 5: TSMPSO algorithm

```
1:  Initialize the swarm ($POP$) of size $NP$;
2:  for each particle in the swarm do
3:      Evaluate fitness function;
4:      Initialize particle's pbest to its initial position;
5:  end for
6:  Update external archive with respect to the non-dominated
    solution from $POP$;
7:  while max$it$ is not reached do
8:      Select leader from external archive;
9:      for each particle in the swarm do
10:         update velocity and position;
11:         Evaluate fitness function;
12:         update pbest;
13:     end for// The new swarm is termed as $POP1$ //
14:     perform genetic crossover over $POP$ to create offspring
        population ($POP2$);
15:     Evaluate each solution in $POP2$;
16:     $TEMP\_POP \leftarrow POP1 \bigcup POP2$;
17:     Create non-dominated pareto fronts for $TEM\_POP$;
18:     Find crowding distance ($CD$) of solutions on last-allowed
        front;
19:     Promote $NP$ better solutions to new swarm ($POP$);
20:     Update external archive with respect to the best non-dominated
        front;
21: end while
```

5 Experimental results and discussions

We perform experiments on TSMPSO along with three well-know multiobjective optimization methods MOPSO [4], NSGA II [9], MABC [1] to optimize two conflicting crite-

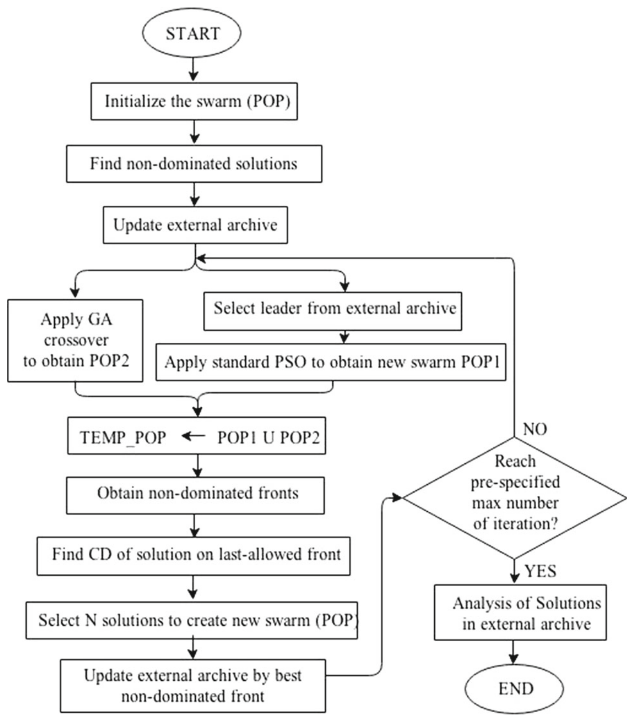

START
Initialize the swarm (POP)
Find non-dominated solutions
Update external archive
Apply GA crossover to obtain POP2
Select leader from external archive
Apply standard PSO to obtain new swarm POP1
TEMP_POP ← POP1 U POP2
Obtain non-dominated fronts
Find CD of solution on last-allowed front
Select N solutions to create new swarm (POP)
Update external archive by best non-dominated front
Reach pre-specified max number of iteration?
NO
YES
Analysis of Solutions in external archive
END

Fig. 1 Flow chart of proposed algorithm

ria intra cluster distance and connectedness simultaneously. The obtained results demonstrate that the proposed algorithm is highly competitive with competing algorithms over seven real data sets from the UCI machine learning repository.[1]

5.1 Parameters setup

As results of the population based optimization methods are influenced by parameters settings, these values should be selected carefully based on empirical and/or theoretical evidence. We choose population size 100 as higher value increases the computational overhead. Based on the experiments, the maximum number of fitness function evaluation selected as stopping criteria is 40,000 as performance of algorithms is not considerably improved afterwards. Parameter settings of competing algorithms which are applied in clustering domain are similar to their original research papers. We select the value of T parameter based on recommended range given in [15]. Further, the value of Number of hypercubes (divisions for adaptive grid) and size of external archive are chosen based on the recommended value in [4]. The value of inertia weight w, cognitive learning factor $c1$ and social learning factor $c2$ are chosen as 0.71, 1.49, 1.49 respectively as this set produce the best results overall among different sets of values. The Table 1 presents all parameter settings for the proposed as well as the competing methods.

[1] http://archive.ics.uci.edu/ml/.

Table 1 Control parameters for TSMPSO, MOPSO, and NSGA II

Name of parameters	Value
Population size	100
Maximum number of fitness function evaluation	40,000
Number of independent runs	30
Size of external archive	100
T for connectedness	10
C1	1.49
C2	1.49
w	0.72
Number of hypercubes	30
Crossover rate of GA (P_c)	1

Table 2 Datasets descriptions

Name of data set	Number of clusters	Number of dimensions	Number of instances
Iris	3	4	150
Glass	6	9	214
Vowel	6	3	871
WBC	2	9	683
Wine	3	13	178
Zoo	7	16	101
Dermatology	6	34	358

5.2 Datasets

The datasets are in matrix of size $n \times$ nvar with real-valued elements, where n is number of objects and nvar is number of features. These objects are partitioned into k non-overlapping clusters. In this work, we perform experiment on seven real datasets Iris, Glass, Vowel, Wisconsin Breast cancer (WBC), Wine, Zoo, and Dermatology from the UCI machine learning repository to validate performance of the TSMPSO. A brief characteristic of these datasets is presented in Table 2. In WBC, we use only 683 samples out of originally 699 samples by removing 16 samples having some missing features.

5.3 F-measure

We use F-measure to judge accuracy of the obtained clusters. F-measure [28] is a balanced measure, which is evaluated by combining precision and recall as harmonic mean. Precision is the fraction of retrieved objects that are relevant, while recall is the fraction of relevant objects that are retrieved. F-measure of a cluster with respect to known class can be mathematically expressed as Eq. 8.

$$F - \text{measure} = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \quad (8)$$

where, precision $= m_{pq}/m_p$, m_{pq} is the number of objects which belong to cluster p and class q both, and m_p is the total number of objects in cluster p; recall $= m_{pq}/m_q$, m_q is the total number of objects in class q. Optimum value of F-measure is 1.

5.4 Performance assessment matrices

Generally, following issues are considered to assess algorithms in order to perform quantitative assessment in the multiobjective scenario [35].

1. Convergence of solutions: solutions should be as close as possible towards optimal pareto front. It is assessed based on minimized distance between obtained pareto-front of an algorithm and optimal pareto-front.
2. Diversity of solutions: solutions in the pareto-front should be distributed in such a way that they represent a set of solutions covering the entire pareto-optimal region uniformly.
3. Number of solutions: number of solutions should be as maximum as possible on non-dominated pareto front.

Therefore, we assess performance of the algorithms using following commonly used performance assessment matrices as no single metric can judge performance of an algorithm in complete sense [7].

– Coverage (C) [34]: this metric is used to compare relative quality of two sets of solutions based on domination. For two pareto-fronts S_1 and S_2, $C(S_1, S_2)$ evaluates the proportion of solutions in S_2, which are dominated by solutions in S_1. It can be mathematically defined as shown in equation 9.

$$C(S_1, S_2) = \frac{|\{a_2 \in S_2; \exists a_1 \in S_1 : a_1 \preceq a_2\}|}{|S_2|} \tag{9}$$

Value of C lies between 0 and 1. A higher value of C indicates a better dominance. $C(S_1, S_2) = 1$ means each solution of S_2 is dominated by at least one solution of S_1. On the other hand, $C(S_1, S_2) = 0$ means no solution of S_2 is dominated by any solution of S_1.

– Distribution $(\triangle)$ [9]: this metric provides useful information about distribution (spread) on obtained non-dominated front (S) of an algorithm. It is calculated with a relative distance measure between consecutive solutions in S. It is mathematically described as shown in Eq. 10.

$$\triangle(S) = \sum_{i=1}^{|S|-1} \frac{|d_i - \bar{d}|}{|S| - 1} \tag{10}$$

Here, for each solution i, $d_i = \sum_{m=1}^{M} |{f_m}^{i+1} - {f_m}^{i}|$ is the sum of absolute difference in normalized objective function values between consecutive solutions ith and $(i+1)$th in (S), which is sorted in order in each objective; $\bar{d}$ is the mean of d_i. Smaller value of $\triangle$ indicates better uniformly distributed solutions where ideal value of $\triangle$ is zero which indicates that the solutions on S are uniformly distributed.

– Convergence [8]: this matric assess convergence of the non-dominated solution set of an algorithm with respect to a reference set P^*. It is mathematically defined as shown in Eq. 11.

$$\text{Convergence}(S) = \frac{\sum_{i=1}^{|S|} d_i}{|S|} \tag{11}$$

where, $d_i = \min_{j=1}^{p^*} \sqrt{\sum_{k=1}^{M} \frac{f_k(i) - f_k(j)}{f_k^{\max} - f_k^{\min}}}$

Here, $f_k^{\max}$ and $f_k^{\min}$ are the minimum and maximum objective function values respectively for kth objective function in P^*. Smaller value of this measure indicates superiority of the solution set. Ideal value of this measure is zero, which indicates all solution set lie on reference (optimal) pareto front.

– ONVG [25]: overall Non-dominat Vector Generation (ONVG) is the count of solutions on obtained non-dominated pareto front. It is mathematically defined as shown in Eq. 12.

$$ONVG(S) = |S| \tag{12}$$

5.5 Comparison of results

We compare performance of proposed algorithm TSMPSO with three well known algorithms MOPSO [4], NSGA II [9], and MABC [1] for crisp clustering on all the experimental data sets. Initially, performance of these algorithms is measured based on the best solution in obtained solution set which has the highest F-measure value. Table 3 shows the mean and standard deviation (shown in bracket) of the highest F-measure values achieved by solution sets in different runs. It is clearly visible that TSMPSO reveals its superiority over competing algorithms in term of mean classification accuracy of F-measure. Moreover, TSMPSO reports the smallest standard deviation in term of Glass, WBC, and Zoo data sets which demonstrates robustness of the algorithm. In case of Wine data set, mean accuracy achieved by these algorithms are close to each other. MOPSO demonstrates better mean accuracy than NSGA II in term of Iris , WBC, and Zoo data sets. Moreover, it achieves smaller standard deviation of accuracy over all the data sets except Glass data set. This indicates robustness of MOPSO with respect to

Table 3 Performance of algorithms is based on F-measure on different data sets

Data Sets	TSMPSO	MOPSO	NSGA II	MABC
Iris	0.9265 (0.0182)	0.9089 (0.0142)	0.8903 (0.0354)	0.8604 (0.0546)
Glass	0.5584 (0.0144)	0.5441 (0.0265)	0.5549 (0.0161)	0.5017 (0.0494)
Vowel	0.6317 (0.0255)	0.6109 (0.0223)	0.6087 (0.0231)	0.6022 (0.0269)
WBC	0.9741 (0.0010)	0.9735 (0.0025)	0.9514 (0.0201)	0.9702 (0.0028)
Wine	0.7294 (0.0037)	0.7272 (0.0001)	0.7290 (0.0033)	0.7271 (0.0095)
Zoo	0.8131 (0.0162)	0.8011 (0.0184)	0.7852 (0.0220)	0.7904 (0.0245)
Dermatology	0.3656 (0.0163)	0.3580 (0.0133)	0.3584 (0.0194)	0.3618 (0.0071)

Average results with standard deviation are presented based on 30 run

Table 4 Coverage for obtained non-dominated solutions from combined pool of sets of solutions in all run

Datasets	C(A, B)	C(A, C)	C(A, D)	C(B, A)	C(C, A)	C(D, A)
Iris	1	1	1	0	0	0
Glass	1	1	1	0	0	0
Vowel	0.8	0.987	1	0.0576	0	0
WBC	1	1	1	0	0	0
Wine	1	1	0.8	0	0	0.1111
Zoo	1	0.8421	1	0	0.1875	0
Dermatology	1	0.58	1	0	0.6315	0

Table 5 Average results of Distribution with standard deviation of obtained solutions sets are presented based on all runs

Datasets	TSMPSO	MOPSO	NSGA II	MABC
Iris	0.1988 (0.0614)	0.2753 (0.1537)	0.3092 (0.1374)	0.3745 (0.2972)
Glass	0.0606 (0.020)	0.19 (0.3221)	0.0641 (0.021)	0.2083 (0.124)
Vowel	0.0712 (0.0225)	0.0308 (0.0093)	0.03107 (0.0089)	0.1548 (0.0578)
WBC	0.1598 (0.0692)	0.1468 (0.3488)	0.1223 (0.032)	0.3812 (0.2355)
Wine	0.2392 (0.0486)	0.4454 (0.1904)	0.547 (0.3347)	0.2602 (0.0915)
Zoo	0.1661 (0.1268)	0.4326 (0.4433)	0.1894 (0.1018)	0.3471 (0.3709)
Dermatology	0.0973 (0.0233)	0.0374 (0.0089)	0.0574 (0.016)	0.12 (0.0345)

NSGA II. Further, we selected different matrices to quantitatively assess performance of algorithms in terms of diversity and convergence. Table 4 presents quantitative comparison of coverage metric (C) obtained pareto-fronts of algorithms from combined pool of sets of solutions in 30 independent runs. Symbols A, B, C, and D indicate TSMPSO, MOPSO, NSGA II, and MABC respectively. The values of $C(A, B)$, $C(A, C)$, and $C(A, D)$ are 1 in most of the data sets, which clearly show that all solutions of competitive algorithms are dominated by at least one solution of TSMPSO. On the other hand, the values of $C(B, A)$, $C(C, A)$, and $C(D, A)$ are 0 in most of the datasets which indicate that no solution of TSMPSO is dominated by any solution of competitive algorithms. Table 5 shows mean and standard deviation (shown in bracket) of distribution ($\triangle$) achieved by obtained sets of solutions in different runs. TSMPSO reports the smallest mean and standard deviation for Iris, Glass, Wine, and Zoo data sets, which indicates that the solutions are closer to a uniform distribution along the pareto-front and algorithm is more robust. Furthermore, Table 6 presents distribution ($\triangle$) for obtained non-dominated solutions from combined pool of sets of solutions in all runs. TSMPSO reports the smallest value over all the data sets with single exception that MOPSO has the smallest value for Wine data set. This indicates that $TSMPSO$ is able to find a distribution of solutions closer to a uniform distribution along the pareto-front. Table 7 presents quantitative comparison of convergence metric for obtained pareto-fronts of algorithms from combined pool of sets of solutions in all runs. TSMPSO reports the smallest value over all the data sets along with zero value for Iris, Glass, and WBC data sets. This indicates that TSMPSO produces solutions that are more closer to the reference pareto-front

Table 6 Distribution for obtained non-dominated solutions from combined pool of sets of solutions in all runs

Datasets	TSMPSO	MOPSO	NSGA II	MABC
Iris	0.1426	0.3619	0.4088	0.3486
Glass	0.1063	0.1644	0.129	0.2188
Vowel	0.1371	0.1788	0.1375	0.1887
WBC	0.1346	0.0756	0.0679	0.1606
Wine	0.0937	0.0605	0.1035	0.3774
Zoo	0.112	0.1969	0.115	0.6923
Dermatology	0.1416	0.0942	0.2213	0.2292

Table 7 Convergence for obtained non-dominated solutions from combined pool of sets of solutions in all runs

Datasets	TSMPSO	MOPSO	NSGA II	MABC
Iris	0	0.1007	0.1235	0.2728
Glass	0	0.2109	0.1506	1.2061
Vowel	0.0003	0.0387	0.0816	0.0865
WBC	0	0.0825	0.084	0.2921
Wine	0.0078	0.0239	0.0196	0.0009
Zoo	0.0236	0.1778	0.0411	0.0964
Dermatology	0.0165	0.0381	0.0277	0.049

Table 8 ONVG for obtained non-dominated solutions from combined pool of sets of solutions in all runs

Datasets	TSMPSO	MOPSO	NSGA II	MABC
Iris	7	7	5	4
Glass	32	13	28	14
Vowel	52	55	77	21
WBC	26	19	21	7
Wine	9	7	7	5
Zoo	16	15	16	5
Dermatology	38	75	50	16

over all the data sets. However, other methods have difficulties in convergence towards optimal pareto-front. Table 8 shows the values of ONVG of algorithms for the pareto-fronts obtained from the combined pool of sets of solutions in all the runs. TSMPSO reports the highest values of ONVG among the competing algorithms over all the data sets with exception that MOPSO and NSGA II have the largest value for Vowel and Dermatology data sets, respectively. This means that TSMPSO has lager number of solutions on obtained non-dominated pareto-front for most of the data sets.

Based on performance exhibited by MOPSO, it can be analyzed that solution achieved by MOPSO does not properly approach to the optimal pareto-front. This indicates that MOPSO does not properly diversify the solution in global search space because PSO suffer from well known

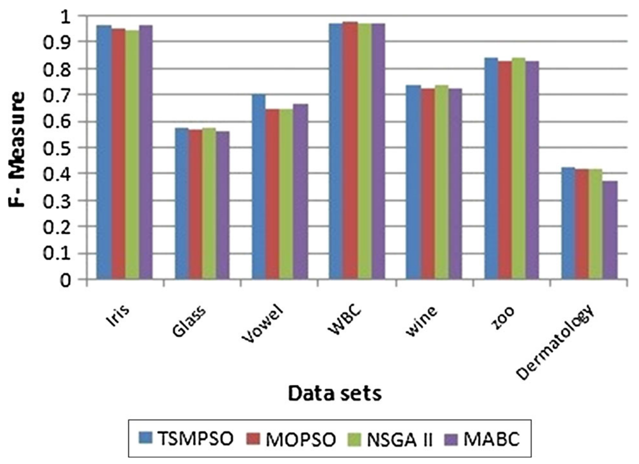
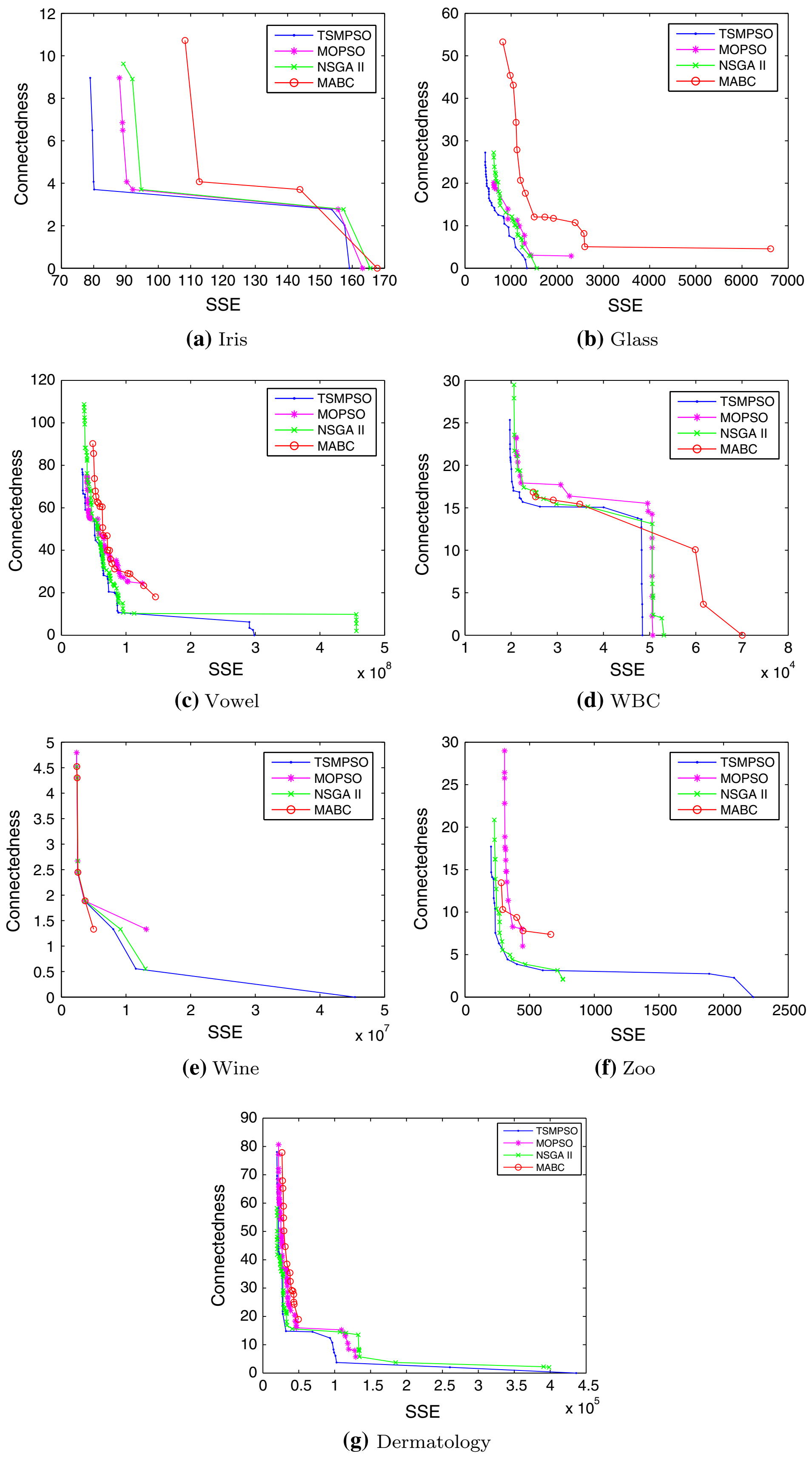


Fig. 2 *Bar chart* shows the comparison of the best F-measure value obtained in 30 runs

stability problem as well as information sharing mechanism is single directional among particles. In NSGA II, genetic crossover is a good way to provide diversity in search space by establishing information sharing mechanism among solutions. Therefore, performance of this algorithm is supposed to be better. However, experimentally it has been shown that NSGA II does not achieve better solutions over the proposed algorithm. Such type of behavior demonstrates that good solutions promoted in next generation are not sufficiently explored in search space by only applying genetic crossover operation. It concludes that it requires better balance of exploration and exploitation in the search space. It can also be observed from empirical results that MABC does not compete with the proposed algorithm over these data sets. Because in the search procedure of ABC, bees exchange only one dimension with a random neighbor, which slow the optimization ability of algorithm when dimension of problem increases [20]. It concludes that it requires sufficient exploration in search space.

As TSMPSO incorporates exploration through GA and exploitation through PSO, it performs better; experimental results also support its superiority over the competitors. Figure 2 shows comparison of the best F-measure value achieved by algorithms in 30 independent runs over all the data sets. It also supports the superiority of the proposed method over competing methods. Further, graphical representation of pareto-fronts obtained by the competing algorithms are shown in Fig. 3 for different data sets, which clearly shows that solutions of TSMPSO on obtained pareto-fronts are closer to uniform distribution with larger extent. In addition, solutions are closer to the reference optimal pareto-fronts. The above validates superiority of the TSMPSO over the other competing algorithms in terms of distribution, convergence, and number of solutions as reported quantitatively by different metrics (refer Tables 4, 5, 6, 7 8). In case of Dermatology data set, based on F-measure values, all four algo-

Fig. 3 Pareto front for non-dominated solutions from combined pool of obtained solutions in all independent runs. **a** Iris; **b** glass; **c** vowel; **d** WBC; **e** wine; **f** zoo; **g** dermatology



rithms perform highly unsatisfactory as it is a high dimensional data set containing 34 dimensions. Even on this data set, proposed method demonstrate better performance over other three methods. Overall, TSMPSO is a good alternative to solve crisp clustering problems through multiobjective approach to produce high quality trade-off solutions.

6 Conclusions

Multiobjective optimization methods are natural way to optimize multiple criteria of clustering by reducing biases of a particular single objective. In this paper, TSMPSO is proposed to increase exploratory capability of the MOPSO by introducing an additional diversity mechanism. Results show that TSMPSO outperforms well-known competitive algorithms in term of classification accuracy (F-measure) and quality assessment metrics across the range of real data sets. As the future work, the authors aim to extend this study for data sets with unknown number of clusters.

References

1. Akbari R, Hedayatzadeh R, Ziarati K, Hassanizadeh B (2012) A multi-objective artificial bee colony algorithm. Swarm Evol Comput 2:39–52
2. Bezdek J (1981) Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum Press
3. Branke J, Mostaghim S (2006) About selecting the personal best in multi-objective particle swarm optimization. In: Parallel problem solving from nature-PPSN IX, Springer, pp 523–532
4. Coello CAC, Pulido GT, Lechuga MS (2004) Handling multiple objectives with particle swarm optimization. Evol Comput IEEE Trans 8(3):256–279
5. Corne DW, Jerram NR, Knowles JD, Oates MJ, et al. (2001) Pesa-ii: Region-based selection in evolutionary multiobjective optimization. In: Proceedings of the genetic and evolutionary computation conference (GECCO2001)
6. Das S, Abraham A, Konar A (2008) Automatic clustering using an improved differential evolution algorithm. Syst Man Cybern Part A Syst Humans IEEE Trans 38(1):218–237
7. Deb K (2001) Multi-objective optimization using evolutionary algorithms. Wiley-interscience series in systems and optimization. Wiley
8. Deb K, Jain S (2002) Running performance metrics for evolutionary multi-objective optimizations. In: Proceedings of the fourth Asia-pacific conference on simulated evolution and learning (SEAL'02),(Singapore), pp 13–20
9. Deb K, Agrawal S, Pratap A, Meyarivan T (2000) A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: Nsga-ii. Lect Comput Sci 1917:849–858
10. Ehrgott M (2005) Multicriteria optimization. vol 2. Springer
11. Esmin AA, Coelho RA, Matwin S (2013) A review on particle swarm optimization algorithm and its variants to clustering high-dimensional data. Artificial Intelligence Review, pp 1–23
12. Ester M, Kriegel HP, Sander J, Xu X (1996) A density-based algorithm for discovering clusters in large spatial databases with noise. KDD 96:226–231
13. Faceli K, Sakata TC, de Souto MC, de Carvalho AC (2010) Partitions selection strategy for set of clustering solutions. Neurocomputing 73(1618):2809–2819
14. Handl J, Knowles J (2004) Evolutionary multiobjective clustering. In: Parallel problem solving from nature-PPSN VIII, Springer, pp 1081–1091
15. Handl J, Knowles J (2007) An evolutionary approach to multiobjective clustering. Evol Comput IEEE Trans 11(1):56–76
16. Handl J, Knowles J (2012) Clustering criteria in multiobjective data clustering. In: Parallel problem solving from nature-PPSN XII, Springer, pp 32–41
17. Hruschka ER, Campello RJGB, Freitas AA, De Carvalho APLF (2009) A survey of evolutionary algorithms for clustering. Syst Man Cybern Part C Appl Rev IEEE Trans 39(2):133–155
18. Jain AK, Dubes RC (1988) Algorithms for clustering data. Prentice-Hall Inc
19. Karaboga D (2005) An idea based on honey bee swarm for numerical optimization. Techn Rep TR06, Erciyes Univ Press, Erciyes
20. Karaboga D, Basturk B (2007) A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm. J Global Optim 39(3):459–471
21. Kennedy J, Eberhart R et al (1995) Particle swarm optimization. In: Proceedings of IEEE international conference on neural networks, Perth, Australia 4:1942–1948
22. Knowles JD, Corne DW (2000) Approximating the nondominated front using the pareto archived evolution strategy. Evol Comput 8(2):149–172
23. Kwedlo W (2011) A clustering method combining differential evolution with the k-means algorithm. Pattern Recog Lett 32(12):1613–1621
24. Miettinen K (1999) Nonlinear multiobjective optimization., International series in operations research & management science-Springer, US
25. Okabe T, Jin Y, Sendhoff B (2003) A critical survey of performance indices for multi-objective optimisation. In: Evolutionary computation, 2003. CEC'03. The 2003 congress on, IEEE. vol 2. pp 878–885
26. Park HS, Jun CH (2009) A simple and fast algorithm for k-medoids clustering. Expert Syst Appl 36(2):3336–3341
27. Reyes-Sierra M, Coello CC (2006) Multi-objective particle swarm optimizers: a survey of the state-of-the-art. Int J Comput Intell Res 2(3):287–308
28. Saha S, Bandyopadhyay S (2013) A generalized automatic clustering algorithm in a multiobjective framework. Appl Soft Comput 13(1):89–108
29. Sheng W, Liu X, Fairhurst M (2008) A niching memetic algorithm for simultaneous clustering and feature selection. Knowl Data Eng IEEE Trans 20(7):868–879
30. Storn R, Price K (1997) Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces. J Global Optim 11(4):341–359
31. Xu R, Wunsch D et al (2005) Survey of clustering algorithms. Neural Netw IEEE Trans 16(3):645–678
32. zdemir S, Attea B, Khalil n (2013) Multi-objective clustered-based routing with coverage control in wireless sensor networks. Soft Comput 17(9):1573–1584
33. Zhou A, Qu BY, Li H, Zhao SZ, Suganthan PN, Zhang Q (2011) Multiobjective evolutionary algorithms: a survey of the state of the art. Swarm Evol Comput 1(1):32–49
34. Zitzler E, Thiele L (1999) Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. Evol Comput IEEE Trans 3(4):257–271
35. Zitzler E, Deb K, Thiele L (2000) Comparison of multiobjective evolutionary algorithms: empirical results. Evol Comput 8(2):173–195