Evolutionary Multiobjective Clustering

Julia Handl* and Joshua Knowles

Department of Chemistry, UMIST, PO Box 88, Sackville Street,
Manchester M60 1QD, UK
*http://dbk.ch.umist.ac.uk/handl/

Abstract. A new approach to data clustering is proposed, in which *two or more* measures of cluster quality are *simultaneously* optimized using a multiobjective evolutionary algorithm (EA). For this purpose, the PESA-II EA is adapted for the clustering problem by the incorporation of specialized mutation and initialization procedures, described herein. Two conceptually orthogonal measures of cluster quality are selected for optimization, enabling, for the first time, a clustering algorithm to explore and improve different compromise solutions during the clustering process. Our results, on a diverse suite of 15 real and synthetic data sets—where the correct classes are known—demonstrate a clear advantage to the multiobjective approach: solutions in the discovered Pareto set are *objectively* better than those obtained when the same EA is applied to optimize just one measure. Moreover, the multiobjective EA exhibits a far more robust level of performance than both the classic k-means and average-link agglomerative clustering algorithms, outperforming them substantially on aggregate.

1 Introduction

The automation of the human ability to recognize patterns in data, and to induce useful hypotheses from them, is the key goal of *data-mining*. A major branch of this project is the development of methods for unsupervised classification of multidimensional data, namely the *clustering* of data into homogeneous groups: by now a classic AI problem with algorithms dating back to the 60s [15]. In a broad definition, clustering of data might include the recognition and removal of outliers, the recognition and focusing on key dimensions of the data (i.e. feature selection) and the estimation of the correct number of clusters inherent to the data. In a far more restricted definition, the k-clustering problem (on which we focus here) simply requires us to find a partitioning of a set of data into k disjoint sets such that some objective function operating on this partitioning, and employing a notion of distance in the data space, is optimized. This restricted (but still very broad) problem is NP-complete when stated as a question, and remains NP-complete for many restrictions on the distance functions used and the nature of the objective function, even when $k = 2$ [3].

Both classic and a vast array of new algorithms for k-clustering exist [12]. Common to almost all of them is the fact that they optimize either implicitly or explicitly just one measure on the partitioning of the data. For example, k-means [15] attempts to minimize the summed variance of points within each cluster from their centroid. Although such a method is very effective on certain sets of data,

it is also clear that it will fail to find even very obvious cluster structure in other data sets. This is because variance is only a proxy for (i.e. one aspect of) a more fuzzy 'concept' of true cluster structure. Thus, *by focusing on just one aspect of cluster quality, most clustering algorithms can fail catastrophically on certain data-sets*: they are not robust to variations in cluster shape, size, dimensionality and other characteristics.

To combat this problem, practitioners in some fields (where time constraints are secondary) are used to applying several different algorithms to their data, in the hope or expectation that one of them will deliver a good solution. Subsequently, these different partitionings can be tested in the real world: e.g. a biologist with several hypothetical groupings of functionally-related genes can devise experiments that test these alternatives. The idea central to our work is that in such a situation, it may be better to generate alternative solutions using a *single* algorithm, but one that explicitly optimizes *multiple* proxy measures of cluster quality: namely, a Pareto multiobjective EA [7]. This approach may offer greater flexibility and variety in the measures that are used to optimize the clustering, affording higher quality solutions, and, in the process, facilitate greater understanding of the data's structure. In future work we may incorporate feature selection, outlier-removal and determination of k, all within a multiobjective EA framework. However, in this our first paper on multiobjective clustering, we focus on the pivotal question whether this approach can generate objectively high quality solutions.

Readers familiar with clustering research may notice similarities between our proposed approach and other recent methods. Certainly, several EAs for clustering have been proposed ([16, 10, 14, 8]), though none to our knowledge have used a Pareto multiobjective EA. Other recent work has also used the term 'multiobjective' with regard to clustering [13], but the approach was based on using an ensemble of clustering algorithms [18] and then obtaining a consensus clustering from these, similarly to the EA proposed in [10]. Our proposed approach, on the other hand, optimizes different objectives explicitly in one clustering algorithm, enabling different tradeoffs to be explored *during* the clustering process. Its originality derives from this.

The remainder of the paper is organized as follows. Section 2 describes our multiobjective EA, including our selected representation and operators. The objective functions are discussed in Section 3. Section 4 briefly introduces the test suite and points to supporting material where more information on this can be found. Section 5 details our experimental set-up including comparison of our multiobjective EA to two single-objective versions as well as k-means and average-link agglomerative clustering. Section 6 presents results and Section 7 concludes.

2 *VIENNA*: an EA for clustering

A multiobjective evolutionary algorithm (MOEA) for clustering was developed through extensive preliminary experimentation on a diverse set of clustering problems. This algorithm, employing specialized initialization and mutation operators, is called *VIENNA* (for Voronoi Initialized Evolutionary Nearest-Neighbour Algorithm).

2.1 PESA-II

We based *VIENNA* on the elitist MOEA, PESA-II, described in detail in [5] and [6]. Briefly, PESA-II updates, at each generation, a current set of nondominated solutions stored in an external population (of non-fixed but limited size), and uses this to build an internal population of fixed size to undergo reproduction and variation. PESA-II uses a selection policy designed to give equal reproduction opportunities to all regions of the current nondominated front; thus in the clustering application, it should provide a diverse set of solutions trading off different clustering measures. No critical parameters are associated with this 'niched' selection policy, as it uses an adaptive range equalization and normalization of the objectives. PESA-II may be used to optimize any number of objective functions, allowing us to simultaneously optimize several clustering measures, but in this paper we will use just two (conceptually distant) measures as objectives, described in Section 3.

2.2 Representation Issues

PESA-II can be applied without changes to the clustering problem, given a suitable representation of a partitioning, and related operators. A number of GA clustering representations have been tried and compared in the literature, with seemingly no clear overall winner [4]. In the end, we have chosen to use a straightforward representation in which each gene represents a data item, and its allele value represents the label of the cluster to which it is assigned. This means that for any partition, multiple genotypes code for it, i.e. it is a non-injective encoding — normally thought to be undesirable [17]. This drawback is not serious, however, provided there is not a significant bias or over-representation of certain solutions, and/or we can design operators that work effectively and quickly with this coding. Regarding undesirable bias, the inherent frequency of solutions is free from bias: for every solution that correctly partitions the data into k clusters, there are exactly $k!$ genotypes coding for it. Regarding operators, we have discovered an initialization and mutation operator that work well with this coding, as described next.

2.3 Initialization based on random Voronoi cells

In preliminary work not reported here, we investigated an alternative representation for our EA to use, based on optimizing Voronoi cells. This representation was inspired by [16], where an EA was used to optimize the location of k cluster 'centres', to minimize overall variance when the data points were assigned to the nearest centre. This GA achieves results similar to (but slightly better than) the k-means algorithm. Our idea was to extend this representation by allowing the EA to use $j > k$ cluster 'centres' (for a partitioning of k clusters) to enable it to cope better with non-hyperspherical, and especially elongated and intertwined, clusters. In our representation, in addition to the location of the j centres, each centre's label is also evolved on the genotype. The kind of clustering solution that this representation affords is depicted in Figure 1.

Although this representation performs well with PESA-II correctly configured, we have found it slightly inflexible compared with the direct encoding we have

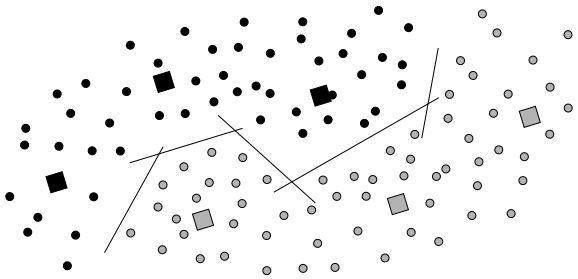

Fig. 1. The kind of complex partitioning boundary enabled by a Voronoi cell genotype coding. Here there are two clusters ($k = 2$) but $j = 6$ centres (squares) have been used to cluster the data. The label of each centre (here visualized by its colour) takes a value in $1..k$. Both the label and location of each centre are coded by the genotype

opted for, as well as adding an unwelcome parameter j, to be chosen. However, we found that the Voronoi coding is very effective at generating diverse and high-quality clustering solutions that can be used to 'seed' our direct-coded EA. The initialization that we have found effective, and which we use in all our experiments, is to set $j = 2k$, and to place the cluster centres uniformly at random in a rectangular polytope centred on the data, and of side-length twice the range of the data, in each objective. The labels associated with each of the j centres is also assigned uniformly at random, from which it is possible to label all of the data items. We then decode this partitioning into our direct coding, and the Voronoi representation is no longer used. This initialization is used for all members of the initial population in *VIENNA*.

2.4 Directed mutation based on nearest neighbours

We have explored numerous recombination and mutation operators in preliminary investigations not reported here, including Grouping GA-based methods [9], as well as multi-parent recombinations based on expectation maximization of an ensemble [18]. Overall, we have found it very difficult to design operators that enable a GA to explore broadly enough to escape the very strong local attractors found in some problems when optimizing certain objectives (e.g. variance on non-spherical clusters), without descending into a fruitless random search of what is a very large search space, and whilst also enabling small clustering differences to be explored.

However, in the end, we have found that a single, deceptively simple, directed mutation operator (and no crossover) is sufficient to drive the search. This mutation operator is applied to every gene with probability p_m, which we set to $1/N$ in all experiments, where N is the size of the data set. When a gene undergoes mutation to a different allele value (i.e. cluster), a number g of other genes are simultaneously 'moved' with it into the same target cluster (and the genotype is updated accordingly). The particular data items that undergo this move are the g nearest neighbours to the data item coded for by the initially mutated gene. The integer g itself is chosen, independently at each mutation event, uniformly at random in $0..N/k$.

This operator enables very large changes to result from a single mutation, yet constrains them to be 'reasonable' moves that respect local distance relations. On the other hand, very small changes in the clustering solution are also possible. The operator works in linear time since the nearest neighbours of every data item can be pre-computed once at the beginning of the EA's run.

3 Objective Functions for Clustering

Given a candidate partitioning of the data, numerous 'internal' measures for estimating its quality exist [11]. These measures are based on intuitive notions of the properties of a desirable partitioning — such as the compactness of clusters and their clear separation.

In the EA we present in this paper, we optimize two such internal measures, described next, though we have tried several others in our preliminary testing.

3.1 Overall deviation

The overall deviation of a clustering solution reflects the overall intra-cluster 'spread' of the data. It is computed as

$$Dev(C) = \sum_{C_k \in C} \sum_{i \in C_k} \delta(i, \mu_k),$$

where C is the set of all clusters, μ_k is the centroid of cluster C_k and $\delta(.,.)$ is the chosen distance function (see Section 4). As an objective, overall deviation should be minimized. Note the relationship to variance (e.g. as used in k-means), which squares the $\delta(.,.)$ in the sum. In preliminary experiments, we found overall deviation to be preferable to variance for use in our EA.

3.2 Connectivity

As a second objective function, we propose a new measure, connectivity, which evaluates the degree to which neighbouring datapoints have been placed in the same cluster. It is computed as

$$Conn(C) = \frac{1}{N} \sum_{i=1}^{N} \left(\frac{\sum_{j=1}^{h} x_{i,nn_i(j)}}{h} \right), \qquad \text{where } x_{r,s} = \begin{cases} 1 & \text{if } \exists C_k : r, s \in C_k \\ 0 & \text{otherwise,} \end{cases}$$

$nn_i(j)$ is the jth nearest neighbour of datum i, and h is a parameter determining the number of neighbours that contribute to the connectivity measure. The value of connectivity lies in the interval [0,1], and as an objective, it should be maximized. Connectivity, unlike overall deviation, is relatively indifferent to the shape of clusters, and we have found it robust to the chosen value of h, independently of the data set. It is also fast to compute as the nearest neighbour list can be pre-computed. One drawback of this measure is that trivial attractors, with all, or nearly all, data items placed in the same cluster, exist.

4 Data Sets

Clustering problems vary greatly along a number of important dimensions. For this reason, it is incumbent on the researcher developing new clustering algorithms to test them on a range of problems that exhibit this variety as much as possible. We use eight synthetic and seven real data sets; the former allow us to control several characteristics in isolation, while the latter help to verify that our results are ultimately meaningful in real applications.

For the real data, we first normalize each dimension to have a mean of zero and a standard deviation of one, and use the Cosine similarity as distance function. For the synthetic data, the Euclidean distance is used with no prior normalization.

4.1 Synthetic Data

All eight of our synthetic data sets consist of 500 two-dimensional data items, enabling us to easily visualize the results of a clustering. Pictures and explanations for all these sets are available at [1] but we briefly describe them below. Note: the synthetic data sets are defined in terms of distributions, and the actual data points are sampled from these, independently, in each successive algorithm run.

Three of the data sets (Square1, Square3 and Square5) consist of a square arrangement of four clusters of equal size and spread, each cluster being a Gaussian distribution about a central point. The difference between the sets is the degree of overlap of the four clusters. In Square1, the clusters touch but hardly overlap, whereas for Square5 the overlap is so much that there is little density difference moving from one cluster to the next.

The next three data sets (Sizes1, Sizes3 and Sizes5) are based on Square1, but change the relative cluster sizes (in terms of the number of constituent data items) such that the ratio of the three smaller to the one larger cluster is respectively 2, 6, and 10. Note: the spread of the clusters is unchanged.

The last two of our synthetic data sets (Smile and Long1) contain different, non-spherically shaped clusters, making it more difficult for methods based on minimizing variance. For pictures of these demanding problems, see [1].

4.2 Real Data

For the real data sets we chose seven from the UCI Machine Learning Repository [2] to obtain a good variety in data dimensionality, size of the data set, number of clusters and evenness/unevenness of the cluster sizes. The sets we chose are Dermatology, Digits, Iris, Wine, Wisconsin, Zoo and Yeast. They range up to size 3498 and dimension 34, with up to 10 clusters. For complete details also see [1]. Note: we permute the data in these sets on each algorithm run.

5 Experimental Setup

In our experiments, we compare *VIENNA* running with two objectives — overall deviation and connectivity — against two classical clustering algorithms with proven performance across a wide range of data sets: k-means and average-link agglomerative clustering. Moreover, to establish the usefulness of a multiobjective approach we also compare against two single-objective versions of *VIENNA*, using identical operators and parameter settings, but optimizing only one objective.

5.1 *VIENNA* Configuration

The parameter settings for *VIENNA*, held constant over all problems, are given in Table 1. There are three versions of the algorithm: a multiobjective one, which we label, *VIENNA*-moo; and two single-objective versions, named *VIENNA*-dev and *VIENNA*-conn. In addition to the objective(s) to optimize, all *VIENNA* algorithms use a constraint that no cluster should be empty, and enforce this constraint by lethally penalizing the value(s) of each objective.

Table 1. Parameter settings for all *VIENNA* algorithms

Parameter	*setting*
Number of generations (synthetic data)	500
Number of generations (real data)	$40\lceil\sqrt{N}\rceil$
External population size	100 (*VIENNA*-moo only)
Internal population size	200
Initialization	random Voronoi cells (see section 2.3)
Mutation type	g nearest neighbours (see section 2.4)
Mutation rate p_m	$1/N$
Recombination	none
VIENNA-moo objective functions	deviation and connectivity ($h = 10$)
VIENNA-dev objective function	overall deviation only
VIENNA-conn objective function	connectivity ($h = 10$) only
Constraints	empty clusters lethally penalized

5.2 Average-link agglomerative clustering and k-means

We use a standard implementation of average-link agglomerative clustering, [20], which is deterministic for a given data order. The implementation of the k-means [15] algorithm is based on the batch version, that is, cluster centres are only recomputed after the reassignment of all data items. As k-means can sometimes generate empty clusters, empty clusters are identified in each iteration and are randomly reinitialized. Obviously, this enforcement of the correct number of clusters can prevent convergence, and we therefore set the maximum number of iterations to 100. To avoid suboptimal solutions k-means is run repeatedly (100 times per 'run') using random initialization, and only the best result in terms of minimum variance is returned.

5.3 Data collection and processing

For each problem, we perform 50 independent runs of the *VIENNA* algorithms and collect data on the entire evolution of the population. In particular, for *VIENNA*-moo we collect the final external population of nondominated points. For k-means and average-link agglomerative clustering we also perform 50 independent runs and collect the final output solution obtained in each run.

We evaluate solutions objectively using the F-measure [19], which combines information on the purity and the completeness of the generated clusters with

respect to the known, real class memberships. This measure is limited to the range $[0, 1]$, where 1 reflects a perfect agreement with the correct partitioning.

Importantly, the F measure we quote for *VIENNA*-moo will be for the solution in the final external population with the best F measure value. This is in-line with our philosophy that in many applications we would be able to test a small number of alternative clustering solutions.

6 Results

The results of the F-measure are presented graphically as boxplots [21] in Figure 2. On several datasets the solutions generated by *VIENNA*-moo are better than those of the other algorithms by a large margin. On Iris, Yeast, Zoo, Long1, and Smile, its superiority is clear. Equally impressive, however, is the fact that it exhibits a far more robust performance than any other algorithm: indeed, the sample median F measure of the *VIENNA*-moo solutions is unbeaten across *all* data sets (not significance tested), and is even slightly better than that of the multiple-restarted k-means algorithm on its 'home territory' of the Sizes and Square series of data-sets, with all-spherical clusters. Tabulated results are available at [1].

Further graphical results are also available at [1], including figures displaying the Pareto fronts obtained on some problems, and plots of the time-evolution of the F measure and the two objectives. A number of these results indicate that the global optima on overall deviation and/or connectivity alone do not correspond with the global optimum on the F measure. It is only by exploring other nondominated local optima (trading off these two objectives) that the EA is able to find good F measure solutions. This exploration is possible as a direct consequence of optimizing multiple objectives.

7 Conclusion

Most clustering algorithms operate by optimizing (either implicitly or explicitly) a single measure of cluster solution quality. Such methods may perform well on certain data-sets but lack robustness with respect to variations in cluster shape, proximity, evenness and so forth. In this paper, we have proposed an alternative approach: to optimize simultaneously over a number of objectives using a multiobjective EA. We demonstrated that with this approach a greater robustness may be achieved — solutions selected from the generated nondominated front were never worse than those generated by either of two classic algorithms, across *all* 15 of our data sets, and were substantially better on a number of them, including three of seven real data sets from the UCI Machine Learning Repository. Much further work is needed to investigate using different and more objectives, and to test the approach still more extensively. However, we will first concentrate on the important issue of developing methods for identifying the best candidate solution(s) from the Pareto front, or reducing the number of solutions that must be assayed. We have already begun with this, and have found it possible to cluster the nondominated front to just a handful of significantly different solutions on the data sets used here.

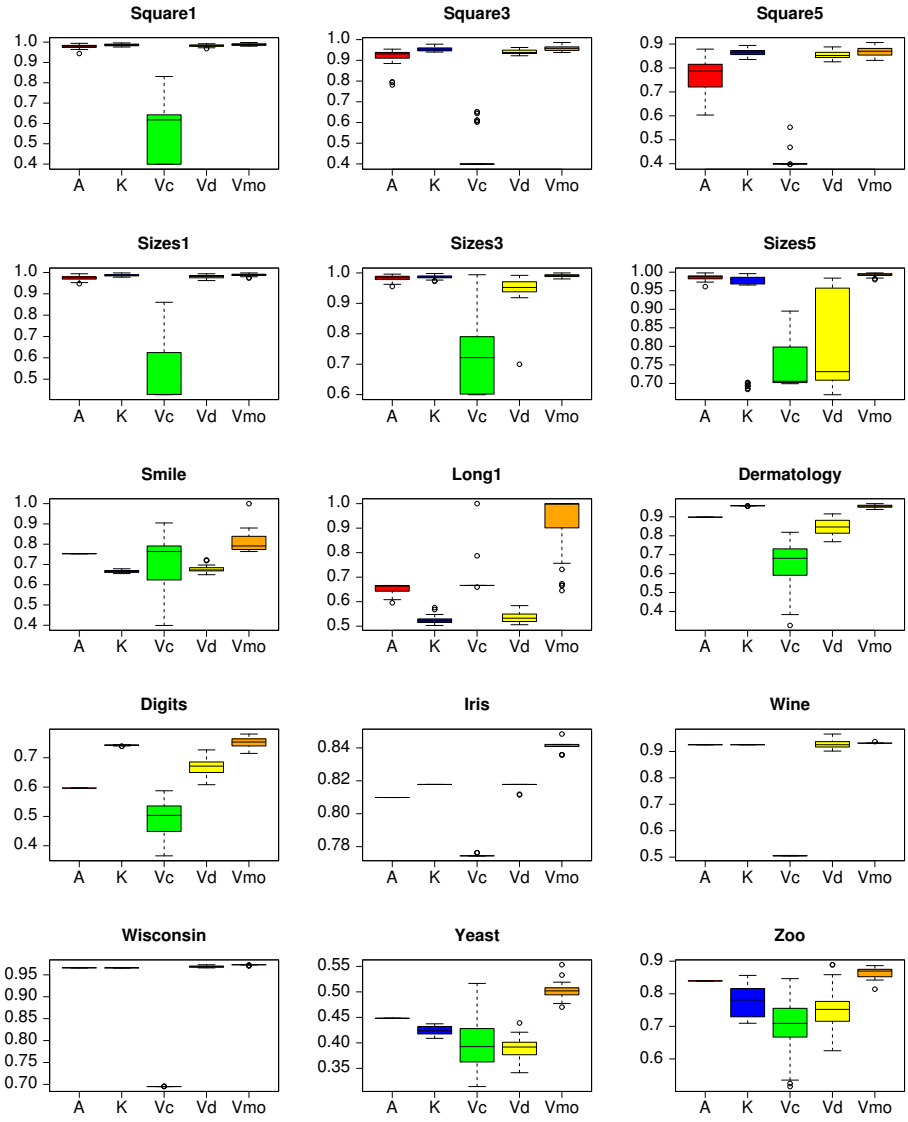

Fig. 2. Boxplots [21] giving the distribution of F measure values achieved for 50 runs of each algorithm on the 15 data sets. Key: A=average-link agglomerative clustering, K = k-means, Vc= *VIENNA*-conn, Vd= *VIENNA*-dev, Vmo= *VIENNA*-moo. Median and IQR values have also been tabulated and can be found at [1]

Acknowledgments VIENNA is built from David Corne's original PESA-II code. JH gratefully acknowledges support of a scholarship from the Gottlieb-Daimler- and Karl Benz-Foundation, Germany. JK is supported by a David Phillips Fellowship from the Biotechnology and Biological Sciences Research Council (BBSRC), UK.

References

1. Supporting material. http://dbk.ch.umist.ac.uk/handl/vienna/
2. C. Blake and C. Merz. UCI repository of machine learning databases. Technical report, Department of Information and Computer Sciences, University of California, Irvine, 1998. http::/www.ics.uci.edu/~mlearn/MLRepository.html
3. P. Brucker. *Optimization and Operations Research*, chapter On the complexity of clustering problems, pages 45–54. Springer-Verlag, New York, 1977.
4. R. M. Cole. Clustering with genetic algorithms. Master's thesis, University of Western Australia, Nedlands 6907, Australia, 1998.
5. D. W. Corne, N. R. Jerram, J. D. Knowles, and M. J. Oates. PESA-II: region-based selection in evolutionary multiobjective optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2001)*, pages 283–290. Morgan Kaufmann Publishers, 2001.
6. D. W. Corne, J. D. Knowles, and M. J. Oates. The Pareto envelope-based selection algorithm for multiobjective optimization. In *Proceedings of the Parallel Problem Solving from Nature VI Conference*, pages 839–848. Springer, 2000.
7. K. Deb. *Multi-objective optimization using evolutionary algorithms.* John Wiley & Sons, Chichester, UK, 2001.
8. A. Demiriz, K. Bennett, and M. Embrechts. Semi-supervised clustering using genetic algorithms. Technical report, Rensselaer Polytechnic Institute, Troy, New York, 1999.
9. E. Falkenauer. *Genetic Algorithms and Grouping Problems.* John Wiley & Sons, 1998.
10. W. Gablentz, M. Köppen, and E. Dimitriadou. Robust clustering by evolutionary computation. 5th Online World Conference on Soft Computing in Industrial Applications (WSC5), The Internet (2000), 2000.
11. M. Halkidi, M. Vazirgiannis, and I. Batistakis. Quality scheme assessment in the clustering process. In *Proceedings of the Fourth European Conference on Principles of Data Mining and Knowledge Discovery*, volume 1910 of *LNCS*, pages 265–267. Springer-Verlag, Heidelberg, Germany, 2000.
12. A. K. Jain, M. N. Murty, and P. Flynn. Data clustering: a review. *ACM Computing Surveys*, 31(3):264–323, 1999.
13. M. Law, A. Topchy, and A. K. Jain. Multiobjective data clustering. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, June 2004. To appear.
14. J. A. Lozano and P. Larrañaga. Applying genetic algorithms to search for the best hierarchical clustering of a dataset. *Pattern Recognition Letters*, 20(911–918), 1999.
15. L. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297. University of California Press, Berkeley, 1967.
16. U. Maulik and S. Bandyopadhyay. Genetic algorithm-based clustering technique. *Pattern Recognition*, 33:1455–1465, 2000.
17. N. J. Radcliffe. Equivalence class analysis of genetic algorithms. *Complex Systems*, 5:183–205, 1991.
18. A. Topchy, A. K. Jain, and W. Punch. A mixture model for clustering ensembles. In *Proceedings SIAM Conf. on Data Mining*, 2004. In press.
19. C. van Rijsbergen. *Information Retrieval, 2nd edition.* Butterworths, London, UK, 1979.
20. E. Vorhees. *The effectiveness and efficiency of agglomerative hierarchical clustering in document retrieval.* PhD thesis, Department of Computer Science, Cornell University, 1985.
21. E. W. Weisstein. Box-and-whisker plot. From MathWorld — A Wolfram Web Resource. http://mathworld.wolfram.com/Box-and-WhiskerPlot.html