Comparing initialisation methods for the heuristic memetic clustering algorithm

B.G.W. Craenen*,†, T. Ristaniemi*, A.K. Nandi†∗

*Department of Mathematical Information Technology, University of Jyväskylä, Jyväskylä, Finland
†School of Engineering & Design, Brunel University, London, UK

Abstract

In this study we investigate the effect five initialisation methods from literature have on the performance of the Heuristic Memetic Clustering Algorithm (HMCA). The evaluation is based on an extensive experimental comparison on three benchmark datasets between HMCA and the commonly-used k-Medoids algorithm. Analysis of the experimental effectiveness and efficiency metrics confirms that the HMCA substantially outperforms k-Medoids, with the HMCA capable of finding better clusterings using substantially less computation effort. The Sample and Cluster initialisation methods were found to be the most suitable for the HMCA, with the results of the k-Medoids suggesting this to be the case for other algorithms as well.

Index Terms— Machine Learning, Memetic Algorithms, Clustering, Heuristics

1. Introduction

Clustering is a data and signal processing task with the objective to determine a finite set of categories, called clusters, to describe a dataset according to the similarities among its data objects [1, 2]. Applications of clustering are many and range from market segmentation [3] and image processing [4], to document categorisation and web mining [5]. Recently clustering has gained prominence for signal processing in the field of bioinformatics [6–9].

Clustering can then be defined as follows. A dataset, \( X \), has \( N \) data objects: \( X = \{x_1, \ldots, x_N\} \), with each data object, \( x_i \), having \( n \) features or attributes: \( x_i = \{x_1^i, \ldots, x_n^i\} \).

A configuration of clusters, or clustering, \( C \), has \( k \) clusters: \( C = \{c_1, \ldots, c_k\} \), with each cluster, \( c_i \), a subset of data objects from the dataset: \( c_i = \{x_1, \ldots, x_{|c_i|}\} \), with \( x_i \in X \).

Clusters are not allowed to be empty, \( c_i \neq \emptyset \), or overlap, \( c_i \cup c_j \cup \ldots \cup c_k = X \) and \( c_i \cap c_j \neq \emptyset \) for \( i \neq j \).

Clustering algorithms seek to assign all data objects of a dataset to clusters and, from an optimisation perspective, strive to maximise the homogeneity of data objects within a cluster, while simultaneously maximising the heterogeneity between clusters [4, 10]. Measuring similarity (homogeneity) is tackled indirectly by using a distance measure quantifying the degree of dissimilarity (heterogeneity) between data objects. Usually distance measures are defined so that similar data objects have lower dissimilarity values, while dissimilar objects have higher values [4, 10].

In Machine Learning, clustering is deemed to be one of the most difficult and challenging problems, mostly due to its unsupervised nature, and the implication that the structural characteristics of the dataset remain unknown [11]. Clustering is formally considered a particular kind of NP-hard grouping problem [12]. This has stimulated the search for efficient approximation algorithms, especially, for more general meta-heuristics like Evolutionary Algorithms (EAs), and Memetic Algorithms (MAs), capable of evolving near-optimal solutions in reasonable time. Many clustering EAs and MAs have been proposed in literature (see [13]).

This study investigates the effect five initialisation methods from literature have on the performance of the Heuristic Memetic Clustering Algorithm (HMCA), as well as the relative effect this has when compared with the k-Medoids. In [14] we designed the HMCA as an easy-to-understand yet powerful clustering algorithm, adaptable for a wide variety of clustering problems and dataset. The basis for this adaptability lies in its single local-search operator, incorporated into the MA meta-heuristic. By applying this operator with different heuristics, the HMCA can be tailored to a wide variety of clustering problems and datasets. In [14], the performance of the HMCA was compared with the commonly used k-Medoids algorithm on three benchmark datasets. The HMCA was found to be substantially more effective in finding good clusterings, and efficient in being able to do so using fewer computational resources.

For all its demonstrated effectiveness and efficiency in clustering datasets, further analysis of the experimental results showed that HMCA’s initial clusterings were often of substantially worse quality than those available to k-Medoids. Subsequently the HMCA had to expend significant computational effort catching up. The HMCA being stochastic, its ability to invariably do so accentuates its power and perfor-

This work was financially supported by TEKES (Finland) under grant 40334/10 “Machine Learning for Future Music and Learning Technologies” (MUSCLES). Asoke K. Nandi would like to thank TEKES for the award of the Finland Distinguished Professorship.
The HMCA was first presented in [14] and we refer to that publication for a detailed description of the algorithm. Because of the limited space available, we provide only a summary description of the HMCA, sufficient for understanding the issues raised here.

The HMCA is a MA for clustering datasets. MAs are a subset of EAs, in that they resemble EAs but include a local-search component. MAs are sometimes referred to as Baldwinian EAs, Lamarckian EAs, Cultural Algorithms, or Genetic Local-Search Algorithms [16]. The HMCA’s local-search component is the single local-search operator using heuristics to guide the evolutionary process towards better clusterings.

The HMCA evolutionary process closely follows the canonical EA process: Clusterings are encoded in individuals, and an initial population of individuals is initialised using an initialisation method. Each individual has a value quantifying its quality or fitness, assigned by the fitness function. The HMCA fitness function calculates the cumulative inner-distance of a clustering:

\[ f(C) = \sum_{c_i \in C} \sum_{l=1}^{|c_i|-1} \sum_{m=l+1}^{|c_i|} d(x_l, x_m) \]  

with \( d(x_l, x_m) \) the Euclidean distance between data object \( x_l \) and \( x_m \) as a distance metric. The fitness value is to be minimised, i.e., good clusterings have tight clusters.

The HMCA then iteratively approximates towards global optimal clusterings by applying a single local-search operator to each individual in the (parent) population. This operator generates one child individual for each parent individual by reassigning one data object from one cluster to another. Empty clusters are avoided, but when they occur are repaired by moving a random data object from the biggest cluster to the empty cluster (iteratively if there is more than one empty cluster). The (repaired) offspring individuals are then assigned a fitness value, and placed in the offspring population. Both parent and offspring populations are merged and duplicate clusterings (based on cluster content) are discarded. Sorted by fitness values, the merged population is pruned to a size equal to the population size parameter by discarding the worst individuals, thus creating the parent population for the next iteration of the HMCA.

The iterative process is halted when the HMCA has exhausted a set maximum number of distance calculations. Distance calculations are used by the fitness function, but also by the local-search operator, and by the initialisation method.

The innovative feature of the HMCA is its heuristic local-search operator. It uses up to three heuristics to select a data object, and then to select a cluster to subsequently assign it to. Selecting the data object may involve two heuristics: one to first select a cluster, from which another then selects a data object, or, alternatively, just one selecting a data object from all data objects in the clustering without regard to which cluster it is assigned. The three heuristic types are then: one for selecting a (source) cluster, one for selecting a data object, and one for selecting label (or destination cluster).

All three heuristic types have a benchmark or reference heuristic, called a null-heuristic. These are, respectively: select no cluster (data objects are selected from all available), select a random data object, and select a random label. Four heuristics are defined for both the cluster and data object heuristic types, while six are defined for the label heuristic type. These heuristics are defined around distance metrics, and use distance calculations, with selection either deterministic or proportionally random. We refer to [14] for precise definitions of all heuristics. Including the null-heuristics, the HMCA uses a total of five cluster and data object heuristics, and seven label heuristics.

The HMCA then has the following input parameters: a dataset to cluster, the number of clusters to cluster it in, the size of the population to use, the maximum number of distance calculations, and the three heuristics used by the local-search operator. A HMCA variant for a dataset is be defined by the three heuristics used by the local-search operator. Some HMCA variants are expected to have superior performance than others, and differing performance for different datasets.

3. INITIALISATION METHODS

Originally, in [14], the HMCA used a two-step approach for initialising individuals: first, all clusters were assigned a unique, randomly chosen data object; then all remaining data objects were assigned to random clusters. This initialisation method avoids initial individuals having empty clusters, but the random assignment of the remaining data objects is also likely to result in unfavourable initial partitions. Mostly be-
cause the resulting clusters are likely to be mixed up to an unfavourably high degree.

The likelihood of producing unfavourably, mixed-up initial clusters was already mentioned in [13], which then goes on to state that this does constitute an effective approach for testing algorithms against hard evaluation scenarios. The authors agree, but this seems rather of limited interest to those simply wishing to evolve good quality clusterings. Furthermore, this drawback makes for rather unfair comparisons between EAs (and MAs) and other algorithm types who need not suffer from this. So, while it is comforting to the find that the HMCA was still capable of comparatively superior performance in [14], in this study, similar to [15], we investigate the (comparative) performance of the HMCA when the original initialisation method is replaced with one of the following five methods:

**Sample** Like the original initialisation method, Sample first assigns each cluster a unique randomly selected data object as a medoid, but then assigns the remaining data objects to the cluster with the least distance between itself and the medoid.

**Cluster** This initialisation method takes a 10% randomly selected sample of all data objects, and then uses a single iteration of \( k \)-Medoids to find \( k \) medoids among them. These are assigned to each cluster, with the remaining data objects assigned as with Sample.

**Uniform** This method calculates the value ranges of each attribute by taking the minimum and maximum values for each attribute for each data object in the dataset. \( k \) new data objects are then generated by taking a random value from a uniform random distribution between these ranges for each attribute. These new data objects are then matched to data objects in the dataset by selecting the one closest to it, and used as medoids for each cluster. The remaining data objects are assigned as with Sample.

**KA0 and KA1** These two initialisation methods are based on the Kaufman Approach (KA) [1]. A deterministic constructive approach, this initialisation method essentially compares the pair-wise distance between all data objects, subtracting from it the minimum distance to the data object and already selected medoids. Using this measure, the KA iteratively selects \( k \) medoids that maximise the heterogeneity in the dataset’s hyperplane. The difference between KA0 and KA1 reflects an interpretation difference in the description in [1]. KA0 uses accumulative distances between data objects, while KA1 uses single distances. The remaining data objects are assigned as with Sample.

4. EXPERIMENTAL SETUP

The experimental setup used here is similar to the one used in [14]. The initialisation methods described in Section 3 were implemented for use in both the HMCA and \( k \)-Medoids. The \( k \)-Medoids algorithm was further adapted to use the same performance measures as the HMCA, making a direct comparison both possible and fair.

Both the effectiveness and efficiency of both algorithms are measured. Effectiveness is an expression of the ability of the algorithms to find (evolve) good quality clusterings of the dataset. Clustering quality in the HMCA is assessed by the fitness value, i.e., the cumulative inner-cluster distance. The same measure is used by \( k \)-Medoids, but, since maintaining fitness values is not integral to \( k \)-Medoids, the distance calculations required for calculating it are not included when measuring efficiency.

Efficiency is an expression of the amount of computational effort required by the algorithm to find good clusterings. Since the commonly used wallclock time to solution metric is affected by range of unquantifiable factors (such as computing platform, implementation quality, etc.), as in [14], we use the number of distance calculations as the atomic efficiency measure instead.

As in [14], we use three commonly-used datasets to evaluate the performance of both algorithms: the Iris flower dataset (Iris), the Glass Identification dataset (Glass), and a generated Quadrature Amplitude Modulation dataset (16-QAM). The Iris dataset is a collected multivariate dataset of three \( k \) Iris flowers for a total of \( N = 150 \) data objects with \( n = 4 \) measured features (attributes) each [17]. The Glass dataset is a multivariate dataset consisting of \( n = 9 \) collected chemical compositions of glass, for a total of \( N = 217 \) data objects, clustered into \( k = 6 \) clusters of variable size [18]. The 16-QAM dataset is a generated dataset consisting of a \( N = 1024 \) data object signal stream, modulated randomly on a \( 4 \times 16 = 64 \) two dimensional rectangular grid (\( k = 16 \)), giving each data object \( n = 2 \) features (coordinates). After generation of the signal stream, 12 dB of Gaussian noise was added. With different numbers of clusters to be found, with different degrees of overlap between those clusters, and varying numbers of data objects and features, we believe that together these datasets provide a diverse environment to evaluate and compare both algorithms on.

Parameters for the algorithms then include: the number of clusters (\( k = 3 \) for Iris, \( k = 6 \) for Glass, and \( k = 16 \) for 16-QAM), population size (for HMCA: \( \{1, 2, 5, 10 \} \)), and all combinations of the heuristics for HMCA (5 · 5 · 7 = 175 HMCA variants). Both algorithms are stochastic, and were repeatedly run 25 times to gain sufficient statistical accuracy. They were both given 10,000,000 distance calculations to cluster the Iris and Glass datasets, and 20,000,000 distance calculations for the 16-QAM dataset. Preliminary experimentation showed that these values allowed both algorithms sufficient computational effort for a fair comparison.

5. RESULTS

Both the effectiveness and efficiency of both algorithms can be assessed by examining the fitness value per distance cal-
In this study we investigated the effect five initialisation methods from literature have on the performance of the HMCA and \( k \)-Medoids. The five initialisation methods are defined, implemented, and applied to both the HMCA and \( k \)-Medoids, the latter extended to make the comparison fair. The comparison evaluates both algorithms on both effectiveness (ability to find good clusterings) and efficiency (ability to find them fast).

6. DISCUSSION & CONCLUSION
An analysis of the experimental results leads us to conclude that the HMCA has superior performance, both in effectiveness and efficiency, over k-Medoids. Particularly the Sample and Cluster initialisation methods were found to have superior performance, on all three datasets. Using these initialisation methods the HMCA was capable of consistently finding global optimal clusterings, while using substantially less computational effort to do so.

The performance of the KA0 and KA1 initialisation methods were particularly disappointing, for both algorithms. These initialisation methods are often used because they are deterministic, limiting stochasticity in the experimental process. However, both methods are also expensive, and become exponentially so as the number of data objects in the dataset increase. We found that this became a prohibitive factor for the 16-QAM dataset, and moreover found that the increased computational effort did not lead to a subsequent performance increase for both algorithms.

When computation effort available is limited (as it almost always is), additional effort spend on initialisation reduces the amount available for evolving the generated clusterings. This must therefore be offset by the quality of the generated clusterings. Our findings show this did not work for the KA0 and KA1, and, to a lesser extend, for the Uniform initialisation methods. In contrast, the Sample and Cluster initialisation methods did get this balance right for the HMCA, and, given their superior performance with k-Medoids as well, we expect that this will be the case for other algorithms as well.

Limited available space has lead us to, thus far, neglect one questions: which HMCA variants have the best performance for which dataset. A succinct answer would go as follows: the curves shown in Figure 1 for the HMCA are compound curves, showing the performance of all 175 HMCA variants merged into one curve. In [14] we found that the best performing HMCA variants were those that were relaxed (more random) about cluster and data object selection, but specific (deterministic) about label selection. Analysis of the results for this study did not alter that conclusion, i.e., similar HMCA variants had the best performance in [14], as well as here.

REFERENCES


