Solving Clustering Problems Using Bi-Objective Evolutionary Optimisation and Knee Finding Algorithms

Gustavo Recio
Department of Computer Science
Universidad Carlos III de Madrid, Spain
Email: gustavo.recio@uc3m.es

Kalyanmoy Deb
Kanpur Genetic Algorithms Laboratory
Indian Institute of Technology Kanpur, India
Email: deb@iitk.ac.in

KanGAL Report Number 2013007

***Abstract*—This paper proposes the use of knee finding methods to solve cluster analysis problems from a multi-objective approach. The above proposal arises as a result of a bi-objective study of clustering problems where knee regions on the obtained Pareto-optimal fronts were observed. With increased noise in the data, these knee regions tend to get smother but still comprise the preferred solution. Thus, being the knees what decision makers are interested in when analysing clustering problems, it makes sense to boost the search towards those regions by applying knee finding techniques.**

I. Introduction

Many computing applications involve data analysis in order to determine how the data are organised. Data analysis procedures often focus on grouping the data based on similarity measurements. Cluster analysis, or just clustering, is the grouping of a collection of patterns into subsets, often called clusters, such as the observations in the same cluster are more similar to each other than they are to a pattern belonging to a different cluster. There are many different ways of representing data and measuring the similarity between data elements, which in turn has produced a large number of clustering methods. It is important to distinguish between supervised and unsupervised learning. Both are similar in the sense that they classify data into groups based on similarity measurements. However, in the former, a collection of labelled or pre-classified patterns is provided. These patterns are used to learn the description of classes which in turn are used to classify or assign labels to a new pattern. Typically, the results of a classification problem involving supervised learning are shown as a percentage of accuracy in the classification of new patterns. In unsupervised learning, which is the case of cluster analysis, the problem consists on assigning labels, or grouping, by mere observation of the given unlabelled patterns. Thus, in unsupervised learning, it is not always possible to compute a percentage of accuracy. Due to similar reasons, there are a number of clustering problems for which obtaining a preferred solution is not straight forwards, i.e. as the patterns are unlabelled it becomes complex to establish decision criteria. It is in this context where multiple criteria optimisation plays an important role in finding a set of candidate solutions from which the decision maker will choose a single preferred solution more easily. One of the methods that decision makers apply for systematically choosing a single preferred solution on bi-criteria Pareto-optimal fronts consists on identifying knee regions. Generalising for bi-criteria optimisation problems, knee points are often single preferred solutions as they require an unfavourably large sacrifice in one objective to obtain a small gain in the other and therefore they represent the best trade-off between objectives.

This paper is aimed at investigating the problem of cluster analysis from an evolutionary multi-objective optimisation approach. A multi-objective clustering algorithm has been developed for this purpose and its performance has been validated over artificial data sets. After having observed the shape of the Pareto-optimal front obtained in the experimental part of this work, an improvement of the above algorithm has been proposed. This improvement consists on finding knee regions directly, which, ideally, would turn out in shorter computational times and less evaluations required to find a preferred solution.

This paper is organised as follows. In next section, the current state of the art in the application of evolutionary computation techniques to clustering problems is reviewed. Section III describes the details of the proposed multi objective approach to the cluster analysis problem. The set of experiments carried out in this work and discussion on the obtained results will follow in in Sections IV and V. Section V also deals with the modifications made to the proposed clustering algorithm to find the knee regions directly. Finally, Section VI points out the main findings and concluding remarks of this work.

II. Clustering Algorithms

In the general case for partition algorithms, the number of desired output clusters is fixed before the algorithm define the actual cluster partition. Combinatorial search of the set of possible clustering partitions is computationally very costly for large data sets, hence metaheuristics play an important role in finding the appropriate number of clusters as they iteratively try to improve a candidate solution with regards to a given quality measure, therefore they can search very large spaces of candidate solutions. Evolutionary algorithms are population based metaheuristics commonly used in optimisation. This work will be focused on applying evolutionary computation techniques to cluster analysis problems. Consequently, this

section will only deal with evolutionary approaches, for a complete survey on clustering techniques the reader is redirected to [1].

One of the first approaches to cluster analysis using genetic algorithms was reported in [2] where the author investigated the potential feasibility of evolutionary techniques for the purpose of clustering. In [3] the ideas of k-means were the inspiration to create a genetic implementation called genetic k-means which computational time was later improved by [4] creating the fast genetic k-means algorithm. Since then, many more approaches have been investigated. A survey of genetic algorithm implementations to solve cluster analysis problems can be found in [5].

The problem of cluster analysis has also been approached from a multi-objective point of view using several confronting objectives in order to obtain a set of non-dominated solutions instead of a single solution. In [6] the authors present a multi-objective algorithm for cluster analysis based on cluster ensembles. Non-ensemble multi-objective approaches typically focus on conflicting similarity measurements such as the distance between cluster centroids and a measure of the dispersion within clusters [7], [8], [9]. Variable length chromosomes were used in [10] to investigate cluster analysis problems without prior knowledge of the actual number of clusters. The problem of overlapping cluster detection was investigated using a multi-objective evolutionary clustering approach in [11] which was successful in detecting complex/spiral shaped clusters. Strategies for decision making within the obtained Pareto-optimal front in multi-objective clustering problems have also been investigated in [12].

III. Bi-objective Clustering Algorithm

The simplest and most commonly used partition algorithm is k-means [13], it starts from a random initial partition, i.e. random location of the different cluster centroids, and keeps reallocating the position of these centroids towards the center of mass of the cluster they represent until a convergence criterion is met (e.g. there are no more reallocations of cluster centroids). The algorithm proposed in the present work makes use of the above methodology to create partitions of the data. Different partition sizes will be allowed by using variable length chromosomes in a similar way as reported in [10]. A population of candidate solutions (variable length chromosomes) will be evolved using the well known multi-objective evolutionary approach NSGA-II [14] for which the typical formalisms for dominance and crowding will be adopted.

A. Encoding

A variable chromosome size encoding was used to allow automatic estimation of the number of clusters for each solution. The chromosome representing each solution is formed by a number genes which represent the number of clusters that are considered by that particular solution. Each gene contains information about the spatial location of the centroid representing its cluster. In this manner a solution with three clusters in a two dimensional cluster analysis problem will have three genes, each one of them containing two coordinates therefore the chromosome will be a six cells array. Whereas a solution consisting of, for example, five clusters will be represented by five genes which in turn take the form of a ten cells array. Thus, solutions with two, three and five clusters are represented as the following P, Q and R arrays respectively.

$P1_x$	$P1_y$	$P2_x$	$P2_y$						
$Q1_x$	$Q1_y$	$Q2_x$	$Q2_y$	$Q3_x$	$Q3_y$				
$R1_x$	$R1_y$	$R2_x$	$R2_y$	$R3_x$	$R3_y$	$R4_x$	$R4_y$	$R5_x$	$R5_y$

Where $P1_x$ and $P1_y$ represent the x and y coordinates of the first cluster centroid for solution P (the same applies for the rest of cluster centroids and solutions). Plots (a) and (b) of Figure 1 show solutions to a particular clustering problem with two and three centroids respectively.

Using the above encoding, the typical implementation of genetic operators for NSGA-II must be modified to deal with chromosomes of different sizes. The genetic operators used in this algorithm will be detailed in Section III-C.

B. Fitness Evaluation

This work focuses on tackling the cluster analysis problem from a multi-objective optimisation approach. Only two objective functions will be used, which allows easy visualisation of Pareto-optimal front solutions and it is a good starting point before extending the algorithm to deal with many objectives. The objective functions used for this research are known in the literature as overall deviation and connectivity [9], [15].

The first objective to be considered consists of a measurement of compactness of a cluster solution. It is computed as the overall summed distances between data items and their corresponding cluster center and represents the overall deviation or dispersion of a cluster partition.

$$Dev(C) = \sum_{c_k \in C} \sum_{i \in c_k} \delta(i, \mu_k) \tag{1}$$

Where C is the set of all clusters, μ_k is the centroid of cluster c_k, and δ is the chosen distance function (i.e. Euclidean distance). This criterion is similar to the widely used intra-cluster distance and should be minimised for compact cluster solutions.

The second objective considered in this work consists of a measurement of cluster connectedness. The connectivity is calculated as the reciprocal of the minimum of centroid to centroid distances between different clusters.

$$Conn(C) = \frac{1}{\min ||c_i - c_k||^2} \quad \forall i, k \in C, i \neq k \tag{2}$$

Where again C is the set of all clusters. This other criterion is analogous to the concept of inter-cluster distance and should be minimise for disperse clusters.

A solution with many clusters will reduce the overall deviation as the data points get closer to the cluster centroids. On the other hand, the same solution will increase the connectivity as the cluster centroids get closer to one another. The adequate number of clusters would be one that tries to reduce both objectives simultaneously, but since the two objectives are conflicting in nature, a preferred solution will be the one that would make a good compromise to both objectives.

Once the overall deviations and connectivities are computed for all individuals of the population, the non-dominance criteria can be applied to make the genetic search evolve

towards the Pareto-optimal frontier. Individuals are then sorted by a level of dominance and for individuals belonging to the same level of dominance, the typical crowding distance of the k^{th} individual within the Pareto-optimal front is computed as

$$Dist^{k}_{Crowding} = \left(\frac{f_1^R - f_1^L}{f_1^{max} - f_1^{min}}\right) + \left(\frac{f_2^R - f_2^L}{f_2^{max} - f_2^{min}}\right) \quad (3)$$

where the superscripts R and L represent the closest solution to the k^{th} individual of the Pareto-optimal front towards the right and left hand side respectively. The subscripts 1 or 2 refer to the objective function being considered. Therefore, the closest solution towards the right will have the values (f_1^R, f_2^R) for each of the objective functions considered. The superscripts max and min refer to the maximum and minimum value of each objective function within all elements in the Pareto-optimal front. Note that individuals at the corners of the Pareto-optimal front should be preserved and therefore they are assigned the maximum value of crowding (large values of crowding mean that the neighbours are far away which is preferred). This crowding distance will be used by the NSGA-II algorithm to distribute solutions within the Pareto-optimal Front.

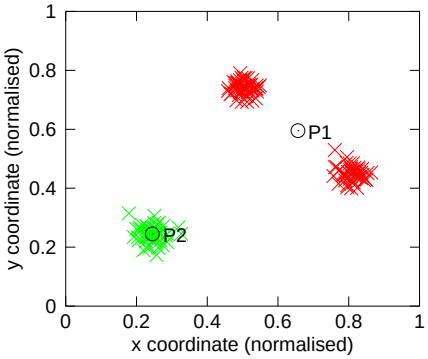
C. Customised Genetic Operators

The evolutionary algorithm developed for this research uses a variable size chromosome encoding and therefore some of the standard genetic operators traditionally used in evolutionary computation need to be modified in order to deal with the variable nature of the chromosomes.

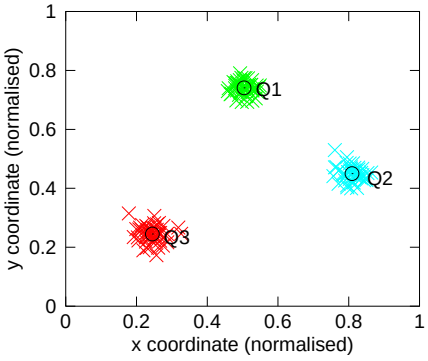
1) Selection: Selection is based on fitness, i.e. overall deviation and connectivity. The selection of individuals is done by tournaments of size two. Individuals with better level of dominance will be preferred. For individuals having the same level of dominance, those having a larger crowding distance will be preferred (i.e. those placed in regions of the corresponding front that are less crowded).

2) Cross-over: The cross-over accounts for all possible combinations of centroids between the two parent individuals. A pool of genes containing information related to all combinations will be created from which the actual genes of the offspring will be selected at random. Considering two different parents with two and three cluster centroids respectively, every possible combination of P and Q, i.e. P_1Q_1, P_1Q_2, P_1Q_3, P_2Q_1, P_2Q_2 and P_2Q_3, will have a contribution to the pool of genes. Plots (a) and (b) of Figure 1 show an example of such parents. Clusters from different individuals may not have common elements, if this is the case, the contribution of that particular cluster combination will be double, both cluster centroids will be in the pool of genes. On the other hand, if there are common instances, the contribution takes the form of a single cluster centroid that lies in the line between the two former centroids. Such a line will also be extended on both ends by an amount that depends on the size of the common set of instances (proportionally to the number of common instances). The particular case where the combination consists of two centroids at the same location will give as a result the same centroid.

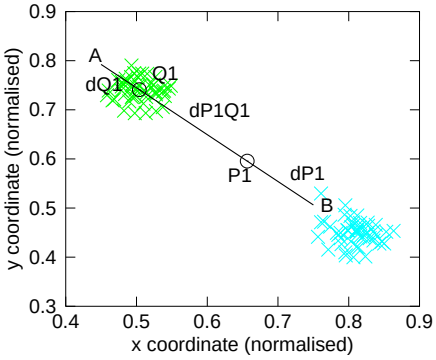
Consider again the example of Figure 1, it is clear that the combinations of clusters P_1Q_1, P_1Q_2 and P_2Q_3 have common



(a) Solution with two clusters.



(b) Solution with three clusters.



(c) Crossover when the contributing clusters contain common instances.

Fig. 1. Example of crossover. Every possible combination of solutions P and Q will contribute to the pool of genes, i.e. P1Q1, P1Q2, P1Q3, P2Q1, P2Q2 and P2Q3. The contribution from combination P1Q1 will be a single cluster centroid which lies in the line between centroids P and Q.

instances whereas the rest do not. Thus, the contribution of the combination P_1Q_1 will be a single cluster centroid randomly chosen from the points that lie in the segment [A, B] of the line that goes from P_1 to Q_1. The amount for the extensions dP_1 and dQ_1 are computed as

$$dP_1 = \frac{|P_1 - P_1 \cap Q_1|}{|P_1 \cap Q_1|} dP_1Q_1 \quad (4)$$

$$dQ_1 = \frac{|Q_1 - P_1 \cap Q_1|}{|P_1 \cap Q_1|} dP_1Q_1 \quad (5)$$

where dP_1Q_1 represents the distance between centroids P_1 and Q_1, $|P_1 - P_1 \cap Q_1|$ represents the number of instances in cluster P_1 that are not in Q_1 whereas $|Q_1 - P_1 \cap Q_1|$ refers to the number of instances in cluster Q_1 that are not in cluster P_1 and $|P_1 \cap Q_1|$ represents the number of instances common to both clusters. Let the contribution to the pool of genes from combinations P_1Q_1, P_1Q_2 and P_2Q_3 be called C_1, C_2 and C_3 respectively. Then, the pool of genes for the simple example of Figure 1 will contain the following genes: C_1, C_2, P_1, Q_3, P_2, Q_1, P_2, Q_2 and C_3 (note that C_3 will be at the same location as P_2 and Q_3 and the pool of genes can be rewritten as: C_1, C_2, P_1, C_3, C_3, Q_1, C_3, Q_2 and C_3). Once the pool of genes has been created, the offspring arises by determining at random its number of genes (within the chromosome size range) and copying them randomly from the pool of genes. This recombination scheme allows chromosome size changes as the offspring size is not bounded to the sizes of its parents.

3) Mutation: The mutation mechanism is fairly simple, every single gene of an individual has a probability of being mutated. Such a mutation takes the form of splitting an existing cluster into two new clusters or distributing cluster instances into neighbouring clusters. The latter can be seen as a form of merging nearby clusters or as an action of removing a particular cluster centroid. Again, these mechanisms of creating new clusters and removing existing ones allow free chromosome size changes for automatic estimation of the number of clusters for a given solution.

4) Reallocation: The centroid location represented in the genes of the offspring must be reallocated towards the center of mass of the cluster they represent. This will be done in the same way as it is done in k-means [13].

5) Replacement: The developed algorithm uses an elitist and stationary replacement policy. At every generation, only one offspring is created which replaces the worst individual in the population, if and only if, that individual is worst than the offspring and there are no copies of it in the population (not allowing copies of individuals helps to preserve a better genetic variability). With regards to fitness evaluation, a good individual will be close to the first level of dominance and will have a large value of crowding. Note that this replacement policy only requires to re-compute the fitness of the entire population when there are no copies of the offspring in the population and therefore it may lead to different computational times for consecutive executions that have been left running for the same number of generations.

D. Operation Sequence

Above ideas are integrated into NSGA-II by following the sequence of instructions described in Algorithm 1. Note that the bi-objective nature of the algorithm comes from the evaluation of the two contradictory objective functions, namely connectivity and overall deviation, which are needed to compute the fitness, i.e. dominance and crowding, of a single solution.

Algorithm 1 Bi-objective Clustering Algorithm.

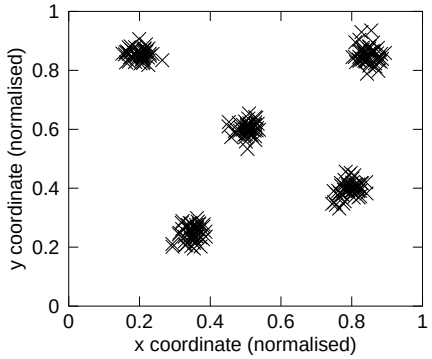
```
Load data set;
Create random initial population;
Evaluate population (dominance and crowding) over data;
while Not Stopping Condition do
    Select first parent (tournament - fitness based);
    Select second parent (tournament - fitness based);
    Apply crossover to obtain offspring;
    Mutate offspring;
    Reallocate offspring;
    Steady state replacement (fitness based);
    if Replacement done then
        Evaluate population (dominance and crowding) over
        data;
    end if
    Store partial data (CPU time, HV, population);
end while
```

IV. Experiments and Results

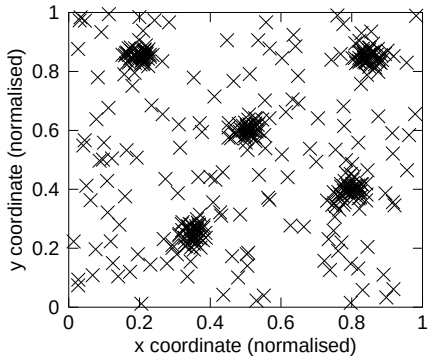
A set of experiments will be carried out to study the performance of the proposed algorithm over numerical data. These data were created for the purpose of this work and consist on six cluster analysis problems where bi-dimensional instances are distributed over 5 and 15 clusters. The instances are distributed in three different ways: they form compact and isolated clusters, they form compact and isolated clusters with added noise (equally distributed) and they form overlapped clusters. The raw data for these clustering test problems are shown in Figures 2 and 3. Note that for the data shown in Figure 3(c) it is not clear, at a simple sight, which would be the preferred partition. When solving the 5 clusters problems a maximum chromosome size of 10 genes was allowed whereas this parameter was set to 20 genes when solving the 15 clusters problems for obvious reasons. All executions of the algorithm were run using a population of 10 individuals. Mutation probabilities for splitting and merging were both set to 5%.

Figure 4(a) shows the Pareto-optimal front solution resulting from the 5 clusters problem described in Figure 2(a). It was observed that the preferred single solution, graphically shown in Figure 2(b), corresponds to the knee point (marked as A). With no surprise, the solutions to the right and left of the knee point, solutions B and C, diverge from the knee solution in that one of the clusters has been split or two the clusters have been merged, substantially reducing either the connectivity or the overall deviation. Plots (c) and (d) of Figure 2 show clustering solutions B and C respectively. The Pareto-optimal front solutions for the noisy and overlapped problems described in Figure 2(b) and (c) are shown in Figure 5. Similarly, the preferred single solutions are found at the knee point (marked as A). Analysing in detail the plots in Figures 4(a), 5(a) and 5(b) yields the following concluding remarks:

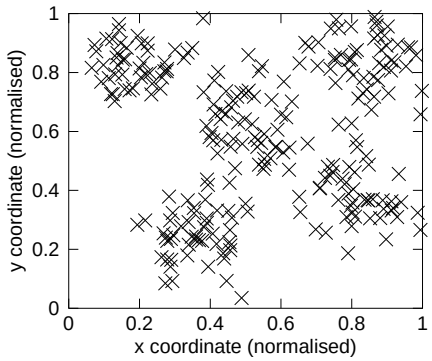
- Knee solutions are interesting and often the desired solution. Similar findings were observed in [15] on a clustering problem.
- When the data to be analysed contains noise or overlapped clusters, the angle measure of the knee point gets substantially reduced (the knee solution is still



(a) Data set with 5 isolated clusters (250 instances).

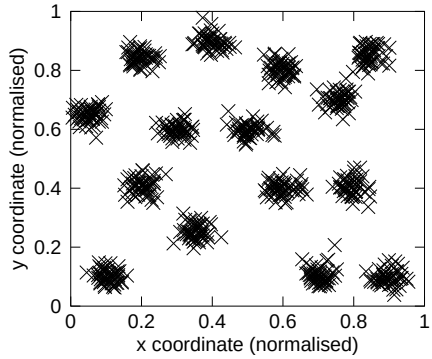


(b) Data set with 5 clusters and noise (400 instances).

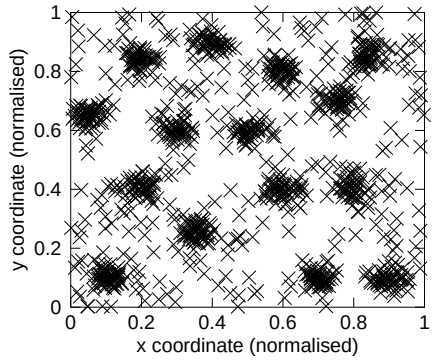


(c) Data set with 5 overlapped clusters (250 instances).

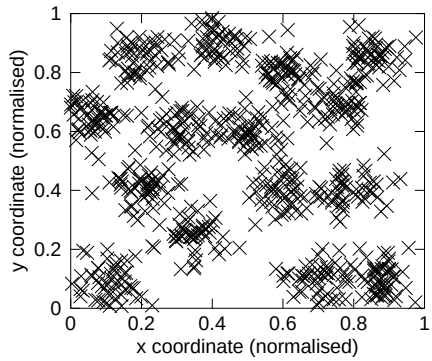
Fig. 2. Data sets with 5 clusters used for validation.



(a) Data set with 15 isolated clusters (750 instances).



(b) Data set with 15 clusters and noise (1050 instances).



(c) Data set with 15 overlapped clusters (750 instances).

Fig. 3. Data sets with 15 clusters used for validation.

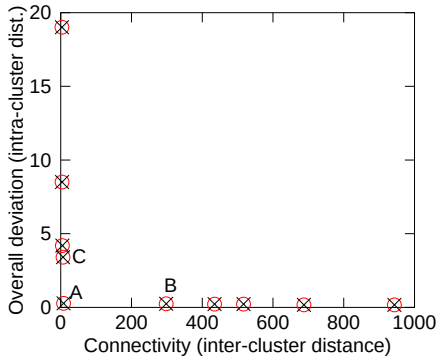
the preferred solution).

It is worth mentioning that the second remark agrees with the fact that if the noise in the data increases to the point where there is not possible to choose a preferred solution, then, there will not be any knee on the Pareto-optimal front. A completely random set of points can be represented by very many clusters or just one big cluster, with no obvious choice.

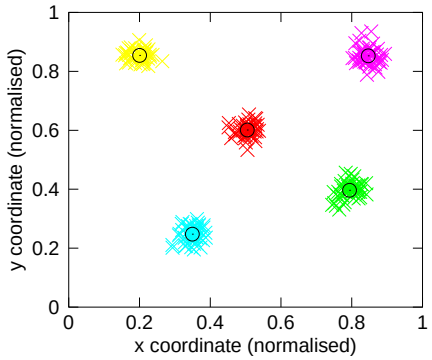
The Pareto-optimal front solutions resulting from the 15 clusters problems graphically described in plots (a), (b) and (c) of Figure 3 are shown in Figure 6. Similar knee regions can be observed. It is interesting to note that for all three cases, the knee point consists on the preferred single solution of the front. Also, the sharpness of the knee region gets substantially decreased when the data contains noise. These observations made on all resulting Pareto-optimal fronts led to the premise that modifying the algorithm such as knee solutions are found directly will improve its performance.

V. Finding Knee Solutions Directly

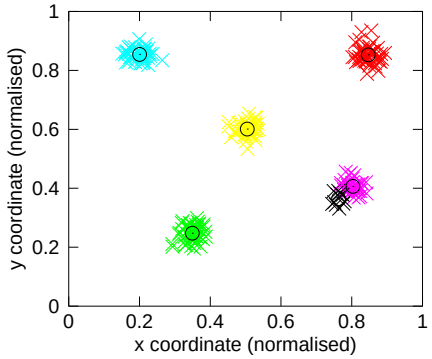
The facts that knee regions are found on the resulting Pareto-optimal fronts from the experiments carried out in Section IV and that these knee regions often consist on the preferred single solution, are interesting and suggest that similar clustering solutions may be found in a faster manner if the algorithm is guided towards finding the knee point



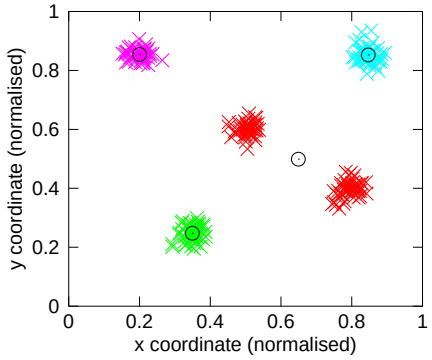
(a) Pareto-optimal front for data set with 5 clusters.



(b) Solution A from the front shown in Figure 4(a).

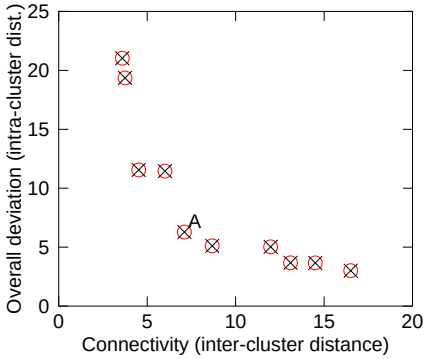


(c) Solution B from the front shown in Figure 4(a).

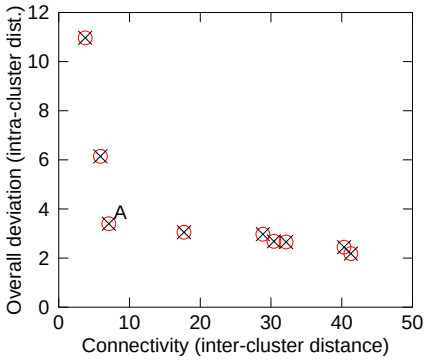


(d) Solution C from the front shown in Figure 4(a).

Fig. 4. Pareto-optimal front and solution for data set with 5 clusters.



(a) 5 clusters and noise.



(b) 5 overlapped clusters.

Fig. 5. Pareto-optimal front for data sets with 5 clusters and noise (a) and with 5 overlapped clusters (b).

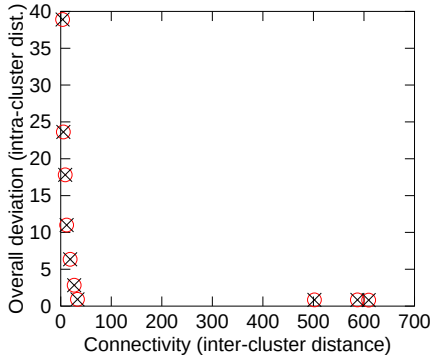
and its neighbouring solutions straightaway. In [16] NSGA-II was modified to find knee points directly yielding promising results. The simplest of such modifications consisted on replacing the crowding distance described in (3) by an angle based measure that identifies the knee points. Similarly, the algorithm presented in III will be modified in such a way that it guides the search towards finding knee regions straightaway and it still keeps using the same encoding and genetic operators as before.

More formally, to calculate the angle measure for the k^{th} individual of the Pareto-optimal front, the following equation applies and replaces the crowding distance

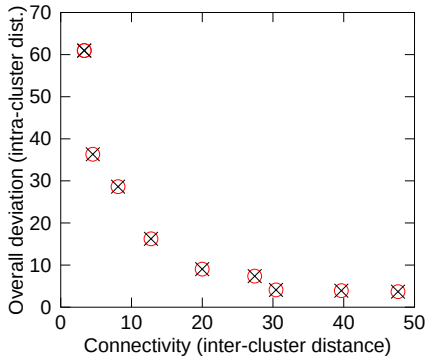
$$Angle^{k}_{Knee} = angle\left((f_1^L, f_2^L), (f_1^R, f_2^R)\right) \qquad (6)$$

where again (f_1^L, f_2^L) and (f_1^R, f_2^R) represent the objective values for the neighbouring solutions towards left and right hand side of the the considered k^{th} solution. Therefore, (6) represents the angle between the individual and its two closest neighbours towards right and left. If the considered k^{th} solution is at the far corners of the front, i.e. no neighbour is found to the left or right, a vertical or horizontal line is used to calculate the angle.

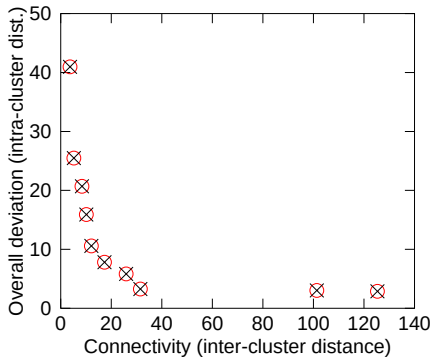
The crowding based algorithm proposed in Section III was modified to account for the knee finding technique described above. The performance of both versions of the algorithm (crowding and knee finding), was compared in terms of generated hypervolumes and computational times. The experimental



(a) 15 clusters.



(b) 15 clusters and noise.



(c) 15 overlapped clusters.

Fig. 6. Pareto-optimal fronts for data sets with 15 clusters.

set up consisted in solving each of the six clustering problems studied here by applying both algorithm versions. In order to reduce the error coming from fluctuations that arise from the stochastic nature of the algorithm, averaged values in 10 executions of each experiment will be presented. Figures 7 and 8 present a comparative on the evolution of the hypervolumes generated by either versions, crowding and knee finding, for each of the studied problems. It can be clearly seen that when the clusters are compact and isolated the knee finding version of the algorithm has a quicker response. When analysing noisy data, the response of the knee finding version is still quicker but not much different from the crowding version. On the other hand, when the studied problems consist on overlapped

TABLE I. COMPARISON OF RESULTS. AVERAGED VALUES IN 10 RUNS.

		Crowding			Knee finding		
Data Set	**Ref-HV**	**G**	**Time**	**HV**	**G**	**Time**	**HV**
5 Clusters	0.9766	58	167.24	0.9853	12	54.63	0.9853
5 C Noise	0.5886	34	439.36	0.6090	31	417.45	0.6090
5 C Overlapping	0.7087	22	254.75	0.7161	33	257.96	0.7181
15 Clusters	0.9385	81	4754.40	0.9482	37	941.81	0.9489
15 C Noise	0.9150	93	5678.28	0.9185	58	4136.34	0.9197
15 C Overlapping	0.8840	55	4522.40	0.8885	66	3705.11	0.8902

clusters, it is no longer possible to claim that knee finding yields a quicker response. However, larger hypervolumes were obtained, i.e. better distributions of Pareto-optimal front solutions, when knee finding techniques were used.

Table I summarises the numerical results of the above experimentation in terms of computational efficiency and maximum hypervolume. In order to fill such table, the reference hypervolume (refHV) was computed as the average value of the hypervolume yielded by both versions of the algorithm for each data set. Then the averaged number of generations (G) and computational times (Time) needed to reach such reference hypervolume were computed. The maximum value of the hypervolume obtained for either version of the algorithm is shown in columns under the tag HV (average value in 10 runs). It can be seen that for isolated clusters and noisy data, a quicker response was obtained using the knee finding version of the algorithm. In general, the knee finding version of the algorithm yields better distribution of Pareto-optimal front solutions in terms of its computed hypervolume.

VI. CONCLUSION

Studying cluster analysis problems from a multi-objective approach was the main motivation behind this work. For that purpose, a bi-objective clustering algorithm has been proposed and its performance over artificial data sets has been evaluated. Two important conclusions have been drawn from the simulation results: (i) For crisp data, the resulting Pareto-optimal front has a clear knee and the most desired clustering solution lies at the knee solution, (ii) For noisy data, the knee region is not clear and multiple solutions to the clustering problem exist. Appropriate modification to the proposed algorithm were applied to re-direct the search towards the knee regions. The improved algorithm was then evaluated over the same artificial data and the results were compared against those of the previous version, yielding shorter computational times in order to reach similar solutions (hypervolume based comparison). Therefore, the final conclusion that can be drawn from this work is that using knee finding methods to solve cluster analysis problems is highly recommendable.

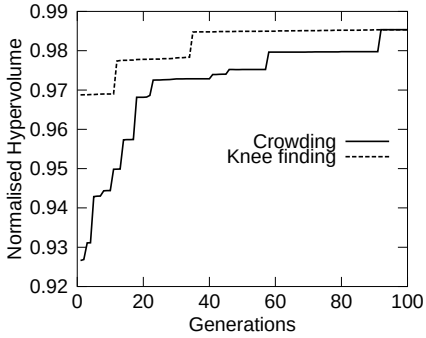
REFERENCES

[1] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: A review," *ACM Computing Surveys*, vol. 31, no. 3, pp. 264–323, 1999.

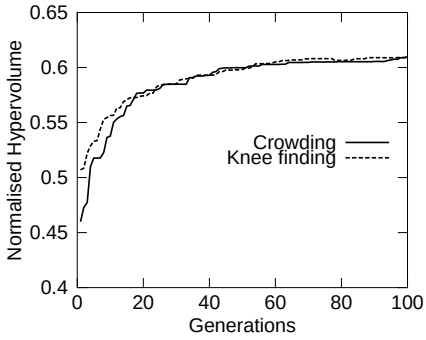
[2] R. Krovi, "Genetic algorithms for clustering: a preliminary investigation," in *System Sciences, 1992. Proceedings of the Twenty-Fifth Hawaii International Conference on*, vol. 4, 1992, pp. 540 –544.

[3] K. Krishna and M. N. Murty, "Genetic K-means algorithm," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 29, no. 3, pp. 433–439, 2002.

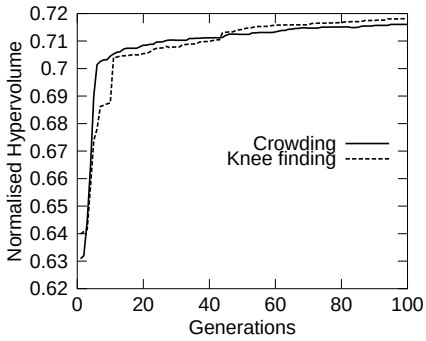
[4] Y. Lu, S. Lu, F. Fotouhi, Y. Deng, and S. J. Brown, "FGKA: a Fast Genetic K-means Clustering Algorithm," in *Proceedings of the 2004 ACM symposium on Applied computing*, 2004, pp. 622–623.



(a) 5 clusters.

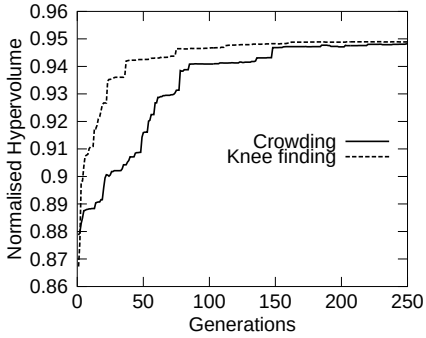


(b) 5 clusters and noise.

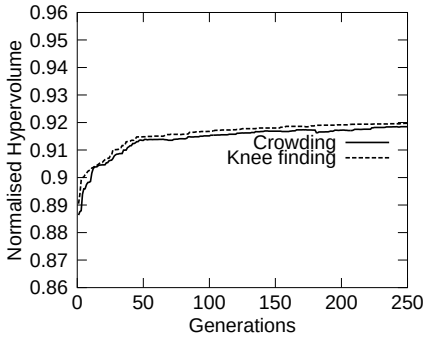


(c) 5 overlapped clusters.

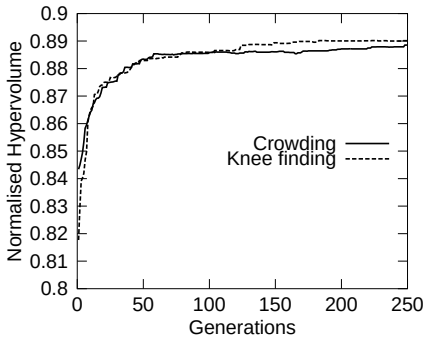
Fig. 7. Hypervolume comparison for the 5 clusters problems.



(a) 15 clusters.



(b) 15 clusters and noise.



(c) 15 overlapped clusters.

Fig. 8. Hypervolume comparison for the 15 clusters problems.

[5] R. H. Sheikh, M. Raghuwanshi, and A. N. Jaiswal, "Genetic algorithm based clustering: A survey," *International Conference on Emerging Trends in Engineering & Technology*, vol. 0, pp. 314–319, 2008.

[6] K. Faceli, A. de Carvalho, and M. de Souto, "Multi-objective clustering ensemble," in *Hybrid Intelligent Systems, 2006. HIS '06. Sixth International Conference on*, 2006, pp. 51 –51.

[7] J. Handl and J. Knowles, "Evolutionary multi-objective clustering," *Lecture notes in computer science*, vol. 3242, pp. 1081–1091, 2004.

[8] ——, "Multi-objective clusterng detection with automatic determination of the number of clusters," Technical report No. TR-COMPSYSBIO-2004-02, UMIST, Department of Chemistry, Tech. Rep., 2004.

[9] ——, "An evolutionary approach to multiobjective clustering," *Evolutionary Computation, IEEE Transactions on*, vol. 11, no. 1, pp. 56 –76, 2007.

[10] K. Ripon, C.-H. Tsang, S. Kwong, and M.-K. Ip, "Multi-objective evolutionary clustering using variable-length real jumping genes genetic algorithm," in *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, vol. 1, 2006, pp. 1200 –1203.

[11] K. Ripon and M. Siddique, "Evolutionary multi-objective clustering for overlapping clusters detection," in *Evolutionary Computation, 2009. CEC '09. IEEE Congress on*, 2009, pp. 976 –982.

[12] K. Faceli, M. de Souto, and A. de Carvalho, "A strategy for the selection of solutions of the pareto front approximation in multi-objective clustering approaches," in *Neural Networks, 2008. SBRN '08. 10th Brazilian Symposium on*, 2008, pp. 27 –32.

[13] J. McQueen, "Some methods for classification and analysis of multivariate observations," *In Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pp. 281–297, 1967.

[14] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *Evolutionary Computa-*

tion, IEEE Transactions on, vol. 6, no. 2, pp. 182 –197, 2002.

[15] K. Deb and S. Gupta, "Understanding knee points and bicriteria problems and their implications as preferred solution principles," Kanpur Genetic Algorithm Laboratory (KanGAL), Indian Institute of Technology, Kanpur, Tech. Rep., 2010.

[16] J. Branke, K. Deb, H. Dierolf, and M. Osswald, "Finding knees in multi-objective optimization," in *Parallel Problem Solving from Nature*, ser. Lecture Notes in Computer Science, 2004, vol. 3242, pp. 722–731.