# An Investigation of Representations and Operators for Evolutionary Data Clustering with a Variable Number of Clusters

Julia Handl and Joshua Knowles

Manchester Interdisciplinary Biocentre, University of Manchester, UK
j.handl@postgrad.manchester.ac.uk, j.knowles@manchester.ac.uk

**Abstract.** This paper analyses the properties of four alternative representation/operator combinations suitable for data clustering algorithms that keep the number of clusters variable. These representations are investigated in the context of their performance when used in a multiobjective evolutionary clustering algorithm (MOCK), which we have described previously. To shed light on the resulting performance differences observed, we consider the relative size of the search space and heuristic bias inherent to each representation, as well as its locality and heritability under the associated variation operators. We find that the representation that performs worst when a random initialization is employed, is nevertheless the best overall performer given the heuristic initialization normally used in MOCK. This suggests there are strong interaction effects between initialization, representation and operators in this problem.

## 1 Introduction

Data clustering [7] is an unsupervised classification problem in which a set of data items is partitioned into a number of disjoint subsets or 'clusters', based on proximity information. The number of clusters, $k$, inherent to the data is usually unknown *a priori*, so clustering algorithms that investigate solutions with different numbers of clusters may be preferred to algorithms requiring a fixed value of $k$ to be specified by the user. In this work, we consider alternative combinations of representations and variation operators that are suitable for exploring solutions with a variable number of clusters. We analyse the different choices within a multiobjective clustering evolutionary algorithm, MOCK, described previously [6], and attempt to understand the performance differences observed in terms of key properties of the representations and operators.

The effects of representations and operators on evolutionary search are a perennial topic in the literature, though few firm rules or conclusions of practical significance exist. Important factors, nonetheless, are thought to be: the size of the search space induced by a representation; whether phenotype space is entirely covered and/or reachable; whether the mapping from genotype to phenotype is injective, or 'degenerate' [11][1]; whether particular (groups of) phenotypes are over-represented [11,12,13]; and the 'heritability' and 'locality' of

[1] Meaning that several genotypes may map to the same phenotype.

T.P. Runarsson et al. (Eds.): PPSN IX, LNCS 4193, pp. 839–849, 2006.

the representation under crossover and mutation, respectively [12]. We consider each of these aspects in the analysis of the alternative representations studied here, and reflect on how much they tell us about performance.

The organization of the paper is as follows. Section 2 briefly recalls the principles behind the multiobjective clustering algorithm, MOCK, and the requirements it imposes on possible representations and operators. We also briefly revisit other work that considers representations for evolutionary data clustering. Section 3 provides the details of the representations and operators that are studied and the parameter setting used. Section 4 presents both theoretical and empirical findings of the study, and Section 5 concludes.

## 2 Previous Work

**Basic principles of MOCK.** In our previous work, we have described a multiobjective evolutionary algorithm (MOEA) for clustering, called MOCK (for Multiobjective clustering with automatic $k$-determination, [6]). MOCK is based on the elitist multiobjective evolutionary algorithm, PESA-II, described in detail in [2]. It minimizes two clustering objectives, *overall deviation* and *connectivity*: overall deviation is computed as the overall sum of the distances between each data item and its corresponding cluster centre,

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

where $C$ is the set of all clusters, $\mu_k$ is the centroid of cluster $C_k$ and $\delta(.,.)$ is the chosen distance function (here, the Euclidean distance); quite differently, connectivity evaluates the degree to which neighbouring data-points have been placed in the same cluster and is computed as,

$$Conn(C) = \sum_{i=1}^{N} \left( \sum_{j=1}^{L} x_{i,nn_{ij}} \right), \quad \text{where } x_{r,s} = \begin{cases} \frac{1}{j} \text{ if } \nexists C_k : r \in C_k \wedge s \in C_k \\ 0 \quad \text{otherwise,} \end{cases}$$

and where $nn_{ij}$ is the $j$th nearest neighbour of datum $i$, $N$ is the size of the clustered data set, and $L$ is a parameter determining the number of neighbours that contribute to the connectivity measure.

When these two objectives are minimized, an approximate Pareto front is obtained with solutions arranged along the front by the number of clusters, $k$. (This arrangement occurs because the objectives have opposite biases with respect to the number of clusters [6]). The shape of the Pareto front gives important clues as to which is the actual 'best solution' and MOCK uses an automatic heuristic strategy to select the best solution, hence determining or estimating $k$ automatically. Using this strategy, MOCK seems to provide solutions of a more consistently high quality compared to some other clustering algorithms, when a range of data sets is considered [6]. However, the final solution returned by MOCK's selection strategy depends crucially on obtaining a good overall Pareto front, and one that

covers a range of values of $k$. This means that any representation used must facilitate finding solutions over a wide range of different numbers of clusters. Moreover, the representation must encourage effective exploration of the search space for another reason: although overall deviation is a relatively easy objective to minimize, connectivity (like most objectives capturing spatial separation or local connectedness [6]) provides relatively poor guidance in some areas of the search space.

**MOCK's representation, operators and initialization.** The representation employed in MOCK in [6] is the locus-based adjacency scheme proposed in [10]. In this graph-based representation, each individual $g$ consists of $N$ genes $g_1, \ldots, g_N$, where $N$ is the size of the clustered data set, and each gene $g_i$ can take allele values $j$ in the range $\{1, \ldots, N\}$. A value of $j$ assigned to the $i$th gene, is then interpreted as a link between data items $i$ and $j$: in the resulting clustering solution they will be in the same cluster. The decoding of this representation requires the identification of all connected components, and all items belonging to the same connected component are assigned to one cluster. This decoding step can be done in linear time.

MOCK's initialization is based on (i) minimum spanning trees (MSTs) and (ii) the $k$-means algorithm [9]. For a given data set, the complete MST is computed using Prim's algorithm. Individuals corresponding to different clustering solutions are then obtained by breaking up the MST, using either a measure of 'interestingness' of individual links or the partitionings prescribed by the $k$-means solutions. This has the effect of generating solutions with a range of cluster numbers that already provide a good and well-spread approximation to the Pareto front [6].

MOCK's variation operators are uniform crossover and a mutation operator that allows data items to be linked to one of their $L$ nearest neighbours only. Hence, $\forall i, g_i \in \{nn_{i1}, \ldots, nn_{iL}\}$, where $nn_{il}$ denotes the $l$th nearest neighbour of data item $i$.

**Alternative representations.** For single-objective clustering tasks, a variety of different EA representations for clustering solutions have been explored in the literature (see [1]), ranging from a straightforward representation (with the $i$th gene coding for the cluster membership of the $i$th data item), to more complex representations, such as matrix-based or permutation-based representations. Falkenauer's grouping GA [3] also provides a general template for the implementation of evolutionary algorithms for grouping problems, although an application of the approach to straightforward data clustering has not been demonstrated, to our knowledge, previously, and under Falkenauer's template, this would require the design of several clustering-specific operators. Much previous work has also explored the use of existing clustering heuristics (most notably the $k$-means algorithm) as the cluster generator in a hybrid coding scheme (see [8]). This restricts the search space 'seen' by the evolutionary algorithm to the set of local optima that can be identified by the clustering heuristic used, and is therefore unsuitable for the use in multiobjective clustering where trade-off solutions (not identifiable by existing single-objective clustering heuristics) are to be found.

## 3 Representations/Operators Studied

The four different combinations of representations/operators investigated in this paper are as follows.

**Baseline:** A straightforward clustering representation, with one gene for every data item and its allele value specifying the cluster membership. An upper limit on the maximum number of clusters is imposed. Uniform crossover and a standard mutation operator (random change of cluster membership for a single data item) are used.

**VIENNA:** Representation identical to the Baseline representation. No crossover, but the mutation operator introduced in [5] is used. 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 $0, \ldots, N/2$.

**Falkenauer:** Following the principles in [3], the genome of every individual consists of two sections, the first being identical to the above Baseline representation, and the second being a list of the groups (i.e. the set of allele values making up the first part). Two-point crossover operates directly on the second section, and the first section is updated, subsequently. We have developed a crossover operator that uses heuristic information to repair the first section: all unassigned data items are assigned the cluster membership of their nearest data items. We have also designed three mutation operators for the splitting, merging and the exchange of data items between groups, respectively. The splitting mutation operates on a randomly selected group: two items from this 'parent' cluster are randomly selected and used as 'seeds' for the two 'daughter' clusters. All remaining data items are then assigned to the cluster whose seed they are closer to. The merging mutation simply joins two randomly selected clusters. The swapping mutation swaps the cluster membership of two randomly selected data items.

**MOCK:** MOCK's standard representation and operators (see Section 2).

These different representations and operators are 'plugged' into our existing multiobjective clustering algorithm, that is, the optimization algorithm (PESA-II) and the objectives used remain constant throughout the experiments. MOCK's heuristic initialization scheme is also used for all representations/operators, but we additionally conduct experiments with random initialization in order to assess the impact of this initialization scheme. Here, the random initialization schemes used for the different representations vary slightly, as we aim to use the initialization that intuitively seems the most 'natural' for each of the representations used. For the adjacency-based representation, a random initialization is obtained by linking each data item to one of its randomly selected $L$ nearest neighbours

**Table 1.** Parameter settings for the four algorithms. In those representations using crossover, a crossover probability of 0.7 is used. The mutations in the Baseline representation and VIENNA, and Falkenauer's swapping mutation are applied with a probability of $\frac{1}{N}$, where $N$ is data set size. For MOCK, the biased mutation probability $p_m = \frac{1}{N} + (\frac{l}{N})^2$ introduced in [6] is used. The merge and split mutations in Falkenauer's representation are applied to a given individual with a probability of 0.2 each.

| *Parameter* | *setting* |
|---|---|
| Number of generations | 1000 |
| External population size | 1000 |
| Internal population size | 10 |
| Resolution of hypergrid per dimension | 10 |
| #(Initial solutions) | 100 |
| Objective functions | Overall deviation and connectivity ($L = 10$) |

of the data set. For the other three representations we first (randomly) determine the number of clusters, and then (randomly) assign cluster numbers to the individual data items.

Apart from representation-specific variations, the parameter settings are kept constant throughout the experiments and are summarized in Table 1.

## 4 Analysis

### 4.1 Empirical Performance Analysis

The data sets used in our empirical performance analysis have been previously described in [6] and are obtained using a generator for Gaussian clusters. For eight different combinations of cluster number and dimension, 10 different instances are generated, giving 80 data sets in all. In our experiments, these groups of 10 instances are referred to as $x$d-$y$c, where $x$ is the dimensionality and $y$ is the number of clusters in the data set.

In order to assess the algorithms' performance at solving the clustering task, two different measures are employed. Firstly, the quality of the best solution present in the Pareto front is analyzed. Here, the quality of a clustering solution is established using an external validation technique (which compares to the known correct partitioning), the Adjusted Rand Index (see [6]). It returns values in the interval $[\sim 0, 1]$ and is to be maximized.

Secondly, the quality of the Pareto fronts obtained is assessed using the hypervolume indicator (a.k.a. the S measure [15]), a standard measure from the literature. This indicator assesses the size of the region dominated by a sample set of points, and is to be maximized. Here, the Pareto front of all runs combined is normalized to lie in (0,1),(1,0); this normalization is then applied to each Pareto front. To compute each hypervolume, the point (2,2) is used to bound the dominated region, and the hypervolumes are divided by 4.0 to normalize them to a maximum value of 1.0.

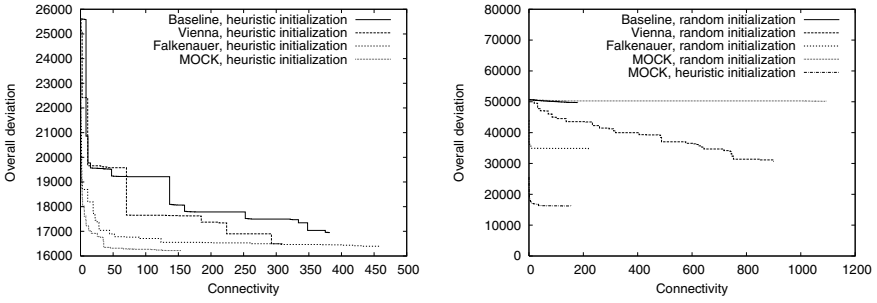


**Fig. 1.** Illustration of the differences between the approximation sets returned by the four algorithms on one of the 10d-40c data sets. (Left) With heuristic initialization. (Right) With random initialization. Results are for the first run of each algorithm.

**Table 2.** Mean values of the Adjusted Rand Index (wrt the known correct partitioning) of the best solution in the approximation sets identified by the four MOEAs, and the hypervolume of the entire approximation sets. Bold font indicates the statistically best performer (and ties) under a paired Wilcoxon test (overall $\alpha = 0.05$).

| | Adjusted Rand Index | | | | Hypervolume Indicator | | | |
|---|---|---|---|---|---|---|---|---|
| Data set | MOCK | Falkenauer | VIENNA | Baseline | MOCK | Falkenauer | VIENNA | Baseline |
| 2d-4c | **0.988905** | 0.98401 | 0.965744 | 0.853658 | **0.98123** | 0.972678 | 0.963554 | 0.953729 |
| 2d-10c | **0.948349** | 0.91774 | 0.858553 | 0.800329 | **0.985935** | 0.97792 | 0.959744 | 0.949203 |
| 2d-20c | **0.949477** | 0.935117 | 0.877584 | 0.862006 | **0.989509** | 0.982987 | 0.968587 | 0.96579 |
| 2d-40c | **0.875799** | 0.81649 | 0.775303 | 0.771587 | **0.987412** | 0.976384 | 0.954132 | 0.949916 |
| 10d-4c | **0.995852** | **0.996396** | 0.96457 | 0.898749 | **0.968248** | 0.95513 | 0.941786 | 0.933688 |
| 10d-10c | **0.969794** | 0.955316 | 0.893811 | 0.859238 | **0.978902** | 0.970791 | 0.949533 | 0.94131 |
| 10d-20c | **0.997959** | **0.997976** | 0.970049 | 0.961713 | **0.983116** | 0.978429 | 0.96757 | 0.966485 |
| 10d-40c | **0.99129** | 0.983576 | 0.943561 | 0.937465 | **0.997303** | 0.988787 | 0.971748 | 0.968548 |

Table 2 summarizes the performance of the four algorithms under these two different measures. The results indicate clear performance differences between the algorithms. MOCK emerges as the strongest overall performer. In terms of the Adjusted Rand Index, it is closely followed by Falkenauer's representation, however, the results of the hypervolume reveal a significantly better convergence of MOCK towards the Pareto front. Both MOCK and Falkenauer's representation outperform VIENNA and the Baseline method. Figure 1 provides some representative examples of the approximation sets obtained by the four algorithms on a complex data set, and contrasts these results with the performance of the algorithms when random initialization is used. With the latter, all four algorithms perform very poorly, with MOCK suddenly being one of the worst performers. These results suggest that MOCK's good performance derives from a synergy between its initialization, representation and operators. This also seems apparent from the results obtained when comparing crossover innovation during a run of each MOEA, with the same for random solutions (see Section 4.4).

### 4.2 Size of the Search Space

In its original formulation, the size of the search space of the adjacency-based representation is bounded by $N^N$. However, the introduction of the nearest-neighbour mutation reduces the upper bound of the size of the search space seen by mutation to $L^N$, where, in our case, $L = 10$.

The Baseline representation and VIENNA result in a search space of $(k_{max})^N$ each, where $k_{max}$ is the upper limit on the number of clusters, which, in this work, has been set to $k_{max} = 50$. For Falkenauer's representation, the search space is of size $(k_{max})^N \times k_{max}!$.

### 4.3 Heuristic Bias

A representation is said to be biased if it leads to an over-representation of certain phenotypes: hence, when sampling the search space without any selection pressure, these phenotypes will have a larger probability of being generated [12]. Here, the presence of a bias is analyzed with respect to two different phenotypic properties: (i) the number of clusters of the solution; and (ii) the clustering quality of the solution.

In the Baseline representation and VIENNA, every phenotype is encoded by $\binom{k_{max}}{k} \times k!$ different genotypes. Furthermore, the number of phenotypes $S(N, k)$ with a fixed number $k$ of clusters is given by the Stirling number of the second kind. Consequently, the number of genotypes coding for a clustering solution with a fixed number of clusters $k$ is given by

$$R_{Baseline}(N, k) = \binom{k_{max}}{k} \times \sum_{i=1}^{k} (-1)^{k-i} \binom{k}{i} i^N.$$

For very small data sets ($n = 1, \ldots, 9$), the resulting distributions are illustrated in Figure 2 (right). The bias of Falkenauer's representation is closely related and is given as

$$R_{Falkenauer}(N, k) = R_{Baseline}(N, k) \times k!.$$

Hence, all three of these representations have a strong bias towards large numbers of clusters. On the other hand, it is clear that the adjacency-based representation suffers from a strong bias towards small numbers of clusters. In particular, for a data set of size $N$, the number of different genotypes coding for solutions with a fixed number of clusters $k$ corresponds to the integer sequence A060281 [14] and is computed as [4],

$$R_{MOCK}(N, k) = \sum_{i=k-1}^{N-1} \binom{N-1}{i} N^{N-1-i} [z^{k-1}](z+1)\ldots(z+i),$$

where $[z^{k-1}]$ is a coefficient operator, and means that only the coefficients of the terms with the specified powers of $z$ contribute to the sum. For very small data sets ($n = 1, \ldots, 9$), the resulting distributions are illustrated in Figure 2 (left).

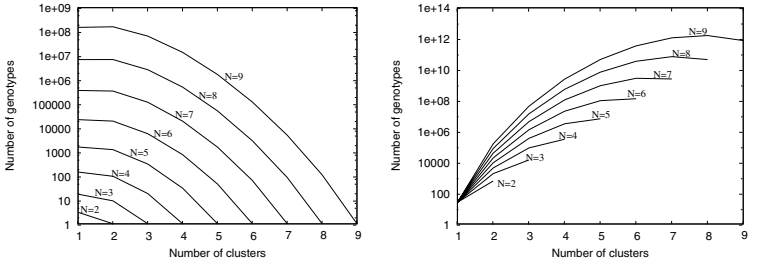


**Fig. 2.** Number of genotypes coding for a clustering solution for a fixed number of clusters for $N = 1, \ldots, 9$ data items. (Left) Adjacency-based representation. Note that, for increasing $N$, the maximum of the curve moves from $k = 1$ to $k = 2$, and this slow shifting to the right can be observed to continue for growing $N$. (Right) Baseline/VIENNA representation.

These calculations are confirmed by random sampling of the search space with neither selection pressure nor the use of specialized heuristic operators (results not shown). Figure 3, illustrates the selection pressure *implicitly* introduced through the use of specialized operators for clustering. The most striking results are MOCK's convergence to high numbers of clusters (caused by the nearest neighbour restriction in the mutation operator), the quick loss of diversity in Falkenauer's and the Baseline method (caused by the use of crossover) and the continuing exploration behaviour exhibited by VIENNA (due to the use of a large-scale mutation operator with strong heuristic bias).

### 4.4 Locality and Heritability

An analysis of locality and heritability of the operators can help to further understand the performance differences observed between the algorithms. This analysis only requires the definition of a distance in phenotype space [12], which is easily defined for the clustering task. A binary version of the Adjusted Rand Index is used to compare the similarity of two given clustering solutions. Note that, due to the use of a similarity (rather than a distance) measure, the following definitions are slightly different from those introduced in [12].

In order to assess heritability under crossover, crossover innovation $CI$ is defined as the phenotypic similarity between an offspring and its phenotypically closer parent:

$$CI = \max(s_P(x^c, x^{p1}), s_P(x^c, x^{p2})),$$

where $x^{p1}$ and $x^{p2}$ are the phenotypes of the two parents and $x^c$ is the phenotype of the offspring, and $s_P(,)$ is the similarity measure chosen.

Accordingly, the locality of the mutation operator is defined as the mutation innovation $MI$, which is the phenotypic similarity between solution $x$ and its mutant:

$$MI = s_P(x, x^m).$$

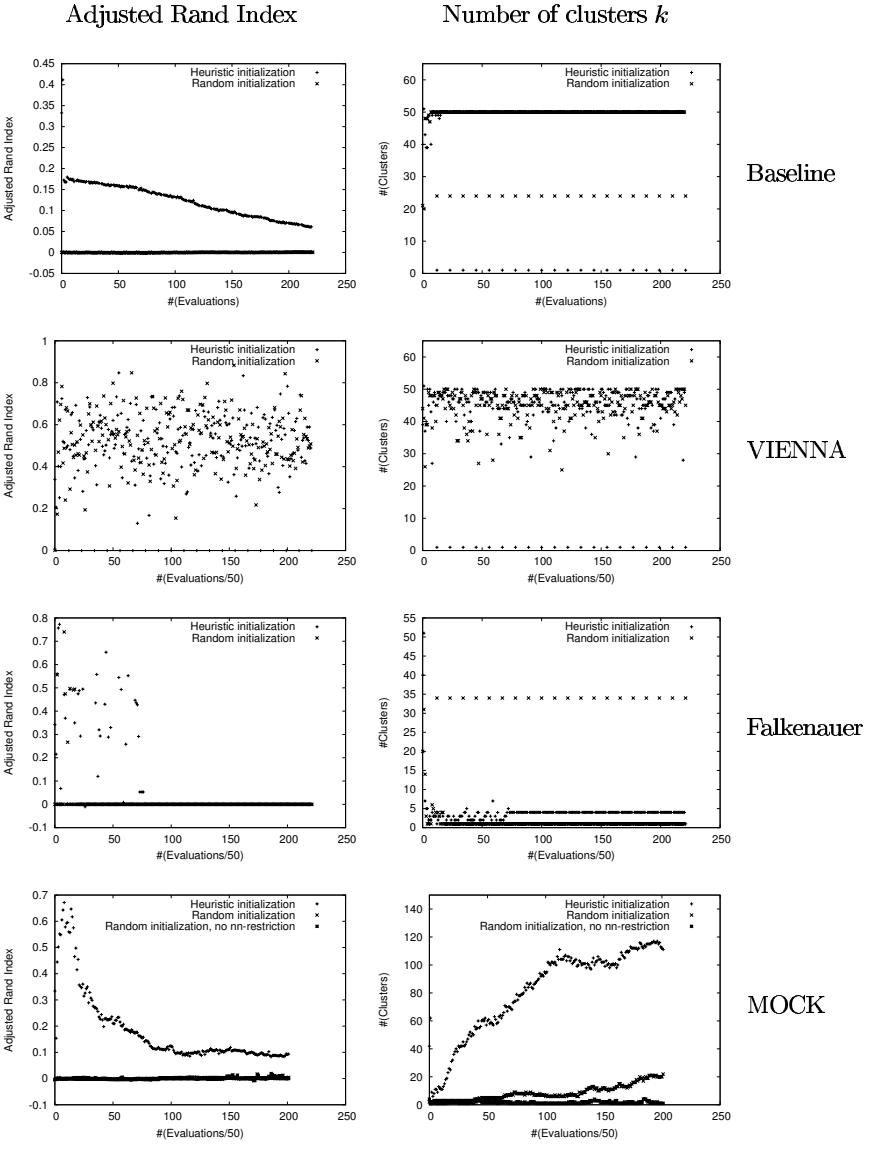


**Fig. 3.** Evolution of the number of clusters and the Adjusted Rand Index during a typical run of the MOEAs without selection pressure on one of the 2d-4c data sets

Figure 4 compares the distributions of crossover innovation and mutation innovation throughout the runs of the four MOEAs, and for randomly generated solutions. These plots illustrate that the reasons for the quick convergence of Falkenauer's representation in the absence of explicit selection pressure lie in the crossover operator's poor performance as a variation operator (it frequently creates identical copies of one parent). In contrast to this, MOCK's crossover

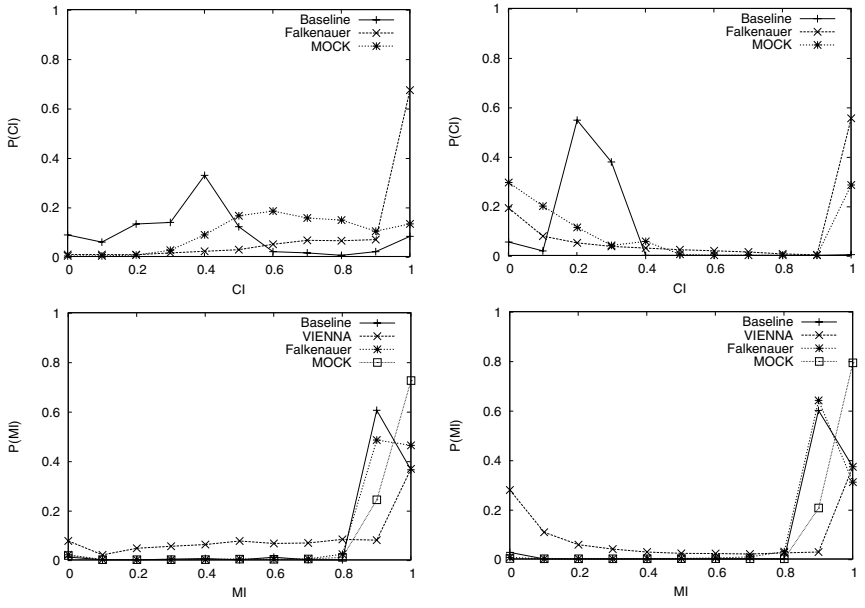


**Fig. 4.** (Top left) Crossover innovation in a normal run of the algorithms on a 2d-4c data set. (Top right) Crossover innovation for randomy generated solutions. (Bottom left) Mutation innovation in a normal run of the algorithm on a 2d-4c data set. (Bottom right) Mutation innovation for randomly generated solutions.

operator generates a large number of potentially interesting solutions (in the range $[0.4, 0.9]$), which may be the reason for its superior performance. Importantly, however, it fails to do so when random initialization is used, underlining once again that this representation relies crucially on the use of a powerful initialization scheme.

## 5 Conclusion

In this paper, four alternative representation/operator combinations for data-clustering with variable-$k$, have been investigated. We have focused on their use in the multiobjective clustering algorithm, MOCK, where it is necessary to be able to represent both compact clusters *and* locally-connected clusters equally well, and represent these for a large range of values of $k$. The analysis has revealed that, in this context, the simple adjacency-based representation, when combined with uniform crossover and nearest-neighbour mutation, performs very well, but only if a specialized initialization scheme is used. The more complicated scheme following Falkenauer is good if random initialization is used. Some of these results were reflected in the crossover innovation plots. The effects of the different representations' heuristic biases in phenotype space were also apparent under random selection, but these were overcome by the effects of standard selection and especially the use of the heuristic nearest-neighbour mutation.

**Acknowledgments.** We would like to thank Vladeta Jovovic and Neil Sloane for pointing us to A060281 in the online encyclopedia of integer sequences. JH acknowledges support of a doctoral scholarship from the German Academic Exchange Service (DAAD) and 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. R. M. Cole. Clustering with genetic algorithms. Master's thesis, University of Western Australia, Australia, 1998.
2. D. W. Corne, 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*, pages 283–290. ACM Press, New York, NJ, 2001.
3. E. Falkenauer. *Genetic Algorithms and Grouping Problems*. John Wiley and Son Ltd, New York, NJ, 1998.
4. I. P. Goulden and D. M. Jackson. *Combinatorial Enumeration*. John Wiley and Sons, Inc., New York, NJ, 1983. (Page 192, 3.3.28).
5. J. Handl and J. Knowles. Evolutionary multiobjective clustering. In *Proceedings of the Eighth International Conference on Parallel Problem Solving from Nature*, pages 1081–1091. Springer-Verlag, Berlin, Germany, 2004.
6. J. Handl and J. Knowles. An evolutionary approach to multiobjective clustering. *IEEE Transactions on Evolutionary Computation*, 2006. (In press).
7. A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data*. Prentice Hall, Englewood Cliffs, NJ, 1988.
8. P. C. H. Ma, K. C. C. Chan, X. Yao, and D. K. Y. Chiu. An evolutionary clustering algorithm for gene expression microarray data analysis. *IEEE Transactions on Evolutionary Computation*, 2006. (In press).
9. L. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297. University of California Press, Berkeley, CA, 1967.
10. Y.-J. Park and M.-S. Song. A genetic algorithm for clustering problems. In *Proceedings of the Third Annual Conference on Genetic Programming*, pages 568–575. Morgan Kaufmann, San Francisco, CA, 1998.
11. N. J. Radcliffe and P. D. Surry. Fitness variance of formae and performance prediction. In *Foundations of Genetic Algorithms 3*, pages 51–72. Morgan Kaufmann Publishers, San Mateo, CA, 1995.
12. G. R. Raidl and J. Gottlieb. Empirical analysis of locality, heritability and heuristic bias in evolutionary algorithms: A case study for the multidimensional knapsack problem. *Evolutionary Computation*, 13(4):441–475, 2005.
13. F. Rothlauf and D. E. Goldberg. Redundant representations in evolutionary computation. *Evolutionary Computation*, 11(4):381–415, 2003.
14. N. J. A. Sloane. Series A060281 in The On-Line Encyclopedia of Integer Sequences.
15. E. Zitzler. *Evolutionary algorithms for multiobjective optimization: methods and applications*. PhD thesis, Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, 1999.