Single and multiobjective evolutionary algorithms for clustering biomedical information with unknown number of clusters

María Eugenia Curi[1], Lucía Carozzi[1], Renzo Massobrio[1], Sergio Nesmachnow[1], Grégoire Danoy[2], Marek Ostaszewski[3], and Pascal Bouvry[2]

[1] Universidad de la República, Uruguay
{maria.curi,lucia.carozzi,renzom,sergion}@fing.edu.uy
[2] FSTC/CSC-ILIAS, University of Luxembourg, Luxembourg
{gregoire.danoy,pascal.bouvry}@uni.lu
[3] LCSB, University of Luxembourg, Luxembourg
marek.ostaszewski@uni.lu

Abstract. This article presents single and multiobjective evolutionary approaches for solving the clustering problem with unknown number of clusters. Simple and ad-hoc operators are proposed, aiming to keep the evolutionary search as simple as possible in order to scale up for solving large instances. The experimental evaluation is performed considering a set of real problem instances, including a real-life problem of analyzing biomedical information in the Parkinson's disease map project. The main results demonstrate that the proposed evolutionary approaches are able to compute accurate trade-off solutions and efficiently handle the problem instance involving biomedical information.

Keywords: clustering, biomedical information, multiobjective

1 Introduction

The clustering problem aims at grouping a set of elements in such a way that elements in the same group (*cluster*) are more similar to each other than to the elements in other clusters [1]. Similarity between elements is evaluated according to a predefined similarity metric to be maximized. Clustering is one of the most important unsupervised learning problems, which models many other problems dealing with finding a structure in a given set of data.

In particular, biomedical research demands dealing with a large number of concepts linked by complex relationships, which are often represented using large graphs. In order to process and understand these knowledge bases, researchers need reliable tools for visualizing and exploring large amounts of data conveniently. In order to get a deep understanding of such knowledge bases, concepts with similar characteristics need to be accurately grouped together.

Clustering is an NP-hard optimization problem [2] that has been thoroughly studied in the last 30 years [3]. Heuristics and metaheuristics [4] have been

applied to solve the clustering problem efficiently. Among them, evolutionary algorithms (EAs) have proven to be accurate and powerful methods [5, 6].

This article addresses two formulations of the clustering problem, a first one in which the number of clusters is known in advance, and a multiobjective variant which simultaneously maximizes the similarity between elements in the same cluster and minimizes the number of clusters. Three EAs are presented, two for the single objective and one for the multiobjective clustering problem. The proposed EAs are compared against several methods from the related literature. The evaluation focuses on a large problem instance from the Parkinson's disease map project [7], a research initiative that proposes building a knowledge repository to describe molecular mechanisms related to that condition [8]. The repository compiles literature-based information about Parkinson's disease and organizes the main concepts and contents in an easy to explore and freely accessible map, including experimental data, drug targets and other concepts.

The article is organized as follows. Section 2 presents the single and multiobjective clustering problem formulation and reviews related works on heuristics and metaheuristics applied to the clustering problem. The proposed EAs are described in Section 3 and the experimental evaluation is reported in Section 4. Finally, Section 5 presents the conclusions and the main lines for future work.

2 Clustering problem and related work

This section defines the clustering problem in both its single and multiobjective variants and reviews related works.

2.1 Problem formulation

Let us consider the following elements:

- The set $E = \{e_1, e_2, ..., e_n\}$ of elements to be grouped.
- The function $s : E \times E \rightarrow [0, 1]$; $s(e_i, e_j)$ is the similarity between e_i and e_j. The following conditions hold: $\forall e_i, e_j, s(e_i, e_j) = s(e_j, e_i)$ and $s(e_i, e_i) = 1$.
- An integer $k > 0$, which indicates the number of clusters to consider for grouping elements (*only for the single-objective version of the problem*).

The clustering problem consists in assigning the elements in E to a set of groups (*clusters*) $G = \{G_1, ..., G_k\}$; $G_i = \{c_i\} \cup \{e_m / s(e_m, c_i) \leq s(e_m, c_j) \forall e_m \in E, c_j, c_i \in C, i \neq j\}$; $C \subseteq E$, $|C| = k$ is the set of centers of the groups. The following properties hold: a) *cluster index in* $[1, k]$) $\forall (i, j), i \neq j : 1 \leq i, j \leq k$, and b) *clusters are disjoint sets* $G_i \cap G_j = \emptyset$.

The goal of the single objective version of the problem is to maximize the *total similarity* metric (TS) defined in Equation 1.

$$\max \quad TS = \sum_{e_i \in E} \max_{c_i \in C} \quad s(e_i, c_i) \tag{1}$$

In the multiobjective version of the problem, the goal is to simultaneously maximize the value of TS and minimize the number of clusters k.

2.2 Related work

Many articles have presented heuristic and metaheuristic methods applied to the clustering problem. Early works considered the single objective version of the problem, based on minimizing the distance or maximizing similarity.

Das et al. [9] reviewed the application of metaheuristics for clustering problems. Trajectory-based metaheuristics offer limited problem solving capabilities, mainly due to scalability issues when solving large problem instances. Deng and Bard [10] applied GRASP for the Capacitated Clustering Problem, which proposes grouping elements in clusters, where each cluster has capacity constraints (minimum and maximum number of elements). GRASP was able to find optimal solutions for the problem instances with 30 and 40 nodes, and outperformed solutions found using CPLEX when using an execution time limit of one hour.

Early proposed EAs did not follow an explicit multiobjective approach. Sheng and Liu [6] compared k-medoids, local search, and Hybrid K-medoid Algorithm (HKA) over two datasets (517 elements/10 groups, and 2945 elements/30 groups). HKA obtained the best results on the largest problem instance and slightly better results for the small test problem. The EA by Cowgill et al. [11] optimized clustering metrics defined in terms of external cluster isolation and internal cluster homogeneity, improving over hierarchical clustering algorithms considering an internal criterion. Bandyopadhyay and Maulik [12] proposed an EA for clustering with a number of clusters not defined a priori, to analyze several clustering metrics.

Multiobjective EAs (MOEAs) for clustering have been presented in the book by Maulik et al. [13], most of them focused on optimizing two similarity metrics, thus studying different features of the data to analyze. The multiobjective approach by Ripon et al. [14] considered intracluster variation and intercluster distance, without assuming the number of clusters. The experimental analysis over problems with up to 3000 elements, nine classes, and two features, showed improved solutions over a custom NSGA-II.Handl and Knowles [15] proposed multiobjective clustering with automatic k determination (MOCK), considering objective functions based on compactness (deviation) and connectedness of clusters. These are conflicting objectives because the overall deviation improves when using more clusters, but the connectivity decreases. MOCK showed good behavior and scalability when compared with single-objective clustering algorithms. Korkmaz et al. [16] presented a Pareto-based MOEA to find a set of non-dominated clusters considering intracluster variation and the number of clusters. The experimental evaluation was performed over two small standard datasets (150 and 75 elements, with only two attributes), but no numerical results or multiobjective optimization analysis is reported.

Most of the previous works have proposed ad-hoc EAs to address the clustering problem and few of them have solved multiobjective variants. This article contributes with simple EAs and an explicit MOEA designed to scale properly for solving large problem instances, and we focus on a real-life instance considering biomedical information in the context of the Parkinson disease map project.

3 Evolutionary algorithms for clustering

This section describes in detail the single and multiobjective EAs proposed to tackle the clustering problem.

3.1 Single objective EAs

Fitness function. The fitness function computes the sum of similarities between each element and its most similar center, as presented in Section 2.1.

Solution encoding. Two solution encodings are proposed and evaluated. Binary encoding: a solution is represented as a binary vector of length n (the number of elements to be grouped). Each position in the vector represents whether the corresponding element is a group center (1) or not (0). Integer encoding: each solution is a vector of k integers in $[1, N]$, representing the set of cluster centers. Numbers only appear once, as the k clusters must have different centers.

Crossover operators. Two crossover operators were implemented for the binary encoding: Single Point Crossover (SPX) randomly selects a crossover position and exchanges the genes after the crossover point between both parents. Two-Point Crossover (2PX) randomly selects two crossover positions and exchanges the genes located between these two points.

For the integer encoding, three crossover operators were implemented: SPX, Generalized Cut and Splice (GenC&S), and Hybrid Crossover (SPX-GenC&S). GenC&S is a variant of Cut and Splice (C&S) [17] for the clustering problem, to preserve useful features of the information in both parents (Algorithm 1). GenC&S selects a random cutting point cp on one parent and a random integer $s \in [0, k]$. Two lists are created, sorted by similarity with the element on position cp in parent1: LP1 (elements on parent1) and LP2 (elements in parent2). The first s elements in LP1 are copied to offspring1 and the $k - s$ remaining elements are copied from LP2, if their similarity to elements already copied to offspring1 is smaller than the input parameter ε. If less than k centers are copied to offspring1, the solution is completed with randomly selected centers. SPX-GenC&S uses a single random number p instead of cp and s. Elements before p in parent1 are copied to offspring1 (like in SPX), and the $k - p$ remaining elements in offspring1 are copied from parent2, if their similarity to elements already copied to offspring1 is smaller than ε (like in GenC&S). If less than k centers are copied to offspring1, the solution is completed with randomly selected centers.

Mutation operators. Five mutation operators were implemented. For binary encoding, Bit Flip Mutation changes encoded values by the opposite binary value; Add Mutation changes data points to centers; and Delete Mutation changes centers to data points. For integer encoding, One Gene Mutation changes elements to another that is not included in the solution (randomly selected according to a uniform distribution in the set E) and Adapted One Gene Mutation changes an element in the encoding to the most similar element, found by applying the following search: all elements in the solution are processed, and the similarity to

Algorithm 1 GenC&S crossover for the clustering problem (integer encoding)

```
1: Input: parent1, parent2, ε; Output: offspring1
2: cp = rand(0,k)
3: s = rand(0,k)
4: cp_element = parent1[cp]
5: offspring1.add(cp_element)
6: LP1 = sortAscending(parent1,cp_element)
7: LP2 = sortAscending(parent2,cp_element)
8: for i = 0 to s − 1 do          ▷ Copy the first s elements from LP1 to offspring1
9:  offspring1.add(LP1[i])
10: end for
11: for j = 0 to k − s do      ▷ Copy the first N − s elements from LP2 to offspring1
12:  if similarity(LP2[j],offspring1)< ε then                    ▷ not too close
13:   offspring1.add(LP2[j])                                     ▷ already in offspring1
14:  end if
15: end for
16: while offspring1.length() < k do              ▷ Complete with random elements
17:  new_center = rand(0,N)
18:  offspring1.add(new_center)
19: end while
```

the element being mutated is evaluated. The best similarity value (γ) is stored and the new center is selected to have a similarity less than γ.

Corrective function. Some evolutionary operators do not guarantee to preserve the number of centers in a solution. A simple corrective function is applied both for binary and integer encodings. For binary encoding, if the number of 1s in the solution is not k, random centers are added or deleted until the solution becomes feasible. For integer encoding, if the same element appears more than once in the vector, each repeated element is replaced with another chosen randomly (uniform distribution) among elements that are not already centers.

Population initialization. The individuals in the population are randomly generated following a uniform distribution in $\{0, 1\}$ (binary encoding) and a uniform distribution in the set of centers C (integer encoding). The initialization procedure generates feasible solutions by applying the corrective function to each individual in the initial population.

3.2 Multiobjective EA

A variant of NSGA-II [18] was implemented to solve the multiobjective variant of the clustering problem. Following an incremental approach, the encoding and evolutionary operators that achieved the best results in the comparative analysis of the single objective EA for the problem were used in the proposed NSGA-II: binary encoding, SPX, and Delete Mutation.

In the multiobjective problem, the solution with all genes set to 0 is not feasible, since it does not represent any grouping at all. To avoid this situation, the corrective function randomly adds a center to the solution. The initial pop-

ulation is randomly generated following a uniform distribution in $[0, 1]$ and the corrective function is applied to the generated individuals.

4 Experimental evaluation

This section describes the evaluation of the proposed EAs for clustering.

4.1 Problem instances

A total number of 13 problem instances were used to evaluate the proposed EAs. These instances correspond to clustering problems arising in different fields of study, including two instances that model the Parkinson's disease map:

- Instance #1 consists of hydrometric data from 46 basins in Uruguay [19].
- Instances #2 to #8 and #10 to #12 are from the Knowledge Extraction based on Evolutionary Learning dataset [20], a data repository for classification problems. These instances have between 80 and 846 elements each.
- Instances #9 and #13 contain data from the Parkinson's disease map, which visually represents all major molecular pathways involved in the Parkinson disease pathogenesis. Instance #9 has 801 elements. Instance #13 has 3056 elements and it is used to test the performance of the multiobjective approach on a large problem instance containing biomedical information.

4.2 Experimental configuration and methodology

Development and execution platform. The proposed EAs were developed using ECJ [21], an open source framework for evolutionary computation in Java. Experiments were performed on an Intel Core i5 @ 2.7GHz and 8 GB of RAM.

Results evaluation The results computed by the proposed EAs are compared against clustering algorithms from the literature in terms of the objective function (total similarity) and in terms of the relative hypervolume (RHV) metric for the multiobjective variant of the clustering problem. RHV is the ratio between the volumes (in the objective functions space) covered by the computed Pareto front and the volume covered by the true Pareto front. The ideal value for RHV is 1. The true Pareto front—unknown for the problem instances studied—is approximated by the set of non-dominated solutions found in each execution.

The algorithms used in the comparison are:

- *k-medoids* [22], a classic partitional method related to k-means. Clusters are built to minimize the distance between points and the center of the corresponding cluster, according to a given distance metric.
- *Linkage*, an agglomerative hierarchical clustering technique based on building clusters by combining elements of previously defined clusters. A distance function evaluates a relevant similarity metric for the problem and different linkage implementations use different distance functions. The Matlab implementation of single linkage (nearest neighbor), which uses the smallest distance between objects in the two cluster, in the results comparison.

- *Local Search* [6], combining k-medoids and an exhaustive search performed for each cluster. Starting from a randomly selected set of centers, the set of p nearest neighbors is found for each center. A local search is performed over these sets to find a new center that minimizes the distance with all elements. The search ends when no center is changed in two consecutive iterations.
- *Greedy*, which builds clusters iteratively, taking a locally optimal decision in each step. Starting from a randomly selected center, in each step searches for the element with the lowest similarity with the solution already built. This element is included in the solution as a new center. All clusters are recomputed and the procedure is applied until building k clusters.
- *Hybrid EA*, combining an EA and the local search by Sheng and Liu [6] (Algorithm 2). The hybrid EA uses binary encoding, random initialization, tournament selection, Mix Subset Recombination, and Bit Flip Mutation.

Algorithm 2 Generic schema of the hybrid EA for the clustering problem

```
Initialize k centers randomly
while not stopping_criterion do
    [parent1, parent2] = TournamentSelection(P)
    if rand(0,1) > p_C then
        [offspring1, offspring2] = Mix Subset Recombination(parent1, parent2)
    end if
    [offspring1, offspring2] = Bit Flip Mutation(p_M)
    if rand(0,1) > p_LS then
        [offspring1, offspring2] = Local Search()
    end if
end while
return best solution found
```

Statistical analysis. Thirty independent executions of each algorithm were performed over each problem instance to have statistical confidence. For each problem instance, the best and the average fitness value (for the single objective problem) and the average multiobjective metrics (for the multiobjective problem) are reported. The Kolmogorov-Smirnov test is applied to each set of results to assess if the values follow a normal distribution. After that, the non-parametric Kruskal-Wallis test is applied to compare the results distributions obtained by different algorithms. A confidence level of 95% is used for both statistical tests.

4.3 Single objective clustering problem

Parameter settings. The parameter values of each algorithm were configured based on preliminary experiments and suggestions from related works:

- *Single objective EAs*: population size (pop)= 100, crossover probability (p_C) = 0.75, mutation probability (p_M) = 0.01, tournament size = 2, and stopping criterion of 10000 generations.
- *k-medoids*: the algorithm stops when the cluster centers remain unchanged in consecutive iterations.

– *Local search*: size of the search neighborhood = 3 and the stopping criterion is the same as for k-medoids, as recommended by Sheng and Liu [6].
– *Hybrid EA*: $pop = 30$, $p_C = 0.95$, $p_M = 0.02$, $p_{LS} = 0.2$, neighborhood size = 3, tournament size = 2, and stopping criterion of 10000 generations.

Comparison of evolutionary operators. For the binary encoding, two crossovers and three mutations were proposed, generating six possible combinations: SPX and Bit Flip Mutation (*SPX-bit*), SPX and Add Mutation (*SPX-add*), SPX and Delete Mutation (*SPX-del*), 2PX and Bit Flip Mutation (*2PX-bit*), 2PX and Add Mutation (*2PX-add*), and 2PX and Delete Mutation (*2PX-del*). Experimental results showed that *SPX-del* performed better on small problem instances, outperforming the other combinations of evolutionary operators. On medium sized instances #5 and #6, *SPX-bit* computed the best results, while on large instances *2PX-del* achieved the best results. Therefore, the rest of the experimental analysis of the single objective EA using binary encoding focused on these three combinations of evolutionary operators.

For the integer encoding, three crossover operators and two mutations were presented, generating six possible combinations: SPX and One Gene Mutation (*SPX-One*), SPX and Adapted One Gene Mutation (*SPX-Adapt*), SPX-GenC&S Crossover and One Gene Mutation (*SPXGCS-One*), SPX-GenC&S Crossover and Adapted One Gene Mutation (*SPXGCS-Adapt*), GenC&S Crossover and One Gene Mutation (*GCS-One*), and GenC&S Crossover and Adapted One Gene Mutation (*GCS-Adapt*). Results showed that *SPX-One* computed the best results in 7 instances and *GCS-One* in 5 instances, both outperforming the other combinations. Therefore, the rest of the experimental analysis of the single objective EA using integer encoding focused on these two combinations.

Comparison of solution encodings. Table 1 reports the average similarity results computed on 30 independent executions of the proposed EA using binary and integer encoding and the evolutionary operators that achieved the best results in the previous analysis.

Table 1: Average similarity using different encodings and evolutionary operators.

#I	integer encoding		binary encoding		
	SPX-One	*GCS-One*	*SPX-bit*	*SPX-del*	*2PX-del*
#1	**18.66**	**18.66**	**18.66**	**18.66**	**18.66**
#2	**1.96**	**1.96**	**1.96**	**1.96**	**1.96**
#3	12.42	12.44	12.27	**12.46**	**12.46**
#4	16.43	16.41	15.93	**16.50**	**16.50**
#5	78.35	78.16	**78.61**	78.51	78.42
#6	116.18	116.39	**116.45**	115.69	115.34
#7	54.71	54.68	54.80	**54.98**	**54.98**
#8	63.27	63.30	61.10	63.42	**63.43**
#9	673.57	656.56	633.91	**675.20**	**675.20**
#10	37.77	36.49	35.88	**38.22**	**38.22**
#11	235.33	229.58	221.17	**236.11**	**236.11**
#12	32.89	32.08	31.20	**33.23**	**33.23**

Results indicate that the binary encoded EA using *SPX-del* and *2PX-del* significantly outperformed the results computed using integer encoding and *SPX-bit*. There is no significant difference when using *SPX-del* and *2PX-del*, and for simplicity, the rest of the experimental evaluation was performed using *SPX-del*.

Comparison against other algorithms. The proposed EA with binary encoding, SPX, and delete mutation was compared against the baseline algorithms. Table 2 reports the average similarity computed over 30 independent executions of each algorithm for the 12 problem instances (the best results are marked in bold). The Kolmogorov-Smirnov test was performed on the results' distributions. In most cases, the test allowed rejecting–with 95% confidence–the null hypothesis that the results follow a normal distribution. Therefore, the Kruskal-Wallis test was used to compare the results' distributions computed by each EA (the p-value is reported in the last column). Kruskal-Wallis allows rejecting the null hypothesis that the results computed by all algorithms follow the same distribution.

Table 2: Comparison of average similarity against other algorithms.

instance	*greedy*	*linkage*	*k-medoids*	*local search*	*hybrid EA*	*SPX-del*	*p-value K-W*
#1	7.28	17.01	17.03	15.49	**18.66**	**18.66**	$< 10^{-15}$
#2	1.12	1.65	1.95	1.70	**1.96**	**1.96**	$< 10^{-15}$
#3	5.77	10.18	12.14	10.50	12.45	**12.46**	$< 10^{-15}$
#4	7.41	14.04	16.00	13.23	16.22	**16.50**	$< 10^{-15}$
#5	47.69	76.08	76.47	69.11	**78.62**	78.51	$< 10^{-15}$
#6	83.61	109.68	116.30	108.86	**116.45**	115.69	$< 10^{-15}$
#7	29.31	50.77	54.98	41.68	**54.98**	**54.98**	$< 10^{-15}$
#8	31.81	62.25	62.51	52.99	63.24	**63.42**	$< 10^{-15}$
#9	499.54	523.19	667.94	615.64	661.48	**675.20**	$< 10^{-15}$
#10	22.90	30.61	37.09	32.94	36.73	**38.22**	$< 10^{-15}$
#11	170.65	198.75	236.10	205.96	229.56	**236.11**	$< 10^{-15}$
#12	22.80	27.02	32.85	28.56	33.10	**33.23**	$< 10^{-15}$

The proposed EA outperformed all other algorithms, computing the best average results in 10 instances. Improvements were up to 9.5% over k-medoids and 156.2% over greedy. The proposed EA also improved over Linkage in up to 29.1% and over the local search on of 31.9%. Finally, the improvements against the hybrid algorithm are smaller. In the best case (instance #10) the proposed EA outperformed the hybrid EA in up to 4.0% (2.3% on average).

4.4 Multiobjective clustering problem

Parameters setting. The parameters of the proposed MOEA were defined based on preliminary experiments: $pop = 100$, $p_C = 0.75$, $p_M = 0.01$, tournament of size 2, and a stopping criteria of 1000 generations.

Numerical results. The best EA for the single objective clustering problem (i.e., using SPX and delete mutation) and k-medoids were used to compare the NSGA-II results. 30 independent executions of each algorithm were executed, changing the number of clusters for the single objective algorithms.

Figures 1 and 2 show sample Pareto fronts computed by the proposed MOEA and the best solutions computed by k-medoids and in 30 independent executions of the single objective EA using different numbers of clusters. These are representative results for the set of problem instances solved.

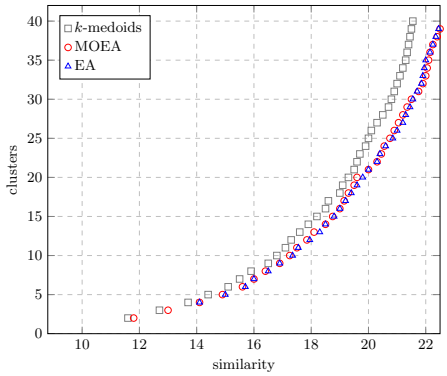

Fig. 1: Pareto fronts for instance #4

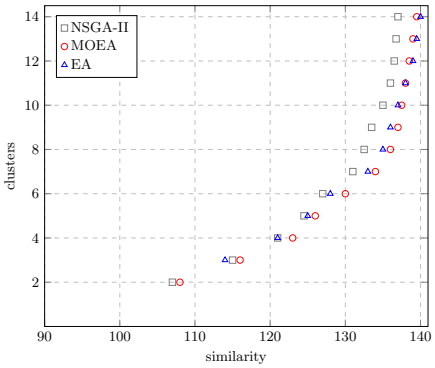

Fig. 2: Pareto fronts for instance #6

Results showed that for small number of clusters there is no significant difference in the solutions computed by EA and MOEA. Both evolutionary approaches improve over k-medoids. As the number of groups increases, the MOEA is able to found solutions with better similarity values than the single objective EA, and both significantly improves over the k-medoids results. In addition, the MOEA is able to obtain a Pareto front of solutions with different trade-off values in a single execution, while several executions (each one for a different number of clusters) are needed for the single objective EA and k-medoids. Therefore, the MOEA is useful for a decision-maker to be able to visualize several groupings with different trade-offs between the problem objectives and select the one that better captures the problem features. This is especially relevant in the case of biomedical information, where the number of clusters is particularly difficult to define a priori for a given dataset.

The RHV results over 30 independent executions, reported in Table 3, indicated that the proposed MOEA is robust and computes accurate Pareto fronts for the problem instances studied. The average RHV was 0.99, the maximum difference from the ideal RHV was 0.02 (instances #6 and #12), and the optimum value of 1.00 was achieved for three problem instances.

Table 3: RHV values obtained by the proposed algorithms.

MOEA		EA		k-medoids	
average	*best*	*average*	*best*	*average*	*best*
0.99	1.00	0.96	1.00	0.83	0.92

Regarding the problem instances from the Parkinson's disease map, the proposed EAs allowed to compute accurate configurations that provide different trade-offs between the problem objectives. Using the evolutionary approaches,

several new possible clusterings have been found. These clusters provide novel promising information, different to the current manually built solutions (see the project website at http://wwwen.uni.lu/lcsb/research/parkinson_s_disease_map).

Overall, considering the complete set of problem instances, EA and MOEA were able to improve over k-medoids 15.8% and 14.1% in average (respectively), and up to 31.4% and 27.0% in the best case. The best improvements were obtained in the problem instances with larger number of elements, clearly demonstrating the good scalability behavior of the proposed evolutionary approaches. The best improvement of EA over MOEA was 8.7% and the best improvement of MOEA over EA was 4.4%.

5 Conclusions and future work

This article presented evolutionary algorithms applied to the clustering problem in its single and multiobjective variants, with unknown number of clusters. This is a very important problem in many research areas that involve dealing with large volumes of information to be categorized and grouped.

The proposed evolutionary algorithms were conceived to apply simple and ad-hoc operators, trying to keep the search as straightforward as possible in order to scale up for solving large instances of the clustering problem.

The experimental evaluation was performed considering a set of real problem instances, including one problem consisting of biomedical information in the context of the Parkinson disease map project. The main results from the experimental analysis indicate that the proposed evolutionary algorithms are able to compute accurate solutions for the problem instances studied. The evolutionary approaches outperform several algorithms of the related literature. In the single objective clustering problem, the proposed evolutionary algorithm is able to compute the best average result in 10 out of 12 problem instances. For the multiobjective clustering problem, the proposed evolutionary algorithm is able to compute accurate Pareto fronts, which offer decision-makers solutions with different trade-offs between the problem objectives.

The evolutionary approach is especially helpful for organizing biomedical information in the case of the Parkinson's disease map project. The proposed EAs are able to find accurate organizations for the data, which provide different trade-offs between the problem objectives and allow capturing different features of the information. The computed solutions provide new promising clustering patterns to be examined along the existing ones, manually built by experts.

The main lines of future work include extending the experimental analysis considering datasets from different fields of study. Additionally, a parallel model for EAs should be considered to both reduce execution times and handle bigger datasets. Finally, the possibility of combining the proposed evolutionary algorithms with visualization tools should be studied, in order to help researchers analyze the information in a more intuitive way.

References

1. Kaufman, L., Rousseeuw, P.: Finding groups in data: an introduction to cluster analysis. Wiley, New York (1990)
2. Welch, W.: Algorithmic complexity: threeNP- hard problems in computational statistics. Journal of Statistical Computation and Simulation **15**(1) (1982) 17–25
3. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning. Springer New York (2009)
4. Nesmachnow, S.: An overview of metaheuristics: accurate and efficient methods for optimisation. International Journal of Metaheuristics **3**(4) (2014) 320–347
5. Hruschka, E., Campello, R., Freitas, A., de Carvalho, A.: A survey of evolutionary algorithms for clustering. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) **39**(2) (2009) 133–155
6. Sheng, W., Liu, X.: A hybrid algorithm for k-medoid clustering of large data sets. In: IEEE Congress on Evolutionary Computation. (2004) 77–82
7. University of Luxembourg: Parkinson's disease map project http://wwwen.uni.lu/lcsb/research/parkinson_s_disease_map [November, 2017].
8. K. Fujita et al.: Integrating pathways of Parkinson's disease in a molecular interaction map. Molecular Neurobiology **49**(1) (2014) 88–102
9. Das, S., Abraham, A., Konar, A.: Metaheuristic Clustering. Volume 178 of Studies in Computational Intelligence. Springer (2009)
10. Deng, Y., Bard, J.: A reactive GRASP with path relinking for capacitated clustering. Journal of Heuristics **17**(2) (2011) 119–152
11. Cowgill, M., Harvey, R., Watson, L.: A genetic algorithm approach to cluster analysis. Computers & Mathematics with Applications **37**(7) (1999) 99–108
12. Bandyopadhyay, S., Maulik, U.: Nonparametric genetic clustering: comparison of validity indices. IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews) **31**(1) (2001) 120–125
13. Maulik, U., Bandyopadhyay, S., Mukhopadhyay, A.: Multiobjective Genetic Algorithms for Clustering. Springer Nature (2011)
14. Ripon, K., Tsang, C.H., Kwong, S., Ip, M.K.: Multi-objective evolutionary clustering using variable-length real jumping genes genetic algorithm. In: 18^{th} International Conference on Pattern Recognition. (2006) 3609–3616
15. Handl, J., Knowles, J.: An evolutionary approach to multiobjective clustering. IEEE Transactions on Evolutionary Computation **11**(1) (2007) 56–76
16. Korkmaz, E., Du, J., Alhajj, R., Barker, K.: Combining advantages of new chromosome representation scheme and multi-objective genetic algorithms for better clustering. Intelligent Data Analysis **10**(2) (2006) 163–182
17. Deaven, D., Ho, K.: Molecular geometry optimization with a genetic algorithm. Physical Review Letters **75** (1995) 288–291
18. Deb, K.: Multi-Objective Optimization using Evolutionary Algorithms. John Wiley & Sons (2001)
19. Ministerio de Vivienda Ordenamiento Territorial y Medio Ambiente (Uruguay): Red de estaciones hidrométricas http://www.mvotma.gub.uy [November, 2017].
20. J. Alcalá et al.: KEEL data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework. Journal of Multiple-Valued Logic and Soft Computing **17**(2-3) (2010) 255–287
21. Sean Luke et al.: ECJ 23: a Java-based Evolutionary Computation Research System [Online], https://cs.gmu.edu/ eclab/projects/ecj, accessed March 2017.
22. Kaufman, L., Rousseeuw, P.: Clustering by means of medoids. In: Statistical Data Analysis Based on the L1-Norm and Related Methods. (1987)