Mathematical Biosciences and Engineering

MBE, 17(3): 2193–2217.
DOI: 10.3934/mbe.2020117
Received: 15 May 2019
Accepted: 08 October 2019
Published: 13 January 2020

http://www.aimspress.com/journal/MBE

Research article

Proposing a novel community detection approach to identify co-interacting genomic regions

Mohammadjavad Hosseinpoor[1], Hamid Parvin[2,3,*], Samad Nejatian[3,5], Vahideh Rezaie[4,6], Karamollah Bagherifard[1,5], Abdollah Dehzangi[7], Amin Beheshti[8,*], Hamid Alinejad-Rokny[9,10,*]

[1] Department of Computer Engineering, Yasooj Branch, Islamic Azad University, Yasooj 1979124119, Iran
[2] Department of Computer Engineering, Nourabad Mamasani Branch, Islamic Azad University, Nourabad Mamasani 1979123114, Iran
[3] Young Researchers and Elite Club, Nourabad Mamasani Branch, Islamic Azad University, Nourabad Mamasani 1979123114, Iran
[4] Department of Electrical Engineering, Yasooj Branch, Islamic Azad University, Yasooj 1979124119, Iran
[5] Young Researchers and Elite Club, Yasooj Branch, Islamic Azad University, Yasooj 1979124119, Iran
[6] Department of Mathematics, Yasooj Branch, Islamic Azad University, Yasooj 1979124119, Iran
[7] Department of Computer Science, Morgan State University, Baltimore 21251, United States
[8] Department of Computing, Macquarie University, Sydney 2109, Australia
[9] Systems Biology and Health Data Analytics Lab, The Graduate School of Biomedical Engineering, UNSW Sydney, Sydney, NSW, 2052, Australia
[10] School of Computer Science and Engineering, UNSW Sydney, Sydney, NSW, 2052, Australia

* **Correspondence**: Email: h.alinejad@ieee.org, parvin@iust.ac.ir.

Abstract: Modern next generation sequencing technologies produce huge amounts of genome-wide data that allow researchers to have a deeper understanding of genomics of organisms. Despite these huge amounts of data, our understanding of the transcriptional regulatory networks is still incomplete. Conformation dependent chromosome interaction maps technologies (Hi-C) have enabled us to detect elements in the genome which interact with each other and regulate the genes. Summarizing these interactions as a data network leads to investigation of the most important properties of the 3D genome structure such as gene co-expression networks. In this work, a Pareto-Based Multi-Objective

Optimization algorithm is proposed to detect the co-expressed genomic regions in Hi-C interactions. The proposed method uses fixed sized genomic regions as the vertices of the graph. Number of read between two interacting genomic regions indicate the weight of each edge. The performance of our proposed algorithm was compared to the Multi-Objective PSO algorithm on five networks derived from cis genomic interactions in three Hi-C datasets (GM12878, CD34+ and ESCs). The experimental results show that our proposed algorithm outperforms Multi-Objective PSO technique in the identification of co-interacting genomic regions.

Keywords: community detection; genomics graph interaction; modularity; multi-objective optimization; health data analytics; genomic interacting regions

1. Introduction

Nowadays, in medical bioinformatics science, because of the complexity of the biomedical data [1], the diagnosis of disease related factors is challenging. Specially, for those genomics factors that are working together to perform a biological function [2]. The detection of these interacting genomic elements is very important for better understating of disease factors. Chromosome Conformation Capture (3C) assays are now the method of choice to study the role of DNA looping in transcriptional regulation. These assays directly identify genomic loci that are brought in close enough proximity to each other in living cells to be cross-linked. This new technology allows for the mapping of chromatin interactions on a whole genome level. Cabreros et al. used a community-based algorithm on Hi-C data to detect community of interacting genomic regions in mice and humans. Their proposed algorithm was able to detect a variety of communities. Also, this algorithm could detect communities of neighboring DNA locations [3]. In 2016, Fotuhi et al. presented a multivariate clustering algorithm for the chromosome configuration data analysis to identify patterns of chromosomal interactions [4]. In 2016, Li et al. presented an optimal multi-objective algorithm based on the Particle Swarm Optimization algorithm (PSO) to detect communities in social networks [5]. In fact, this algorithm was able to detect the communities of nodes in each run. To test the effectiveness of this algorithm, the authors performed extensive experiments on artificial and real data. Finally, their experiments showed that their proposed method works better that those previous methods found in the literature [5]. In 2019, Zhou et al., proposed a graph-based clustering approach called "AR-Cluster" to identify communities in a complex network [6]. In this method, nodes in the graph are grouped together by a K-medoid framework.

As it was mentioned earlier, the detection of the communities in a complex network is challenging in most research fields such as computer science, social networks, biology, physics and medicine. Many of the proposed methods are typically related to the topological issue, the similarities between the attributes, or the degree of input and output of each vertex [7–15]. However, when the graph is widespread and complex, the identification of the communities would be either inefficient or time consuming [16–20]. Therefore, community detection in complex graphs has always been challenging [21–26]. In order to resolve the challenge ahead, in this study, a Pareto-based Multi-Objective Optimization Genetic Algorithm is proposed to identify communities in the complex networks. We performed our proposed method on Hi-C interactions in mouse genome to

identify interacting genomic regions. Our benchmarking results demonstrate that our proposed method work better than existing methods found in the literature to identify genomic interacting regions.

2. Material and methods

In the following sections, our proposed genetic-based multi-objective optimization algorithm to identify communities in Hi-C interactions of genomic regions is explained. In the network, regions of genome are demonstrated as the nodes and edges demonstrate the interactions between them. In addition, the weight of each edge is interrelated with these vertices. In this study, the Hi-C data obtained from NCBI database (GSE35156 and GSE69600) and analyzed by HiC-Pro package [27].

2.1. Problem statement

A non-oriented weighted graph provides a network with nodes and edges which can be represented as G = (V, E). Here the graph components are: V which is the set of nodes and E which is the edge set. The non-oriented weighted graph G consists of $|V| = N$ nodes with $V = \{v_1, v_2, \dots, v_n\}$ and $|E| = M$ as $E = \{e_1, e_2, \dots, e_M\}$ and $W = \{w_1, w_2, \dots, w_M\}$. Also, the set of communities of the graph is represented as $C = \{c_1, c_2, \dots, c_n\}$, in which any $c_i \in C$ represents a community of the graph G.

2.2. Detect of community in the graph

In this section, our proposed method for exploring and extracting community in the genomic grid network is described. The proposed approach is a multi-objective genetic optimization algorithm based on Pareto optimization. To explain our model, assume that the graph G = (V, E) is the input of the algorithm in accordance with what was explained in the previous section. Below the objectives of our proposed method are explained one by one.

2.2.1. First objective: Modularity function

The modulatory function f_1 in G is defined as follows:

$$f_1 = \sum_{i=1}^{nc} \left[\frac{ne_i}{sl} - \left(\frac{dv_i}{2M} \right)^2 \right] \quad (1)$$

In this function, *nc* is the number of total community, ne_i the total number of edges in the community i, dv_i the sum of the degrees of the vertices in the *i-th* community, and *M* the total number of edges in the graph [28–31]. In the proposed algorithm, the value of the f_1 function lies in the interval [0,1], where the best mode of the function f_1 is when its value is maximal [19].

2.2.2. Second objective: Average weight vertex function

The average weight function of community f_2 is defined as follows:

$$f_2 = \sum_{i=1}^{nc} \sum_{j=1}^{N_i-1} \sum_{k=j+1}^{N_i} \frac{w_{jk}}{N_i} \quad (2)$$

Here, N_i is the number of vertices in the *i-th* community, W_{jk} the weight between vertex j and k. f_2 is the weighted average of the community obtained by the proposed algorithm. The best mode for f_2 is when it is maximal.

Therefore, in genomic graphs, both objectives (f_1 and f_2) must be considered to detect community. So, in each run, both objectives to be optimum in the sense of maximal.

2.3. Multi-Objective Optimization (MOO) concept

A multi-objective optimization problem with *m* decision variable and *n* objective is defined in relation 3 [32].

$$\text{Max } (y = f(x)) = \text{Max } ([f1(x) \ldots fn(x)]) \quad (3)$$

where $x = (x_1, \ldots x_m) \epsilon X$ is a vector of the *m*-dimensional decision and X is the search space, and $y = (y_1, \ldots y_n) \epsilon Y$ is the target vector and Y is the target space. In general, in MOO, there is no single optimal solution for all purposes. In such cases, the optimal solution is a set of optimal solutions for one or more goals [25,33–36]. This set is known as the optimal Pareto collection. Some of the Pareto concepts used in the multi-objective optimization are explained below.

2.3.1. Concept of Pareto dominance

To compare the qualities of the two solutions X and Y, we shall use the concept of dominance. For two decision vectors x_1 and x_2, the dominance (represented by $\prec$) is defined as Eq (4):

$$x_1 \prec x_2 \Longleftrightarrow \forall_i f_i(x_1) \leq f_i(x_2) \wedge \exists f_j(x_1) < f_i(x_2) \quad (4)$$

The decision vector x_1 dominates x_2 if and only if x_1 is better than x_2 for all targets, and x_1 is exactly higher than x_2 for at least one target [34].

2.3.2. Pareto optimal collection

The collection of all optimal Pareto decision vectors is referred as the P_S optimal Pareto collection.

$$P_S = \{x_1 \epsilon X, \mid \nexists x_2 \epsilon X: x_2 \prec x_1\} \quad (5)$$

The decision vector x_1 is called the optimal Pareto when it is not dominated by all the other decision vectors x_2 of the set.

2.3.3. Optimal Pareto front

The optimal front of the Pareto P_F is the optimal Pareto image in the target space.

$$P_F = \{f(x) = (f_1(x) \ldots fn(x)) \mid x \epsilon P_S\} \quad (6)$$

2.3.4. Crowding distance

The next concept used in multi-objective optimization based on Pareto is the crowding distance. Here we calculate the crowding distance for each of the objective functions separately. For example, if we have two objective functions, for each solution *i*, we calculate the crowding distance from *i* to all the other solutions *j* on the common front with *i* for both objective functions f_1 and f_2.We then consider the sum of these two distances as the crowding distance of the solution *i*. The crowding distance for the solution *I* is calculated as:

$$cd_i = \sum_{\substack{j \in \{F_{f_i}\} \\ i \neq j}} d_{ij} \tag{7}$$

To calculate the crowding distance *i* for each objective function, we also use the following formula:

$$d_i = \frac{|f^{i+1} - f^{i-1}|}{f^{max} - f^{min}} \tag{8}$$

were f^{max} and f^{min} are the minimum and maximum of the target function, respectively, and f^{i-1} and f^{i+1} are the solutions before and after the solution *i*, respectively.

In other words, for each objective function, first, the solutions are arranged in descending order, and then the maximum value is considered as f^{max} and the minimum value is considered as f^{min}. Afterwards due to sorting of solutions, one can also easily identify the previous and next solutions. The Eq (8) is computed for each *i*, and finally, after calculating d_i, we can calculate the distance of crowding for all target functions [33,34,37,38].

2.3.5. Non-dominated Sorting (NS) algorithm

We use this algorithm to sort the paths and determine the Pareto fronts. This algorithm works in the following manner.

1) For all members of the population, we define a set called *sp* with null value and one variable called *np* with zero initial value. Hence, we will have:

sp = The set of answers that dominated by *p*.

np = the number of times the solution is dominated by the other solutions.

2) For each possible pair *p* and *q* of the population members we have:

If *p* dominates *q*, then add *q* to *sp*

If *q* dominates *p*, then add one unit to *np*

3) Add all the members of the population with *np* = 0 to F_1 (the first Pareto Front).

Using the actual Pareto Front (F_k) the next Pareto Front (F_{k+1}) is created. For this purpose, by eliminating the effect of the members of the F_k, the members are not dominated form the F_{k+1} members.

4) We put the counter of fronts or fronts equal to 1, that is, $k = 1$

5) Consider *Q* as a draft of F_{k+1}.

6) For each member of F_k, such as *p*, and for each member of *sp* such as *q* (all *q*s that are dominated by *p*), one unit of *nq* is subtracted (i.e. the effect of *p* to *q* is not considered)

7) If we get $nq = 0$ while decreasing, then add *q* to *Q*.

8) If *Q* is empty (that is, nothing is left to add), the sorting process is over. And if *Q* is not empty, consider F_{k+1} as *Q* and add one unit to *k* (Pareto front counter) and go to step 5. This will allow us to complete our Pareto fronts gradually [32,39].

2.4. Multi-objective genetic optimization algorithm based on Pareto

As mentioned before, the proposed algorithm for community in the genomic grid network is a multi-objective optimization algorithm based on Pareto optimization. Here, the optimal evaluation mode function is when this function is maximized. In fact, the maximum of the evaluation function is obtained when the values of both f_1 and f_2 are maximal.

$$Optimum(F) = \begin{cases} \max(f_1) \\ \max(f_2) \end{cases} \tag{9}$$

In this research, we have converted the GA algorithm into the multi-objective algorithm to discover community by adding the following steps:

1) The quality of the solution based on the concept of dominance and using the Non-dominated Sorting algorithm or the NS algorithm.

2) Arranging the solutions based on the concept of crowding distance.

In fact, multi-objective operations of the algorithm can be achieved by adding the following steps in the selection section of the solutions:

a) Non-dominated sorting

b) Calculating the Crowding distance

c) Sorting the answers

In multi-objective optimization, two criteria of the quality of solutions and their order are important:

i) We look for an appropriate approximation of the Pareto front, which means that the answers we receive are surely non-dominated.

ii) These answers cover virtually all of Pareto Front.

The goal of solving a multi-objective optimization problem is to find a form that has the quality and the order at the same time. An algorithm can be suited only when it has, first and foremost, a good quality, and second, provides order. Here, our primary criterion is to compare Dominate answers (i.e., which solution will dominate).

If, based on the dominance, we were not able to choose one of the two solutions, then the second factor would be the order. The proposed method is described in Figure 1.

The highlights of the proposed algorithm are:

i) The answer with no other answer better than that has more points. The answers are ranked and arranged based on how many answers are better than them.

ii) The fitness for the answers is based on their ranks and failure of dominance by the other answers.

iii) The fitness crossover method is used for close answers so that the distribution of the answers is optimally adjusted and the answers are distributed uniformly in the search space.

2.4.1. Main components

In this section, we introduce the main components of the proposed algorithm and describe each of them.

1) The process of deleting and re-selecting

Generally, the selection of parent members for the operation of the crossover operator occurs probabilistically. In other words, each member of the population with a specific probability of p_c may

be involved in the creation of a child member. Also, it is necessary to consider the following when choosing the parent particles.

a) Because of the probability of selecting the parent members, a member of the population may be selected twice as a parent member. In other words, a certain member may have the role of both parents at the same time. In this case, the child member will be the same as his male parent. For this reason and to avoid unnecessary crossovers, a combination test should be used.

b) Sometimes a member may have a role in creating a parent member several times. Alternatively, one member may be selected many times as a parent member. This is problematic when using the fitting pattern appropriately.

Before we introduce this component, we must first describe a comprehensive random selection method because ideas have been taken from this method.

1. Create an initial Population
2. Calculation of fitness criteria
3. Sorting the population based on dominance conditions
4. Calculate the distance of crowding
5. Selection: As soon as the initial population is sorted according to the dominance conditions, the distance of the crowd will be calculated and the selection starts from the initial population. This selection is based on two elements:

5.1. POPULATION: Population is selected from lower ranks.

5.2. Calculation of crowding distance: Assuming that p and q are two members of the same rank, that member is selected which has a greater crowding distance. It should be noted that the priority of the selection is first with the rank and then based on the distance of crowding.

6. Performing of crossovers and mutations to produce new offspring.
7. Composing the primary population with the population obtained from crossover and mutation.
8. Replacing the parent population with the best members of the population integrated in the previous stages. In the first step, lower-ranking members replace older ones and are then ordered according to the crowding distance. Primary population and the population induced by crossover and mutation are first categorized by rank, and then, some of those ranked lower are eliminated.
9. The remaining population is arranged according to the distance of crowding. Here the sorting is done in one front.
10. All stages are repeated until reaching the desired generation (or optimal conditions).

Figure 1. Pseudo-code of proposed algorithm.

2) Comprehensive random selection method

Using comprehensive random selection, it is possible to select members of the population based on their target function. In other words, the probability of chromosome selection is proportional to the value of the objective function of the chromosome. By this method, the time to find optimal solutions can be reduced. However, this method has its own disadvantages. For example, in the early generations, there is a tendency to dominate a number of superior chromosomes over the selection process while in the latter generations when the population converges completely, the competition between the chromosomes is not very serious but almost randomized. In the early generations, usually there are a

lot of differences in fitting values. Hence, the likelihood of the presence of chromosomes with greater fittings is far higher. In the late generations, since the fittings of chromosomes are closely matched, choices are roughly random and the chances of choosing most of the chromosomes are equal.

In this process, the proposed algorithm initially selects two parents to perform the crossover process similar to the general genetic algorithm. Parents are selected using binary tournament selection method. The goal here is to select the high-quality chromosomes immediately after the parents are selected. However, these two parents may not be the best of the population. The idea of a comprehensive random selection method is taken here. Here we have a control parameter for the substitution of the worst chromosome. The goal is to select the widest chromosomes each to carry out the crossover process. The value of this parameter in tests was 0.005. If the difference between the two selected parents exceeds the control parameter, the chromosome is worse than the crossover cycle and another parent is selected. The process of removing and re-replacing continues as long as the difference between the parents is less than the control parameter value. Parent comparison is based on a fitness function, and the parent who has a lower fitness value will be selected for the removal process. Fitness function is the sum of the functions f_1 and f_2. Figure 2 shows pseudo code of the process of deleting and re-selecting.

```
α = 0.005;
select Tow Chromosome with binary tournament selection method
Calculate difference between fitness function of the two selected Chromosomes
While (difference > α and difference <> 0)
Find the worst Chromosome and select new chromosome
Flag = 0;
While flag == 0
If newly chromosome was better than worst Chromosome
Flag = 1;
replace new chromosome with worst chromosome
end while
Calculate difference fitness function between new chromosome and selected Chromosome
end while
```

Figure 2. Pseudo code of the process of deleting and re-selecting.

2.4.2. Participation of the best chromosome in different generations in the crossover process

The crossover process in the genetic algorithm creates children's chromosomes from parent chromosomes. A crossover operator is applied on one or more parent chromosomes at a time and creates one or more children. In practice, operators are defined in terms of the type of problem and are fully dependent on the ability of the analyst. The efficiency of these operators in giving the optimal solution varies from problem to problem. Some operators consider only one chromosome and based on their information create new chromosome. However, others do further operations on some or even all of the chromosomes in the population.

In addition to choosing parent chromosome and the crossover process, the crossover operator takes into account an alternative policy so that after creating a child member, this one can replace the worst parent member. This type of replacement can be the source of the restriction that a child member should be better than parent member. Accordingly, the crossover operator must be executed so that the worst member in the population is replaced by the child member.

In this process, the proposed algorithm utilizes the position of the best population chromosome in the current generation to carry out the crossover process. The purpose of using this component is to produce new opportunities near to the global optimal one. In this component, we use the one-point crossover method, but with the difference that the position of the best chromosome in this process will be considered. In this component, first, the one point is selected along the parent chromosomes, and then first the parent and after that the best chromosome takes the first child's position. The second child is also produced in the same way, but with the difference that first the second parent genes and then the genes of the best chromosome make up the child's chromosome. Figure 3 shows the crossover method with the participation of the best chromosome.

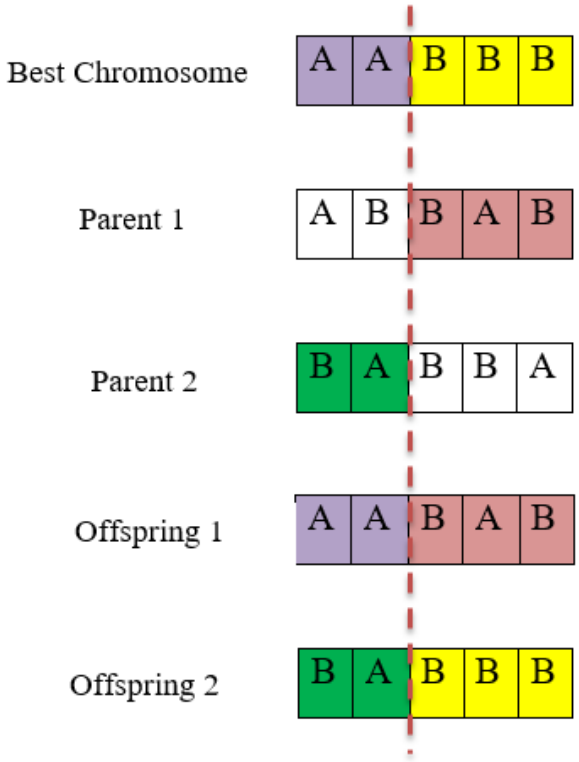

Figure 3. Participation of the best chromosomes in different generations in the crossover process.

2.4.3. The three-point mutation process

The goal of the mutation is to express a genetic property that increases the diversity of the population's responses. In three-point mutation method, as in the usual methods in the general genetic algorithm, a member of the population is randomly selected and entered into the mutation process. In this case, three points are chosen randomly along the chromosome. Then, using the uniformly

continuous randomized mutation operator, these three points will be changed in a way that the values of the two points of the three selected points are modified by the pattern of the best current chromosome. Here, according to Figure 4, three genes are selected randomly along the chromosome. Then these three genes are modified using the mutation operator, but with the difference that the position of the best chromosome is involved in the mutation process. In fact, two randomly selected genes are modified by the pattern of the best chromosome, and the other gene changes in accordance with the random procedure.

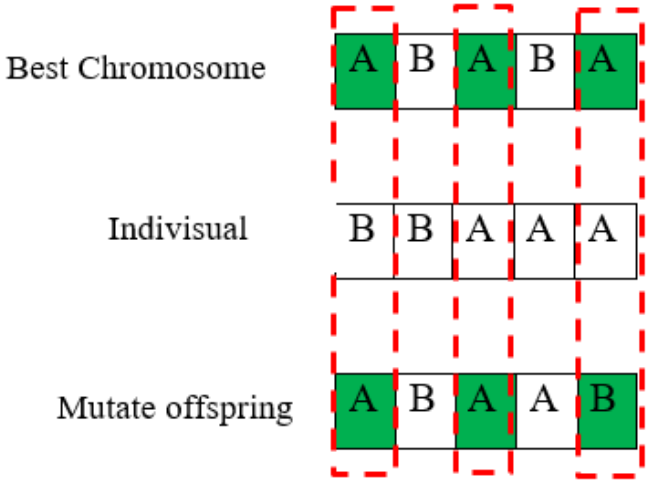

Figure 4. The process of mutation operation in the proposed algorithm.

2.5. Structure of chromosome in the proposed algorithm to detect community

The structure of a chromosome in the proposed algorithm, as a $1 \times N$ vector, contains the N genes. The N genes in this structure represent the vertices in a graph (Figure 5).

C1	C1	C1	.....	C_{nc-1}	C_{nc}
1	2	3	.....	N-1	N

Figure 5. Structure of chromosome.

The content of each gene in the chromosome represents the community number that its vertex belongs to. In this structure, *nc* is the number of communities in each chromosome structure the amount of which is variable in each structure. Therefore, the desired chromosome is an N element array with each element indicating a vertex in the graph and its content denotes the community number to which it belongs. An example is given below for further explanation. Suppose there is a graph with 5 vertices and 5 edges as shown in Figure 6.

Then a chromosome structure can be defined as shown in Figure 7.

In this case, the chromosome can be a solution to the problem of discovering the community in the graph with 5 vertices. Accordingly, the vertices 1 and 2 are in the first community and the vertices 3–5 are in the second.

Community 1: 1, 2

Community 2: 3, 4, 5

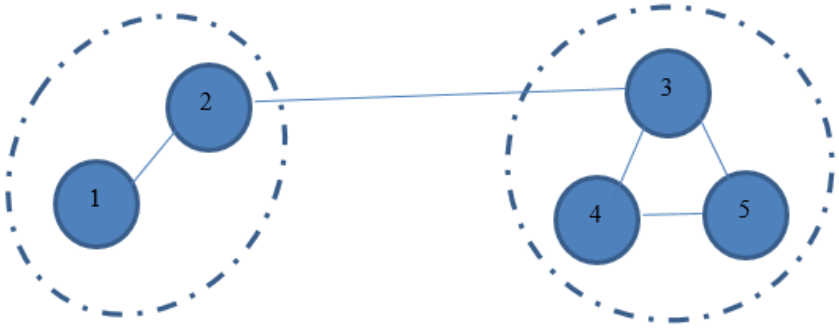

Figure 6. Discover two consonants in a graph with 5 vertices and 5 edges.

1	1	2	2	2
1	2	3	4	5

Figure 7. The sample of chromosome.

2.6. Parameters

The parameters of the proposed algorithm were adjusted according to experiments with different values, and also by analyzing the researchers conducted in [21,33,39–42]. Figure 8 shows the diagram of the results pertaining to 100 executions of the algorithm upon the 5 Kbp graph concerning the data set ESCs with regard to different values of crossover operation, mutation percentage, and initial population. The diagram depicts the value results of the evaluation function which is the sum of two functions f_1 and f_2 for the 100 executions. Accordingly, for any of the three parameters, four different values were examined. The results show that the best values for crossover percentage, mutation percentage, and initial population are 0.8, 0.3 and 50, respectively. As observed, the algorithm gives similar results close to the 100 iterations. Hence, in the proposed algorithm according to Table 1, the maximum iteration is equal to 100, the number of sub iteration is 30, the population size is 50, the crossover rate is 80%, and the mutation rate is 30%. Also, this algorithm chooses roulette wheels to select people for crossover and mutation operations. After the crossover and mutation operations, the children obtained from these operations are evaluated using the Pareto optimal frontier. After that, these children are merged with the previous population and 50 members that are better than the population are chosen as the new population. After 100 iterations, the best member of the population is considered as the answer to the problem in accordance with Pareto optimization, which consists of all community detected in the input graph (Figure 8).

3. Results and discussion

We evaluate our proposed method using three new benchmarks. These three benchmarks are the genomic interaction graphs namely, GM12878, $CD34^+$, and ESCs. In this section, the multi-objective optimization algorithm is used to find community in 10, 100, 500 kb, and 1 Mbp graphs resulting from interactions in the GM12878 and $CD34^+$ blood cells and the 5 kb graph from the existing interactions in the Embryonic Stem Cells (ESCs) of mouse. Also, the efficiency of the proposed algorithm has been analyzed compared to multi-objective particle swarm optimization algorithm in community detection [5,32,40]. In Table 2, the detail information is provided for each of these graphs.

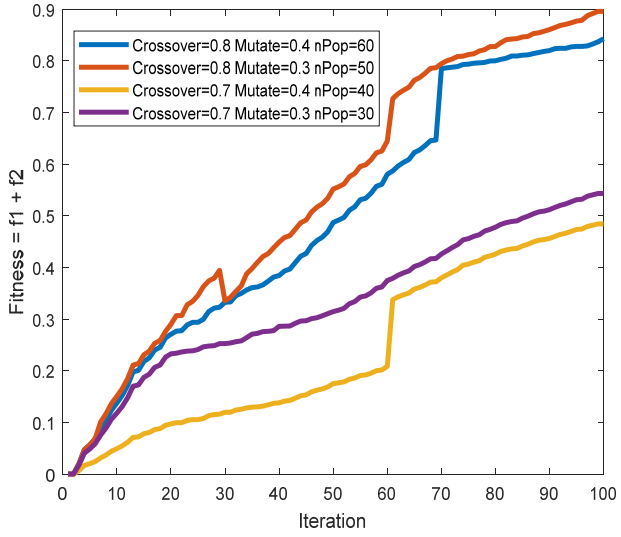

Figure 8. Results 100 times implementation of the proposed algorithm for parameter adjustment.

Table 1. Proposed algorithm parameters.

Parameter	Value
Iteration Number	100
Sub-iteration Number	30
Population size	50
Crossover rate	0.8
Mutation rate	0.3

Table 2. Graphs used for testing.

Graph	Number of vertex	Number of edge
5 kb	333	202
10 kb	3715	2117
100 kb	777	399
500 kb	1086	622
1 Mbp	497	309

3.1. Computational complexity

In this section, our proposed algorithm, Multi Objective Genetic Algorithm Optimization Community Detection (MOGAOCD), is compared to Multi Objective Particle Swarm Optimization Community Detection (MOPSOCD) algorithm [32] from the viewpoints of CPU usage, RAM usage, and execution time. Here, the graphs are sorted increasingly according to the number of nodes as 5, 10, 100, 500 kb, and 1 Mbp. Both algorithms are run on a same HP server and in the same conditions according to Table 3 with the following specifications.

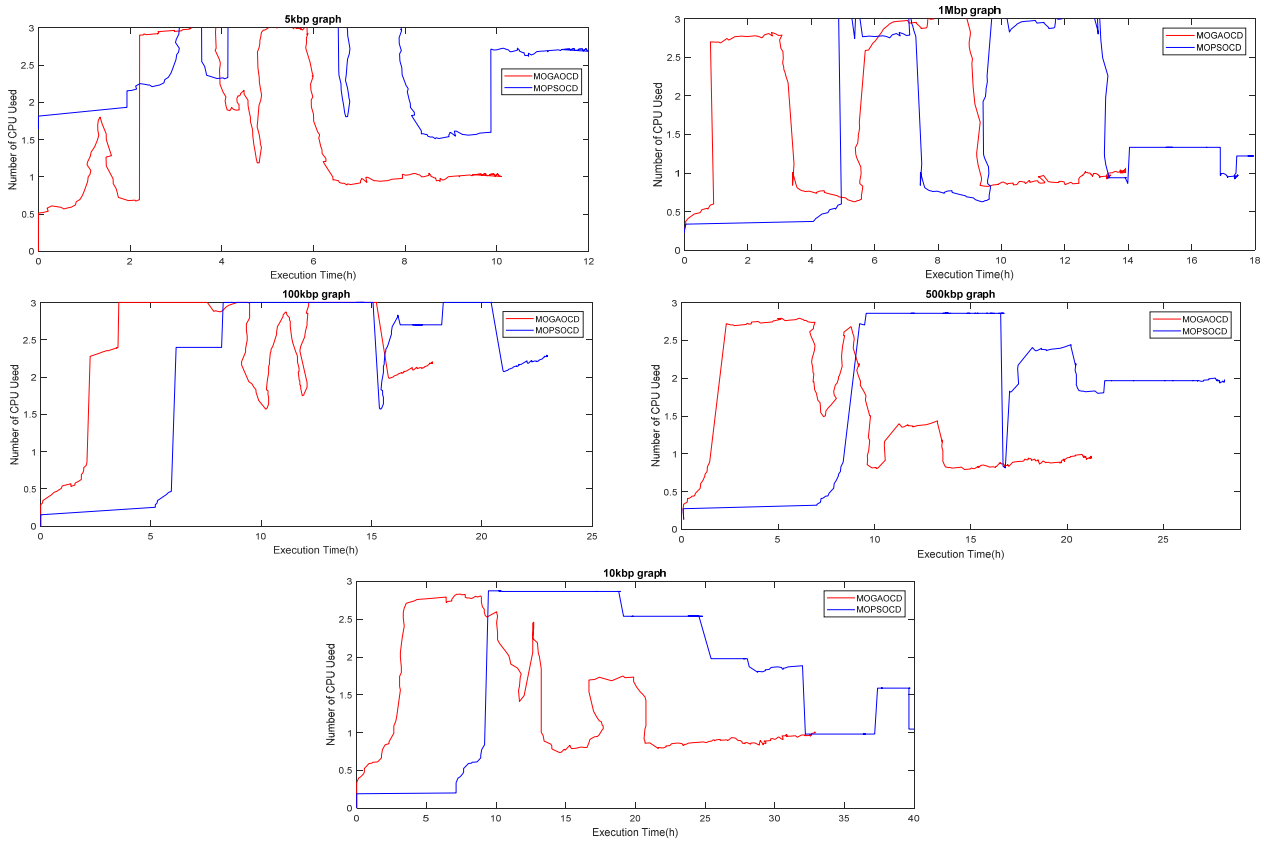

Figure 9. Number of CPU used, 3 CPU: In both algorithms (MOGAOCD (in red), MOPSOCD (in blue)) to community detection in all genomic graphs.

3.1.1. CPU usage

In this part, the CPU usage in the five graphs to detect the community in both algorithms are compared. Figure 9 shows the CPU usage in the MOGAOCD and MOPSOCD relative to execution time. As shown in this figure, the greater graph the less CPU usage due to longer execution time. Accordingly, both CPU usage and execution time are greater in the MOPSOCD algorithm compared to the MOGAOCD.

To provide more insight, the CPU usages in both algorithms for all graphs are demonstrated in Figure 10. As shown in this figure, the CPU usage in our proposed algorithm is also less in average compared to other algorithms.

Table 3. Specification of the system.

Computer system	Number of CPU	Capacity of RAM	Capacity of H.D.D
HP ProLiant DL380p Generation8 (Gen8)	3 Intel® Xeon® E5-2609 v2 (2.5 GHz/4-core/10 MB/6.4 GT-s QPI/80W)	16 (GB)	25 (TB)

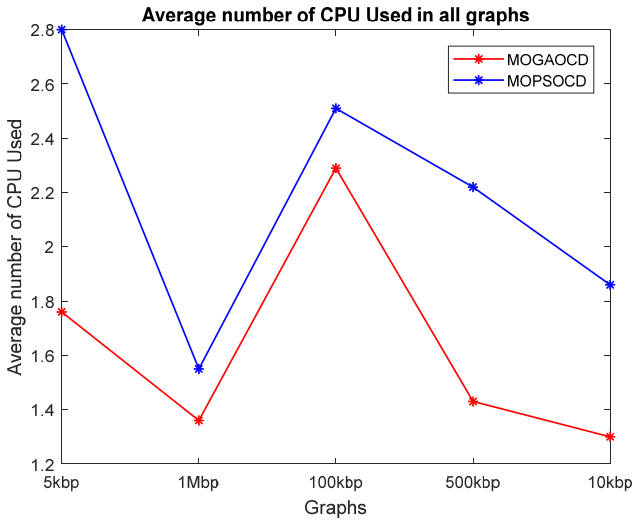

Figure 10. Average number of CPU used, 3 CPU: In MOGAOCD and MOPSOCD to community detection in five genomic graphs.

3.1.2. RAM usage

In this part, the RAM usage in the five graphs to detect the community in both algorithms are compared. According to Figure 11, in general, more RAM is used when the graph is bigger. As a result, RAM usage is greater in the MOPSOCD algorithm than in the MOGAOCD.

To provide more insight, the RAM usages in both algorithms for all graphs are portrayed in Figure 12. As shown in this figure the RAM usage in our proposed algorithm in average is also less than that other algorithms.

3.1.3. Execution time

Figure 13 shows the execution times in the MOGAOCD and the MOPSOCD algorithms for five graphs. As observed, the execution times of all graphs in the MOGAOCD are shorter than those in the MOPSOCD which is a token of the superiority of the MOGAOCD algorithm over the other. This preference due to a shorter execution time is more apparent in bigger graphs of 10, 100 and 500 kb which are more computational complex in community detection.

We next compare the number of CPU cores used in both algorithms in the 5kb graph. Figure 14 shows that the number of CPU cores used in both algorithms in this graph. According to this figure, the number of CPU cores used as well as the execution time in the MOPSOCD algorithm are greater than the MOGAOCD algorithm. Therefore, the MOGAOCD algorithm performs better than the MOPSOCD algorithm in number of CPU cores consumption and execution time.

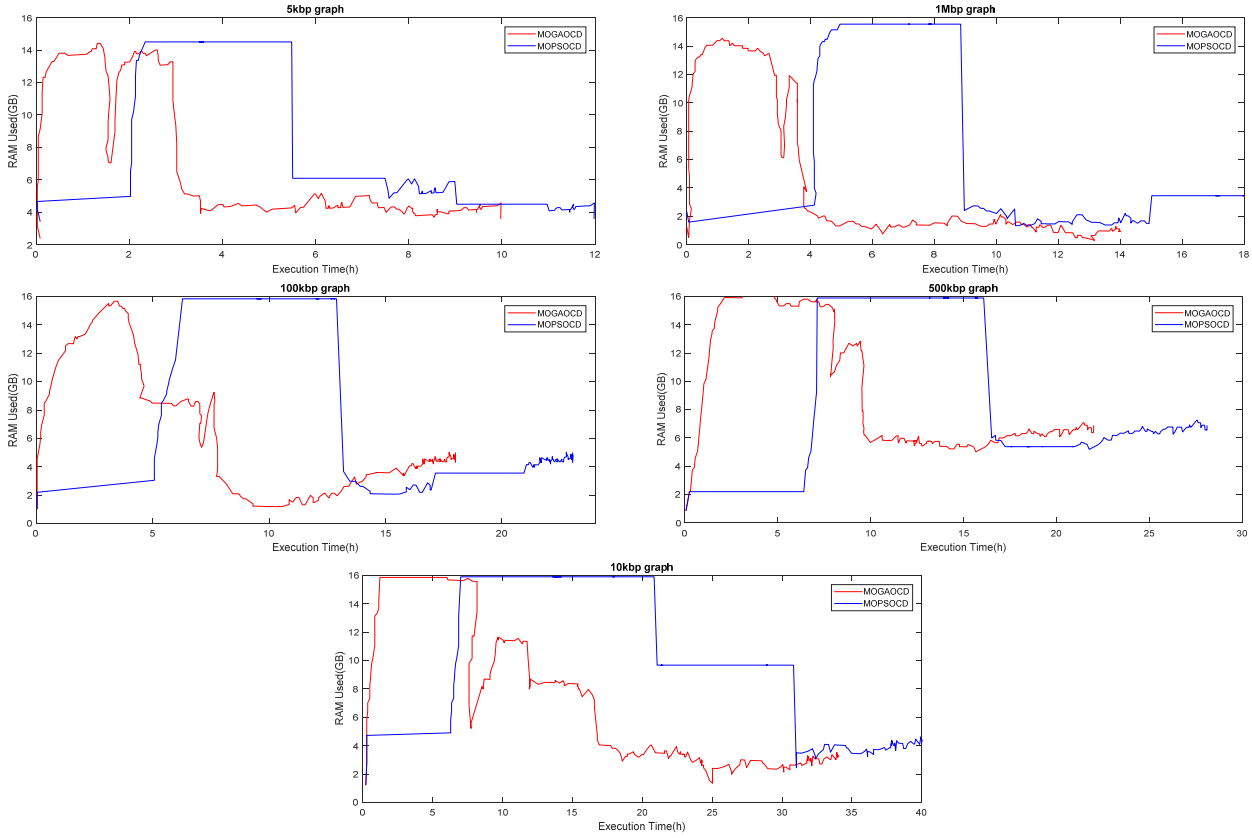

Figure 11. RAM used (GB), 16 GB of RAM: In both algorithms (MOGAOCD (in red), MOPSOCD (in blue)) to community detection in all genomic graphs.

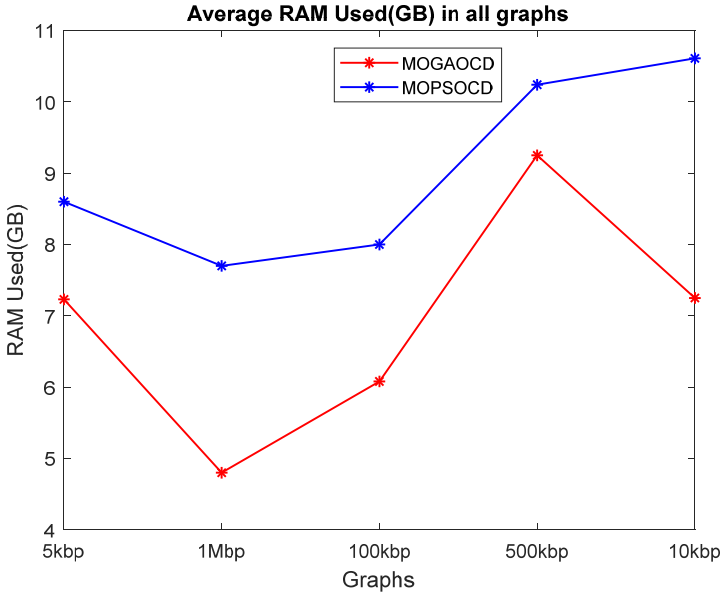

Figure 12. Average RAM used (GB), 16 GB of RAM: In MOGAOCD and MOPSOCD to community detection in five genomic graphs.

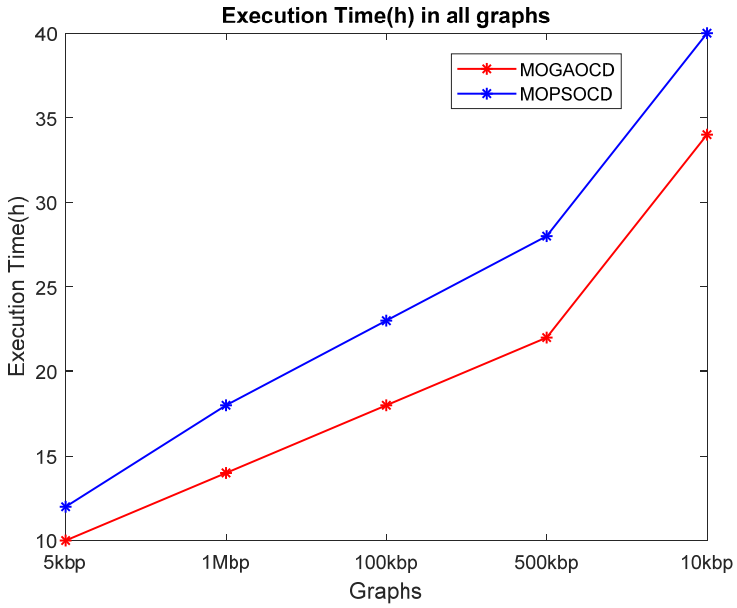

Figure 13. Average execution time (h), 16 GB of RAM, 3 CPU: In MOGAOCD and MOPSOCD to community detection in five genomic graphs.

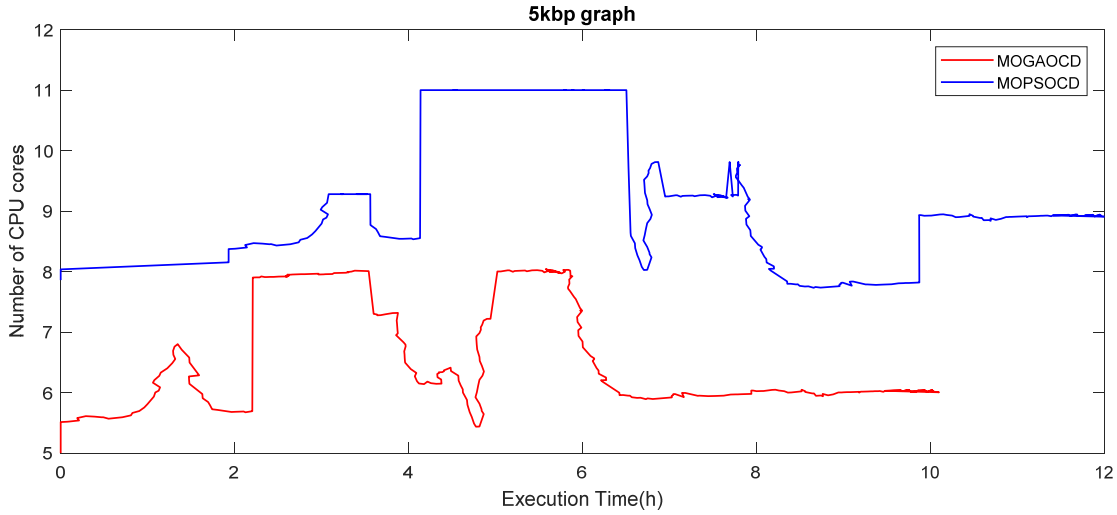

Figure 14. Number of CPU cores, 12 CPU cores: In both algorithms (MOGAOCD (in red), MOPSOCD (in blue) to community detection in 5 kb graph.

3.1.4. Scalability

Figure 15 shows the scalability of the proposed algorithm in each of the five graphs. In this experiment, the graphs are given to the system individually in five stages where the execution times are computed, respectively. As it is shown, at each stage, 20% of the graph enters the system and the resultant execution time is recorded. As illustrates in Figure 15, the system is able to achieve better execution times through MOGAOCD when the number of nodes in each graph is increased.

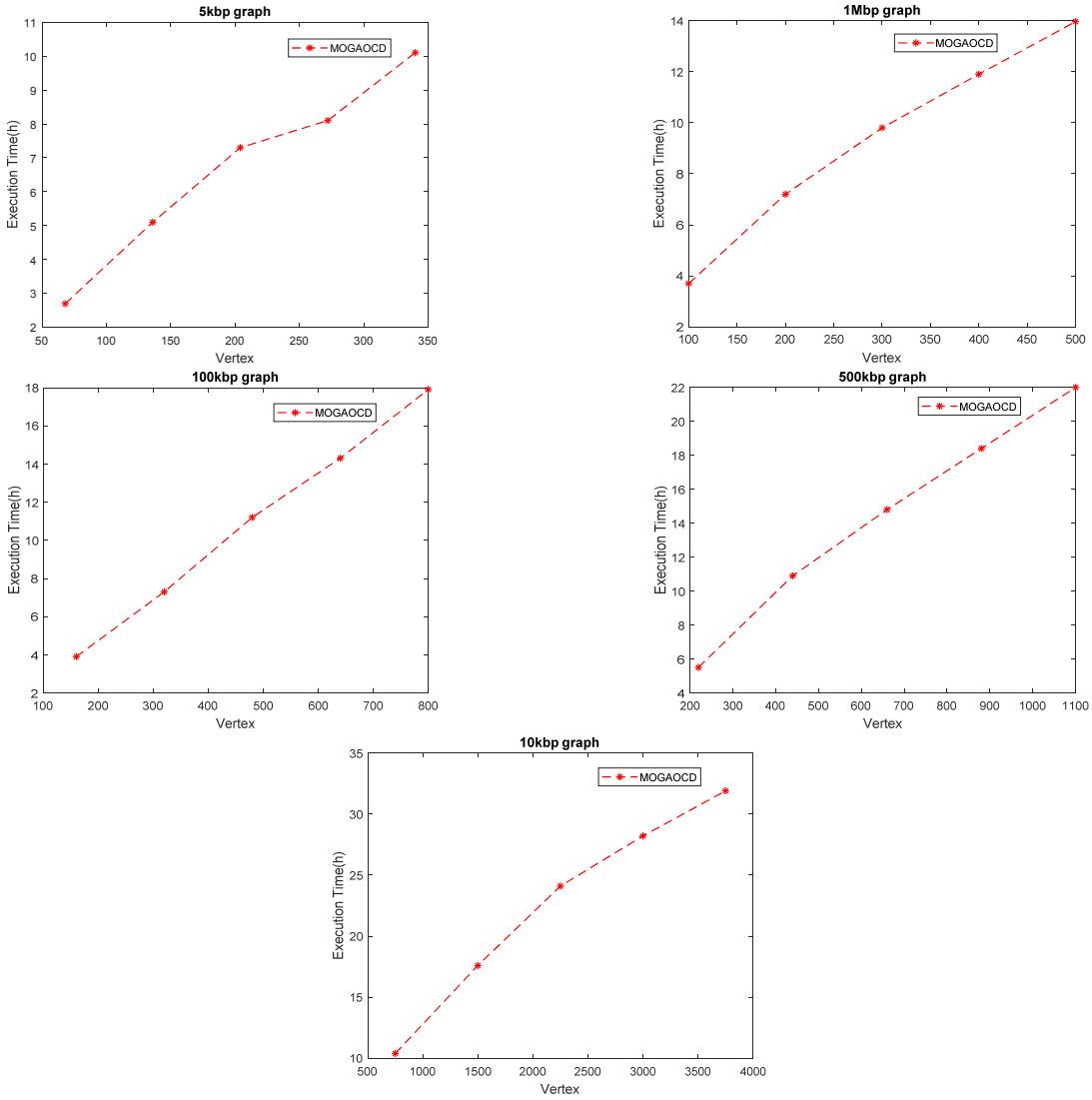

Figure 15. Scalability of MOGAOCD to community detection in five genomic graphs.

3.2. The performance of the proposed method in community detection for GM12878 and CD34$^+$ genomic graphs

In this section, the performance of the proposed algorithm using GM12878 and CD34$^+$ graphs (in both graphs, the inter-genomic interactions are found at the same points of the genome) in the 10, 100, 500 kb and 1 Mb size fragmentation are investigated and analyzed. Here, our proposed algorithm (MOGAOCD) is compared and analyzed along with the MOPSOCD algorithm [32] based on three criteria namely, the number of community detected, modularity value, and the mean weight of the vertices. The aim in each benchmark is to maximize these three criteria. Here both algorithms are implemented in MATLAB. In the evolutionary algorithms, the result of a single run is usually not enough to conclude generality. Hence the algorithm is executed 100 times and the average is derived from the obtained results. In each run, for the archival collections, the values of the two objects (modularity, the mean weight of the vertices) have been calculated. The 10 kbp graph contains 3715 vertices and 2117 edges.

Here, the results of the implementation of the proposed algorithm and the Pareto-based comparison algorithm are depicted in order to optimize the two objectives. Also, the average results of the two target values in each run are also given. In order to display the results for two purposes, a two-dimensional diagram is considered, each dimension of which represents the amount of a target.

In Figure 16, the solutions produced by the MOGAOCD and the MOPSOCD algorithms are shown in accordance with the Pareto front. The diagram consists of a number of red and blue points. The red dots represent the Pareto front solutions generated by the MOGAOCD and the blue dot, representing the Pareto-particle algorithm solutions in the MOPSOCD. As shown in Figure 16, the red dot contains the best responses as it has the highest modularity and the average weight of the vertices. By viewing the position of each solution, including red and blue points, the modularity value and the average weights of vertices in each solution can be observed. Figure 16 illustrates the preference of the MOGAOCD in community detection in the 10 kb graph over the MOPSOCD.

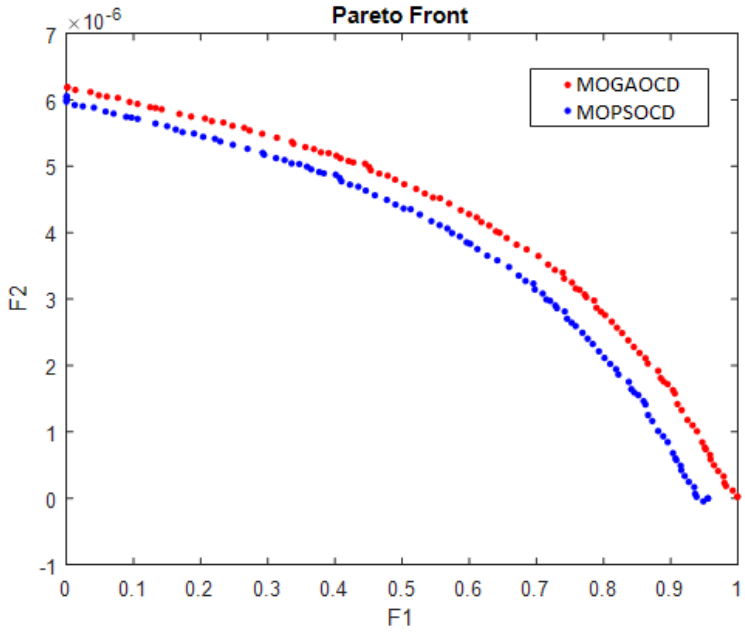

Figure 16. The Pareto front diagram in two MOGAOCD and MOPSOCD algorithms on a 10 kb graph to detect community.

The results of the comparison of the MOGAOCD and MOPSOCD algorithms on the 100kb graph, including 777 vertices and 399 edges are presented in Figure 17. This figure shows the values of the two objectives obtained from the 100-times implementation of the MOGAOCD and the MOPSOCD algorithms. This figure also shows the optimization of the Pareto front for both algorithms. The red dot in the image shows the values of f_1 and f_2 for the solutions of the MOGAOCD and the blue points representing the same values for the solution of the generated the MOPSOCD. According to this figure, the MOGAOCD is able to outperform the MOPSOCD algorithm. As shown in Figure 17, both the MOGAOCD and the MOPSOCD algorithms have the same performance for the values of f_1 in the range between 0 and 0.1. However, from the value of $f_1 = 0.1$ to $f_1 = 1$, our proposed algorithm has a better performance than the MOPSOCD in optimizing the solution.

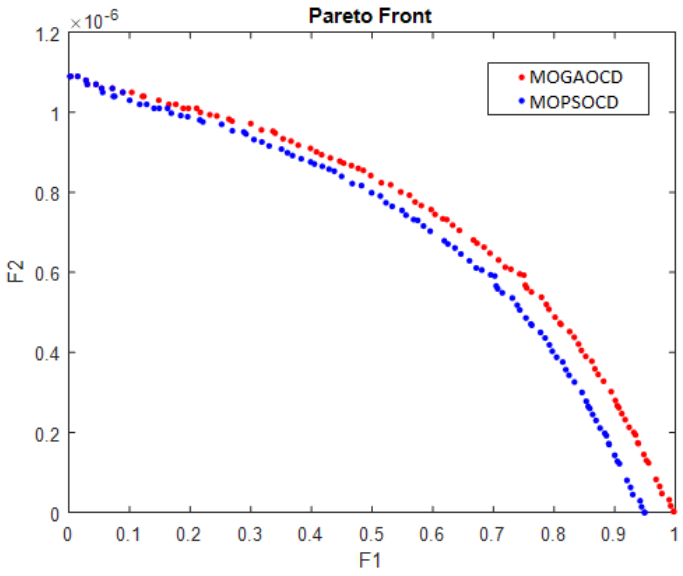

Figure 17. The Pareto front diagram in two MOGAOCD and MOPSOCD algorithms on a 100 kb graph to detect community.

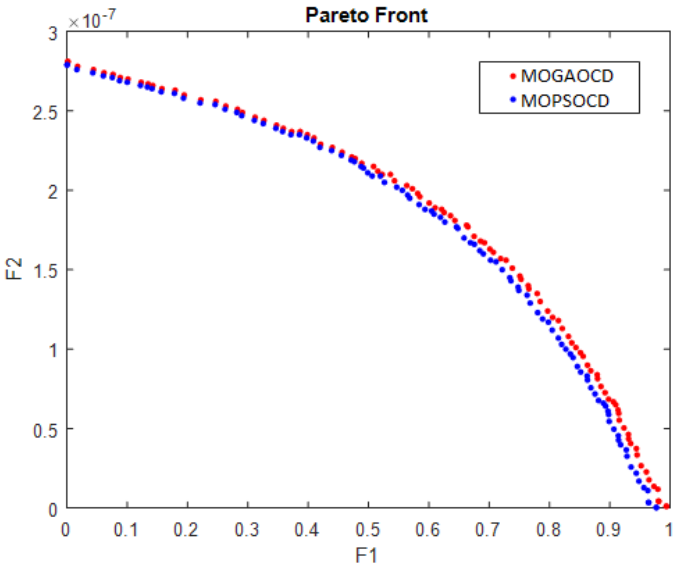

Figure 18. The Pareto front diagram in MOGAOCD and MOPSOCD algorithms on a 500 kb graph to detect community.

As shown in Figure 17, the red dot contains the best responses, since it has the highest modularity and the average weight of the vertices. By viewing the position of each solution, including red and blue points, the modularity value and the average weights of vertices in each solution are observable. The diagram in the Figure 18, illustrates the promising performance of the MOGAOCD in community detection in a 100 kb graph.

In Figure 18, solutions generated by the MOGAOCD and the MOPSOCD algorithms in 500 kb graph are shown. In accordance with Figure 11, the red dot represents the Pareto front solutions via MOGAOCD, and the blue points represent the Pareto front solutions in the MOPSOCD. As the

generated diagram shows, the red dot contains the best responses since it has the highest modularity and the average weight of the vertices. By viewing the position of each solution including red and blue points, the modularity value and the average weights of the vertices in each solution can be observed. Note that some of the points (solutions) overlap each other which indicated the proximity of their values.

Figure 19 shows the solutions generated by the MOGAOCD and MOPSOCD algorithms in 1 Mbp graph. According to this figure, the red dots indicate the solutions of the Pareto front created by the MOGAOCD and the blue points, representing the Pareto front in the MOPSOCD algorithm. This figure depicts the optimization of the Pareto front for both algorithms. According to Figure 19, MOGAOCD is able to perform better than MOPSOCD algorithm. As shown in this figure, MOGAOCD and MOPSOCD algorithm for the values $f_1 = [0, 0.6]$ have the same functionality. However, for the values $f_1 = [0.6, 1]$ the proposed algorithm has better performance than MOPSOCD in discovering the optimal solutions. Therefore, in general, the performance of the MOGAOCD algorithm is better than the MOPSOCD algorithm in community detect in the 1 Mbp graph.

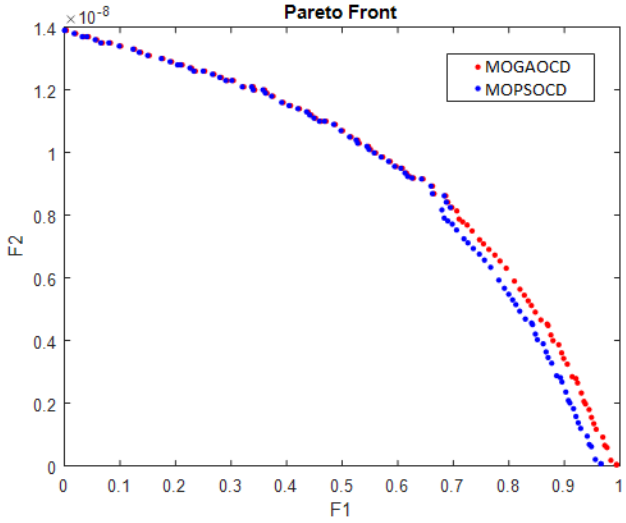

Figure 19. The Pareto front diagram in two MOGAOCD and MOPSOCD algorithms on 1 Mbp graph to detect community.

3.3. The performance of the proposed method in community detection on the ESCs genomic graph

The Figure 20 shows all the community detected by the MOGAOCD and MOPSOCD algorithms in the 5 kb graph. The graph has 333 vertices and 202 edges. According to the figure, the red points represent the solutions of the Pareto front produced by the MOGAOCD and the blue points represent the solutions of the MOPSOCD algorithm. According to the Figure 20, the proposed algorithm is able to perform better than the MOPSOCD algorithm. As seen, the MOGAOCD and MOPSOCD algorithms for the values $f_1 = [0, 0.3]$ have the same functionality. However, for the values of $f_1 = [0.3, 1]$ the MOGAOCD algorithm has better performance than MOPSOCD in discovering optimal solutions. Therefore, it can be concluded that the performance of the MOGAOCD algorithm is better than the MOPSOCD algorithm in community detection in the 5 kb graph.

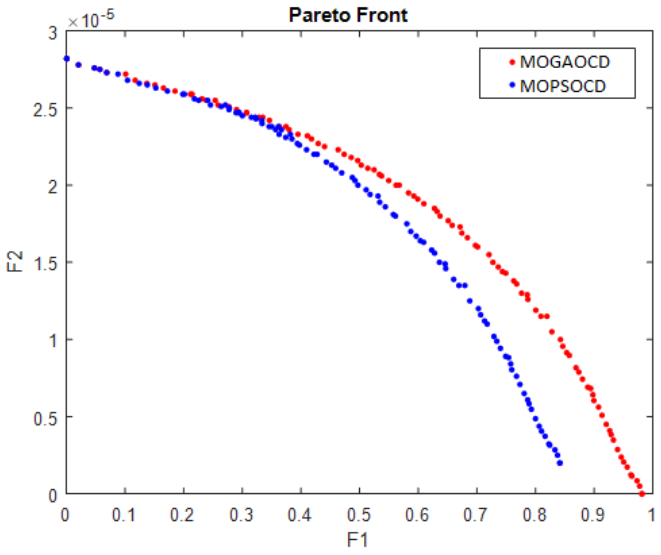

Figure 20. The Pareto front diagram in MOGAOCD and MOPSOCD algorithms on a 5 kb graph to detect community.

Table 4. Comparison between two MOGAOCD and MOPSOCD algorithms in accordance with three criteria.

Graph	MOPSOCD			MOGAOCD		
	Average values of *f1*	Average values of *f2*	Number of community detected	Average values of *f1*	Average values of *f2*	Number of community detected
5 kbp	0.48	1.65×10^{-5}	123	0.58	1.79	164
10 kbp	0.55	3.47×10^{-6}	1121	0.59	3.63×10^{-6}	1632
100 kbp	0.53	6.53×10^{-7}	327	0.57	6.55×10^{-7}	378
500 kbp	0.59	1.58×10^{-7}	438	0.61	1.6×10^{-7}	464
1 Mbp	0.56	8.36×10^{-9}	181	0.57	8.37×10^{-9}	207

3.4. Analysis of results in community detection

In this section, the average results obtained from 100 implementations of MOGAOCD and MOPSOCD algorithms are analyzed on five benchmarks (5, 10, 100, 500 kb, 1 Mb) according to the three criteria of f_1, f_2, and the number of communities detected. According to Table 4, the MOGAOCD algorithm in 5 and 10 kb graphs in all three criteria has a better performance than the MOPSOCD algorithm.

Meanwhile, the MOGAOCD algorithm in the 100, 500 kb, and 1 Mb graphs in accordance with the three evaluation criteria yields a slightly better performance than the MOPSOCD algorithm. Also, in many implementations, according to the results represented in the previous Sections, the MOGAOCD algorithm demonstrates better results compared to the MOPSOCD algorithm. As a result, it can be concluded that the MOGAOCD in graphs with smaller-size fragmentation has a better performance than genomic graphs with larger-size fragmentation in community detection.

4. Innovation of research

This research deals with the current unsolved challenge in Genetics, that is, community detection in the genomic graph arisen from the inter-genome interactions. In view of that, we presented a Pareto-based genetic multi-objective algorithm. In the genomic graphs, nodes, edges and weights are respectively regions of genome, interactions between nodes, and the number of interactions. The related challenge is that the number of communities is not known in advance, with the corresponding graph having no definite topology. Also, there should be graph regions in the detected community with maximum weights, namely, the most interactions. This means that detection of the community hinges on the edge's weights. In these conditions, an algorithm that is capable of detecting the community when the nodes have the greatest weights is required. Thus, the weights of the edges between the nodes are put in the community. The present article offers a bi-objective heuristic algorithm based on genetics to solve the problem by detecting the community in five genomic graphs using two objective functions f_1 and f_2. In the following, benefits and drawback of proposed algorithm is described.

Benefits of proposed algorithm: (1) Consideration of objectives in decision of a solution. (2) Optimization operations to decide the best solution. (3) Detecting of community without knowing the number of communities at first, and taking in to account the sum of weights of edges between the nodes. (4) Helping the science of Genetics to detect and treat diseases by detecting genomic communities which interact strongly. We believe the drawback of our method is that despite better performance, it still suffers from high computational and time complexity and further improvement is required.

5. Conclusion

Transcriptional regulatory elements can target protein coding and non-coding genes in different genomic distances through chromatin interactions. Chromosome conformation capture technique (Hi-C) enables researchers to study the three-dimensional (3D) conformation of chromosomes in the cell nucleus and identify such regulatory interacting regions. Here, we proposed MOGAOCD as a new algorithm for community detection in chromosome conformation capture (Hi-C) data. MOGAOCD is able to identify sets of genomic interacting regions from Hi-C data, acting as a co-interaction regions. This would to study spatially colocalized genomic regions that are functionally relevant. Identified clusters by MOGAOCD share transcription factors and are enriched for transcriptional machinery, suggesting that chromosome intermingling regions play a key role in genome regulation. Our method provides a unique quantitative framework that can be broadly applied on chromosome conformation capture from different cells/tissues.

Acknowledgements

This work was supported by grants to HAR and HP. HAR is currently supported by the UNSW Scientia Fellowship.

The paper is extracted from a PhD thesis compiled by student Mohammadjavad Hosseinpoor. MH and HAR designed the study. MH, HP, SN, VR and KB wrote the paper. MH, HP, HAR, AD, and AB revised the manuscript. HR and MH provided the data. MH carried out tool implementation. MH, HP generated all figures and tables. HAR and AB were not involved in any analysis, writing, figures

and tables generation of the paper. MH, HP, SN, VR and KB performed statistical analyses. All authors have read and approved the final version of the paper.

We Acknowledge the AI-enabled Processes (AIP) Research Centre (https://aip-research-center.github.io) for funding part of this research.

Conflict of Interest

The authors declare that they have no financial or non-financial competing interests.

References

1. J. Wang, J. Xie, Z. Tu, J. Wang, W. Pan, J. Hu, et al., Cloning and expression analysis of the nuclear factor erythroid 2- related factor 2 (Nrf2) gene of grass carp (Ctenopharyngodon idellus) and the dietary effect of Eucommia ulmoides on gene expression, *Aquacult. Fish.,* **3** (2018), 196–203.
2. C. Essien, B. K. Via, G Acquah, T. Gallagher, T. McDonald, L. Eckhardt, Effect of genetic sources on anatomical, morphological, and mechanical properties of 14-year-old genetically improved loblolly pine families from two sites in the southern United States, *J. For. Res.,* **29** (2018), 1519–1531.
3. I. Cabreros, E. Abbe, A. Tsirigos, *Detecting community structures in Hi-C genomic data*, 2016 Annual Conference on Information Science and Systems (CISS), 2016, 584–589. Available from: https://ieeexplore_ieee.xilesou.top/abstract/document/7460568.
4. A. F.Siahpirani, F. Ay, S. Roy, A multi-task graph-clustering approach for chromosome conformation capture data sets identifies conserved modules of chromosomal interactions, *Genome biol.,* **17** (2016), 114.
5. Z. Li, L. He, Y. Li, A novel multiobjective particle swarm optimization algorithm for signed network community detection, *Appl. Intell.,* **44** (2016), 621–633.
6. A. Beheshti, B. Benatallah, A. Tabebordbar, H. R. Motahari-Nezhad, M. C. Barukh, R. Nouri, Datasynapse: A social data curation foundry, *Distrib. Parallel Databases*, **37** (2019), 351–384.
7. V. Kawadia, S. Sreenivasan, Sequential detection of temporal communities by estrangement confinement, *Sci. Rep.,* **2** (2012), 794.
8. Q. C. Zhang, D. Petrey, J. I. Garzón, J. I. Garzón, L. Deng, B. Honig, PrePPI: A structure-informed database of protein-protein interactions, *Nucleic Acids Res.,* **41** (2013), D828–D833.
9. G. Pan, W. Zhang, Z. Wu, S. Li, Online community detection for large complex networks, *Plos One*, **9** (2014), e102799.
10. N. K. Fox, S. E. Brenner, J. M. Chandonia, SCOPe: Structural Classification of Proteins-extended, integrating SCOP and ASTRAL data and classification of new structures, *Nucleic Acids Res.,* **42** (2014), D304–D309.
11. J. W. Hoskins, J. Jia, M. Flandez, H. Parikh, W. Xiao, I. Collins, et al., Transcriptome analysis of pancreatic cancer reveals a tumor suppressor function for HNF1A, *Carcinogenesis,* **35** (2014), 2670–2678.
12. I. Masoudiasl, S. Vahdat, S. Hessam, S. Shamshirband, H. Alinejad-Rokny, Proposing an Integrated Method based on Fuzzy Tuning and ICA Techniques to Identify the Most Influencing Features in Breast Cancer, *Iran. Red Crescent Med. J.*, **21** (2019), e92077.

13. M. Yasrebi, A. Eskandar-Baghban, H. Parvin, M. Mohammadpour, Optimisation inspiring from behaviour of raining in nature: Droplet optimisation algorithm, *Int. J. Bio-inspired Comput.,* **12** (2018), 152–163.
14. H. Parvin, H. Alinezad, N. Seyedaghaee, S. Parvin, A heuristic scalable classifier ensemble of binary classifier ensembles, *J. Bioinf. Intell. Control*, **1** (2012), 163–170.
15. B. Minaei-Bidgoli, H. Parvin, H. Alinejad-Rokny, H. Alizadeh, W. F. Punch, Effects of resampling method and adaptation on clustering ensemble efficacy, *Artif. Intell. Rev.*, **41** (2014), 27–48.
16. J. S. Bernardes, F. R. J. Vieira, L. M. M. Costa, G. Zaverucha, Evaluation and improvements of clustering algorithms for detecting remote homologous protein families, *BMC Bioinf.,* **16** (2015), 34.
17. H. Parvin, H. Alinejad-Rokny, B. Minaei-Bidgoli, S. Parvin, A new classifier ensemble methodology based on subspace learning, *J. Exp. Theor. Artif. Intell.*, **25** (2013), 227–250.
18. J. Creusefond, T. Largillier, S. Peyronnet, *On the evaluation potential of quality functions in community detection for different contexts,* International Conference and School on Network Science, Springer, Cham, 2016, 111–125. Available from: https://link_springer.xilesou.top/chapter/10.1007/978-3-319-28361-6_9.
19. J. Chowdhary, F. E. Löffler, J. C. Smith, Community detection in sequence similarity networks based on attribute clustering, *Plos One,* **12** (2017), e0178650.
20. H. Parvin, M. MirnabiBaboli, H. Alinejad-Rokny, Proposing a Classifier Ensemble Framework Based on Classifier Selection and Decision Tree, *Eng. Appl. Artif. Intell.*, **37** (2015), 34–42.
21. G. B. Orgaz, S. Salcedo-Sanz, D. Camacho, A Multi-Objective Genetic Algorithm for overlapping community detection based on edge encoding, *Inf. Sci.,* **462** (2018), 290–314.
22. B. Kong, W. Wu, N. Valkovska, C. Jäger, X. Hong, U. Nitsche, et al., A common genetic variation of melanoma inhibitory activity-2 labels a subtype of pancreatic adenocarcinoma with high endoplasmic reticulum stress levels, *Sci. Rep.,* **5** (2015), 8109.
23. U. Maulik, S. Mallik, A. Mukhopadhyay, S. Bandyopadhyay, Analyzing large gene expression and methylation data profiles using StatBicRM: Statistical biclustering-based rule mining, *PloS One,* **10** (2015), e0119448.
24. G. Reali, M. Femminella, E. Nunzi, D. Valocchi, Genomics as a service: A joint computing and networking perspective, *Comput. Networks,* **145** (2018), 27–51.
25. S. Nejatian, R. Omidvar, H. Mohamadi, A. E. Baghbani, V. Rezaie, H. Parvin, An optimization algorithm based on behavior of see-see partridge chicks, *J. Intell. Fuzzy Syst.,* **33** (2017), 3227–3240.
26. M. M. Jenghara, H. E. Komleh, H. Parvin, Dynamic protein-protein interaction networks construction using firefly algorithm, *Pattern Anal. Appl.,* **21** (2018), 1067–1081.
27. N. Servant, N. Varoquaux, B. R. Lajoie, E. Viara, C. J. Chen, J. P. Vert, et al., HiC-Pro: An Optimized and Flexible Pipeline for Hi-C Data Processing, *Genome Biol.*, **16** (2015), 259.
28. M. E. J. Newman, Detecting community structure in networks, *Eur. Phys. J. B,* **38** (2004), 321–330.
29. H. Parvin, B. Minaei-Bidgoli, H. Alinejad-Rokny, A New Imbalanced Learning and Dictions Tree Method for Breast Cancer Diagnosis, *J. Bionanosci.*, **7** (2013), 673–678.
30. H. Parvin, B. Minaei-Bidgoli, H. Alinejad-Rokny, W. F. Punch, Data weighing mechanisms for clustering ensembles, *Comput. Electr. Eng.*, **39** (2013), 1433–1450.

31. R. Javanmard, K. Jeddisaravi, H. Rokny, Proposed a New Method for Rules Extraction Using Artificial Neural Network and Artificial Immune System in Cancer Diagnosis, *J. Bionanosci.*, **7** (2013), 665–672.
32. T. Sureshkumar, M. Lingaraj, B. Anand, T. Premkumar, Non-dominated sorting particle swarm optimization (NSPSO) and network security policy enforcement for Policy Space Analysis, *Int. J. Commun. Syst.,* **31** (2018), e3554.
33. T. Sureshkumar, B. Anand, T. Premkumar, Efficient Non-Dominated Multi-Objective Genetic Algorithm (NDMGA) and network security policy enforcement for Policy Space Analysis (PSA), *Comput. Commun.,* **138** (2019), 90–97.
34. S. Boyd, L. Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004.
35. H. Alinejad-Rokny, E. Sadroddiny, V. Scaria, Machine learning and data mining techniques for medical complex data analysis, *Neurocomputing*, **276** (2018), 1.
36. H. Alinejad-Rokny, Proposing on optimized homolographic motif mining strategy based on parallel computing for complex biological networks, *J. Med. Imaging Health Inf.*, **6** (2016), 416–424.
37. H. Alinejad-Rokny, H. Pourshaban, A. G. Orimi, M. M. Baboli, Network motifs detection strategies and using for bioinformatic networks, *J. Bionanosci.,* **8** (2015), 353–359.
38. H. Parvin, H. Alizadeh, S. Parvin, H. Shirgahi, A new conditional invariant detection framework (CIDF), *Sci. Res. Essays*, **8** (2013), 265–273.
39. H. Motameni, H. Alizadeh, M. M. Pedram, Using sequential pattern mining in discovery DNA sequences contain gap, *Am. J. Sci. Res.,* **14** (2011), 72–78.
40. A. Amir, L. Dey, A k-mean clustering algorithm for mixed numeric and categorical data, *Data Knowl. Eng.,* **63** (2007), 503–527.
41. M. Ahmadinia, M. R. Meybodi, M. Esnaashari, H. Alinejad-Rokny, Energy-efficient and multi-stage clustering algorithm in wireless sensor networks using cellular learning automata, *IETE J. Res.,* **59** (2013), 774–782.
42. A. Beheshti, B. Benatallah, R. Nouri, A. Tabebordbar, CoreKG: A knowledge lake service, *Proc. VLDB Endowment*, **11** (2018), 1942–1945.

AIMS Press