Comparative results between GenClust++ and Multi-objective GenClust++ algorithms

1. Multi-objective GenClust++

A multi-objective version of GenClust++ (MGenClust++) algorithm has been implemented and compared to the original GenClust++ clustering method showing promising results. MGenClust++ optimizes two objectives, connectivity and cohesion, instead of only one (i.e. Davies-Bouldin Index) utilized in the originally proposed algorithm.

The concept of MaxiMin strategy is applied to determine pareto optimal solutions. This strategy is utilized in chromosome selection operation, which merges all chromosomes between two populations: the last (most recent) population and the resulting generation from all operations. MaxiMin is used here to choose the best chromosomes from the merged populations for the next population. First, non-dominated solutions are chosen and added to the next population. At this point, next population may not be filled up completely, and therefore weakly dominated solutions are picked randomly and added to the next population until it's filled up. The MaxiMin is also used to determine the final best chromosome, used as the initial solution of a final full-length K-Means to deliver the final clustering solution: the solution with the minimum distance to utopia point (best values obtained for the objective functions during optimization process) is chosen is as the final best chromosome.

To enable the genetic operations, a fitness function is calculated using a weighted sum of normalized values of objectives functions. Values of objectives in MGenClust++ are normalized using the best and worst values of objectives found so far, since the value of cohesion is not constrained in the certain interval.

2. Datasets and Performance metrics

To test the algorithms, a set of ten diverse real-world and synthetic datasets from the UCI Machine Learning Repository (UCI) and from the Clustering Repository of the Speech and Image Processing Unit, at the University of Eastern Finland is collected. Glass Identification (Glass) and Breast Cancer Wisconsin (Wdbc) are standard real-world datasets wildly used in literature for clustering analysis. The latter one has a high percentage of outliers. Dim064 and Dim256 are high-dimensional synthetic datasets, having 64 and 256 attributes respectively. S1 and S3 are synthetic datasets used to test algorithms against the datasets with different degree of cluster overlap. Flame, Compound, Pathbased and Jain datasets are non-linearly separable synthetic datasets with different shapes and clusters. The results are compared based on three cluster validity indices, namely Adjusted Rand Index, Davies-Bouldin Index and Silhouette Coefficient, as well as, the number of clusters automatically determined by the algorithms. Thus, the results consist of both detecting the proper number of clusters and the quality of clustering solutions.

Table 1. A brief description of the datasets.

Name of dataset	Total No. of records	No. of attributes	No. of classes
Glass	214	10	6
Wdbc	569	30	2
Flame	240	2	2
Compound	399	2	6
Pathbased	300	2	3
Jain	373	2	2
S1	1000	2	15
S3	1000	2	15
Dim064	1024	64	16
Dim256	1024	256	16

3. Experimental setup

The values of parameters MGenClust++ share with GenClust++ are the same as recommended the original paper, except that a maximum number of generations is set to 20 to decrease the runtime of the algorithm, as the quality of clusters doesn't improve much after 20 generations. The maximum number of generations for GenClust++ is also set to 20. As suggested, the number of short K-Means iterations after cloning is 15, while the final K-Means on the best chromosome has its number of iterations bound to 50. The number of chromosomes in the population is set to 30. For all experiments and all algorithms that use some form of hill-climber of K-Means, the termination condition, that the difference of SSE of two consecutive iterations is less than 0.005, is imposed. The fitness function for GenClust ++ is DB Index, while MGenClust++ is using weighted sum of objectives. In addition, just as in the original paper, K-means with the Manhattan metric as the distance between among attributes and random seed initialization is used.

4. Experimental results

Each technique is run thirty times on each dataset recording their cluster quality for each run. All tables present the average results of the thirty clustering solutions on each dataset for each technique. In addition to tables, Fig.1 is used to present the average results over all datasets for each algorithm in terms of three clustering evaluation techniques.

Multi-objective GenClust++ achieves better mean ARI scores on all the datasets except Wdbc than GenClust++. On the other hand, GenClust++ shows a better performance in terms of DB Index on most of synthetic datasets including high dimensional datasets, and worse performance on both real-world Glass and Wdbc datasets. In terms of achieving optimal number of clusters, MGenClust++ produces better clustering solutions for datasets with lower number of clusters

and datasets with different shapes and clusters (Flame, Compound, Pathbased and Jain), while GenClust++ - for datasets with higher number of clusters, which can be

When considering the average results over all datasets, MGenClust++ achieves better average results on two out three validity indices, namely ARI and Silhouette Coefficient, and nearly same average DB Index.

Thus, MGenClust++ shows promising results, producing good quality clustering solutions and in many cases outperforming original GenClust++ algorithm on the selected benchmark datasets, as well as determining nearly optimal number of clusters.

Table 2. Mean and standard deviation of ARI (higher the better) measured on the outputs of GenClust++ and MGenClust++ (over 30 independent runs).

Dataset	GenClust++	MGenClust++
Glass	0.27 ± 0.0831	**0.5562** ± 0.1753
Wdbc	**0.7616** ± 0.0019	0.707 ± 0.0242
Flame	0.3997 ± 0.0811	**0.4585** ± 0.0403
Compound	0.7164 ± 4.0E-4	**0.7192** ± 0.0137
Pathbased	0.2972 ± 0.0661	**0.4466** ± 0.0293
Jain	0.1425 ± 0.0229	**0.5767** ± 0.0
S1	0.8678 ± 0.1031	**0.9244** ± 0.0292
S3	0.6212 ± 0.0638	**0.6361** ± 0.0161
DIM064	0.9813 ± 0.0361	**0.9849** ± 0.0136
DIM256	0.9979 ± 0.0115	**0.9987** ± 0.0037

Table 3. Mean and standard deviation of DB Index (lower the better) measured on the outputs of GenClust++ and MGenClust++ (over 30 independent runs).

Dataset	GenClust++	MGenClust++
Glass	1.2986 ± 1.0282	**0.5103** ± 0.0028
Wdbc	1.1263 ± 8.0E-4	**1.1241** ± 0.0632
Flame	**0.7202** ± 0.0264	0.8544 ± 0.1249
Compound	**0.5525** ± 0.0014	0.5602 ± 0.0178
Pathbased	0.7461 ± 0.0321	**0.7148** ± 0.0398

Jain	**0.6971** ± 0.0149	0.782 ± 0.0
S1	**0.4517** ± 0.065	0.5957 ± 0.0551
S3	**0.6884** ± 0.0221	0.7955 ± 0.0438
DIM064	**0.1675** ± 0.2123	0.6502 ± 0.4075
DIM256	**0.0427** ± 0.0942	0.1067 ± 0.193

Table 4. Mean and standard deviation of Silhouette coefficient (higher the better) measured on the outputs of GenClust++ and MGenClust++ (over 30 independent runs).

Dataset	GenClust++	MGenClust++
Glass	0.2657 ± 0.1435	**0.575** ± 0.0208
Wdbc	0.3812 ± 3.0E-4	**0.3889** ± 0.0111
Flame	**0.4341** ± 0.027	0.4051 ± 0.0157
Compound	**0.6045** ± 2.0E-4	0.6019 ± 0.0081
Pathbased	0.3795 ± 0.0252	**0.5117** ± 0.0287
Jain	0.4489 ± 0.0146	**0.509** ± 0.0
S1	**0.653** ± 0.0484	0.6273 ± 0.0256
S3	**0.4615** ± 0.0201	0.442 ± 0.0168
DIM064	**0.9475** ± 0.035	0.869 ± 0.0606
DIM256	**0.9806** ± 0.0111	0.9694 ± 0.0323

Table 5. Mean and standard deviation of the average number of clusters (over 30 independent runs) for GenClust++ and MGenClust++

Dataset	GenClust++	MGenClust++
Glass	3.1667 ± 2.4642	**3.3333** ± 0.6992
Wdbc	**2.0** ± 0.0	2.0333 ± 0.1795
Flame	4.8 ± 2.3007	**2.8333** ± 0.4534
Compound	3.0 ± 0.0	**3.0667** ± 0.2494
Pathbased	13.3 ± 2.3402	**3.0667** ± 0.7717

Jain	10.4 ± 1.8903	**2.0** ± 0.0
S1	**13.2333** ± 1.82	20.5667 ± 1.5424
S3	**12.7333** ± 2.0806	23.5667 ± 1.4302
DIM064	**15.7** ± 0.5859	18.9 ± 2.3714
DIM256	**15.9667** ± 0.1795	16.2667 ± 0.6799

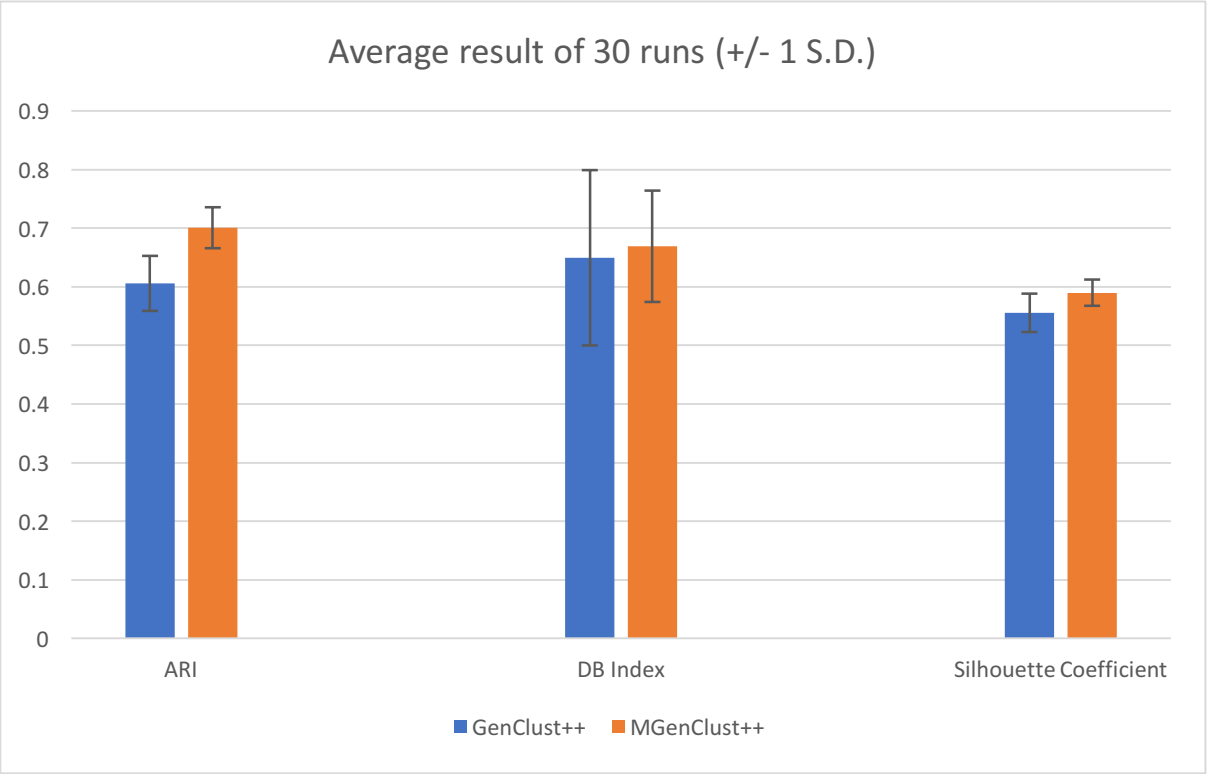

Fig. 1. Comparative average results between GenClust++ and MGenClust++ on 10 datasets based on ARI, DB Index and Silhouette Coefficient

*** Statistical significance tests will be included later as well.**