Multiobjective Clustering with Automatic k-determination for Large-scale Data

Scalable automatic determination of the number of clusters

Nobukazu Matake
Doshisha University graduate school, Kyoto, Japan
matake@mikilab.doshisha.ac. jp

Tomoyuki Hiroyasu
Doshisha University
tomo@is.doshisha.ac.jp

Mitsunori Miki
Doshisha University
mmiki@mail.doshisha.ac.jp

Tomoharu Senda
Doshisha University
tsenda@mikilab.doshisha.ac.jp

ABSTRACT

Web mining - data mining for web data - is a key factor of web technologies. Especially, web behavior mining has attracted a great deal of attention recently. Behavior mining involves analyzing the behavior of users, finding patterns of user behavior, and predicting their subsequent behaviors or interests. Web behavior mining is used in web advertising systems or content recommendation systems. To analyze huge amounts of data, such as web data, data-clustering techniques are usually used. Data clustering is a technique involving the separation of data into groups according to similarity, and is usually used in the first step of data mining. In the present study, we developed a scalable data-clustering algorithm for web mining based on existent evolutionary multiobjective clustering algorithm. To derive clusters, we applied multiobjective clustering with automatic k-determination (MOCK). It has been reported that MOCK shows better performance than k-means, agglutination methods, and other evolutionary clustering algorithms. MOCK can also find the appropriate number of clusters using the information of the trade-off curve. The k-determination scheme of MOCK is powerful and strict. However the computational costs are too high when applied to clustering huge data. In this paper, we propose a scalable automatic k-determination scheme. The proposed scheme reduces Pareto-size and the appropriate number of clusters can usually be determined.

Categories and Subject Descriptors

I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search—*Heuristic methods*;
I.5.3 [**Pattern Recognition**]: Clustering—*Algorithms*

General Terms

Algorithms, Performance

Keywords

Multi-objective optimization, Pattern recognition and classification, Data mining, Speedup technique

1. INTRODUCTION

We are studying data clustering for web behavior analysis. Data clustering is a technique of separating huge datasets into groups according to their similarities. Web behavior analysis is performed to find characteristic behavior patterns of users and predict their subsequent behaviors. Web behavior analysis is the key factor behind personalization and effective ad delivery systems, etc. This clustering technique is very important to allow the analysis of millions of sets of web behavior data.

There are many techniques for data clustering, like $k-$means, the agglutination method, and other methods based on EAs, etc[3][4][5]. The main points involved in clustering are determining the appropriate number of clusters[1], and finding the partitions among k clusters. k-means is better at finding partitions but worse at determination of appropriate k, while, the agglutination method is better at determination of appropriate k but worse at finding partitions. Each method uses a single objective, the objective of k-means is compactness of clusters and that of agglutination is connectedness of similar data.

In our study, we used a multiobjective evolutionary clustering algorithm with these two objectives. We used the multiobjective clustering algorithm proposed by Handl and Knowles[1][2], MOCK (Multiobjective clustering with automatic $k-$determination)[1]. This algorithm optimizes two complementary objectives based on cluster compactness and connectedness. It has been reported that MOCK finds the partitions better than k-means, agglutination methods, and other evolutionary clustering algorithms. It also has an automatic k-determination scheme and simultaneously finds the appropriate number of clusters and partitions among the k clusters. MOCK is one of the most powerful and interesting algorithms for data-clustering problems.

[1]usually called k

One goal of our research was to apply MOCK to web data clustering. However, the original algorithm is difficult to use for web data, because of the limits of the computation cost. Especially, the cost of the automatic k-determination scheme is huge. Therefore we investigated a scalable automatic k-determination scheme. In this paper, we propose the scheme. The remainder of this paper is organized as follows. Section II presents a review of MOCK. In Section III, we describe the scalability of the original MOCK algorithm and our new scheme. The effectiveness of our scheme is discussed in Section IV. Finally, Section V concludes this paper.

2. MULTIOBJECTIVE CLUSTERING WITH AUTOMATIC K-DETERMINATION

MOCK is a powerful clustering algorithm based on a multiobjective genetic algorithm. It optimizes two complementary objectives based on cluster compactness and connectedness. One is intended for determination of appropriate k, and the other is to find partitions between k clusters. In this evolutionary multiobjective clustering algorithm, Pareto solutions are a set of different trade-off partitioning over a range of different cluster numbers, k. This algorithm also is able to determine the final solution with good accuracy and diversity.

In this section, we describe the MOCK algorithm, graph-based representation of partitioning, initialization using a minimum spanning tree (MST), and two objective functions. The representation allows treating solutions, which contain different numbers of clusters, and the MST-based initialization makes well-connected initial solutions like those made by the agglutination method. After optimizing two objective functions through genetic operations - crossover, mutation, and selection - Pareto solutions with a range of different k are made. Finally, we describe Pareto of MOCK, and the automatic k-determination scheme. The Pareto solutions contain different trade-off partitioning over a range of different k, and the automatic $k-$determination scheme is able to determine the final solution from the Pareto solutions, and find the appropriate k.

2.1 Graph-based representation

MOCK uses graph-based representation[6] in which each datum is represented as a node, and an edge between two nodes indicates that these data are in the same cluster. This allows treatment of a wide range of k in a single trial.

In Figure 1, each datum is numbered $i \in \{1..9\}$. The dotted lines indicate removed edges. Removing the edges from 8 to 3 and from 6 to 8 separates the nodes into three clusters. All k solutions are represented in this way. Removing edges is the key to creating clusters. It is effective in the initialization scheme with a minimum spanning tree.

A gene is represented as the list of edges. The phenotype is the list of edges in the directed graph, and the genotype is the list of node ids to which each edge is connected. In this representation scheme, each node has only one edge connecting from it.

2.2 Initialization using minimum spanning tree

A minimum spanning tree (MST) is the *shortest* tree that contains all nodes in the graph and has no loops. The *shortest* tree means that the total cost of all edges is the minimum. In the MST created for a given dataset, similar data are connected and dissimilar data aren't. The MST of Figure 1 is shown at the top of Figure 2. Removing $k-1$ edges results in the creation of a k cluster. The removed edges are selected according to their cost; the edges with higher cost are removed. In this way, MOCK makes high-quality initial solutions[2]. (See bottom of Figure 2)

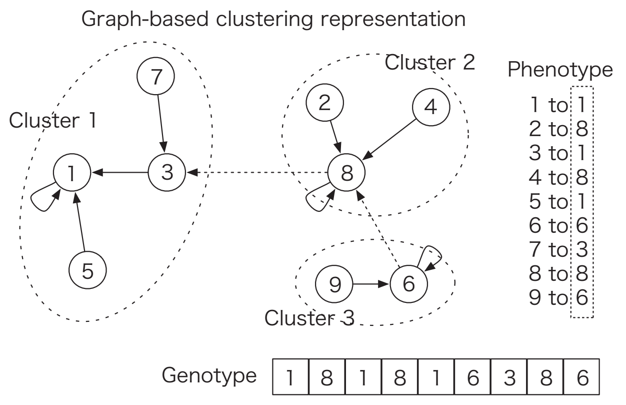


Figure 1: Graph-based representation

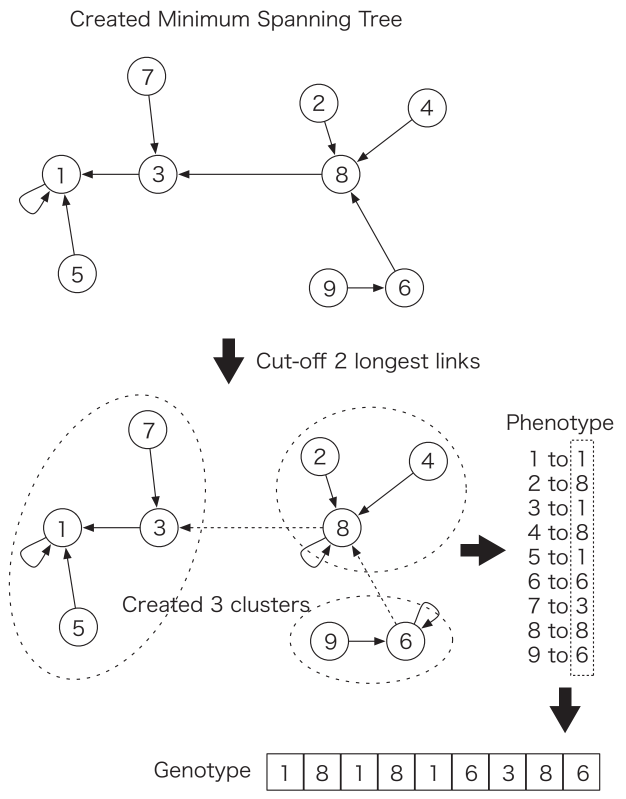


Figure 2: MST-based initialization

MOCK uses Prim's algorithm[8] to create the MST. This algorithm has order $N \log N$ time complexity.

[2]The initial solutions of MOCK are similar to those created by the agglutination method.

2.3 Objective functions

The two objectives of MOCK are connectivity and overall deviation. Connectivity is based on cluster connectedness, and overall deviation is based on cluster compactness.

2.3.1 Connectivity

Connectivity is intended to divide similar data into the same cluster. It is defined as equation (1), where i is the data id, C is the total dataset, C_k is the kth cluster, and $nn_i(j)$ is the ordinal number[3].

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

Connectivity is minimized. When all $similardata$[4] are clustered in the same cluster, connectivity of the solution becomes zero. When these data are in different clusters, connectivity increases $1/j$. Therefore, minimizing connectivity decreases k. When $k = 0$, connectivity is always zero.

2.3.2 Overall deviation

Overall deviation, defined as equation (2), is intended to make clusters more compact. $\delta()$ is the distance function designed to calculate the similarity between every datum in the dataset. μ_k is the center coordinate or the median datum of kth cluster.

$$Dev(C) = \sum_{C_k \in C} \sum_{i \in C_k} \delta(i, \mu_k) \quad (2)$$

Overall deviation is also minimized. More compact separation makes this value smaller. Therefore, minimizing overall deviation increases the number of clusters. When separating N data into N clusters, overall deviation is zero.

Minimizing these two objectives, a wide range of different k solutions are generated. This is efficient to find the appropriate number of clusters, k. For each k, overall deviation is minimized and most compact clusters are made. Therefore, MOCK returns a range of different k solutions, and each solution is well separated.

2.4 Pareto solutions

While reduction of overall deviation, each cluster size becomes smaller and k becomes larger. In contrast, with reducing connectivity, each cluster size becomes larger because many neighbor data are collapsed into same clusters, and k become smaller. Therefore, Pareto of MOCK contains solutions over a wide range of k, as shown in Figure 3. In Figure 3, smaller k solutions are on the top left and more k solutions are on the bottom right.

The Pareto curve shows the relation between k and $\delta D/\delta C$, where C is connectivity and D is overall deviation. This relation is used in the next step, the automatic k-determination scheme.

[3] $nn_i(j)$ is the jth nearest data from i

[4] Data within Lth nearest each datum.

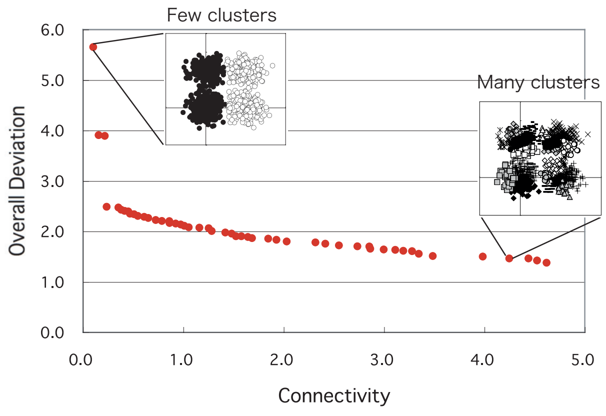


Figure 3: Pareto of MOCK

2.5 Automatic k-determination scheme

The automatic k-determination scheme of MOCK is based on the Gap statistic[7]. In this scheme, we make random data [5] based on principal component of the original data, and divide these data into clusters by MOCK. The Pareto curves of the random and original data are different. In the original Pareto, the most distant solution from the nearest solution of all Control solutions is selected as the most characteristic solution. In the Gap statistic, we guess that such a solution has appropriate k. (Figure 4)

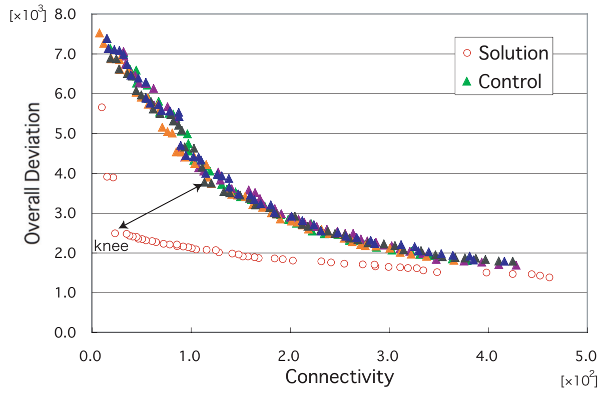


Figure 4: Automatic k-determination scheme of MOCK

As described above, the k-determination scheme of MOCK requires a random data clustering step. The calculation cost is as high as when clustering the provided dataset. If using the random data clustering step in M trial, the original calculation cost is $M + 1$ times as large. This computational cost is very high, especially in large-scale dataset such as web data. Therefore, we propose a scalable k-determination scheme without the random data clustering step in the next section.

[5] called $Control$ data

3. IMPROVEMENT OF SCALABILITY OF THE K-DETERMINATION SCHEME

Determination of the number of clusters is one of the most important concerns in the clustering problem. In this process, the k-determination scheme of MOCK based on the GAP statistic works well. However, as mentioned in the previous section, the original scheme has a problem. The computation costs for determining the cluster number in the original MOCK scheme are high. The cost is serious problem when classifying over a million data.

In this section, we describe the cost of the original k-determination scheme. In addition, we propose new scalable k-determination scheme, which uses the information of the Pareto curve. In our scheme, the relation between connectivity and overall deviation is very important.

3.1 Relation of connectivity and overall deviation

In Figure 3, fewer k solutions are on the top left, and more k solutions are on the bottom right. We can consider the Pareto curve as a function of k. When Pareto solutions contain the solution of the appropriate k, the smaller k solutions are on the left of the appropriate k solution, and larger on the right.

The left figure in Figure 5 shows the situation where the data should be separated into two clusters. In this situation, when the cluster is separated into two, the value of connectivity is small and the overall deviation becomes smaller. On the other hand, the figure on the right shows the situation where the data should not be separated. In this situation, when the cluster is separated into two, the overall deviation becomes smaller but the value of connectivity increases. This is because the number of data penalized becomes very large. Therefore, $|\delta D/\delta C|$, where C is connectivity and D is overall deviation, rapidly decreases near the appropriate k solution. This *knee*[9] is shown in Figure 4.

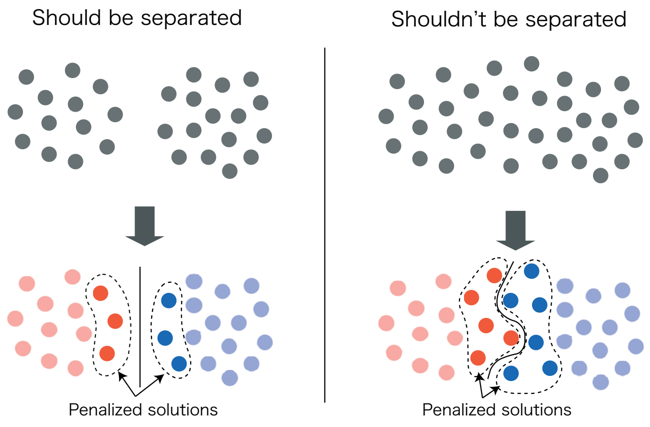


Figure 5: Should be separated or not

3.2 Scalable k-determination scheme

Figure 6 shows the operation flow of MOCK and our new k-determination scheme. Our proposed scheme is composed of two steps: a Pareto-size reduction step and a k-determination step. In the first step, removing the local non-convex part of solutions reduces the Pareto size. This reduction step helps to determine the appropriate k more easily in the next step. The second step is also composed of two sub-schemes. The first scheme usually determines the appropriate solution, although it makes errors in several situations. The second scheme overcomes the defect of the first. This set of k-determination sub-schemes is applied in parallel. Each scheme is applied to same reduced Pareto solutions, and returns each final solution. When the returned solutions are not the same, the user chooses either one of the two.

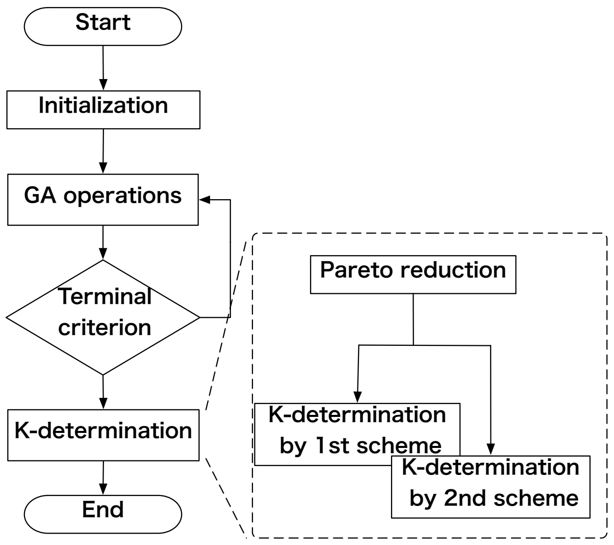


Figure 6: The flow of our automatic k-determination scheme

3.2.1 *Reduction step*

The set of Pareto solutions obtained by the original MOCK algorithm contains several same-k solutions, and low-potential solutions. These solutions are removed in this step. (Figure 7)

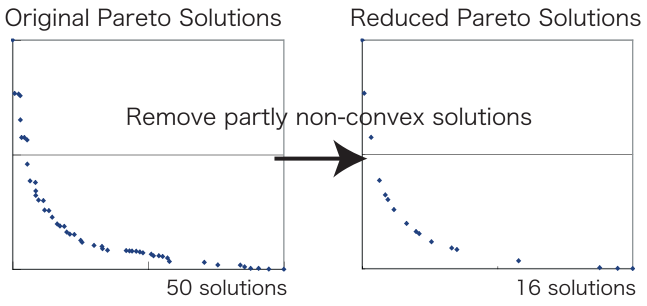


Figure 7: Reduction of Pareto size

We empirically assume that Pareto is convex. The criterion to remove is that the solution is in a partly non-convex part. Solutions in the partly non-convex part do not have more compact clusters than those of the nearby convex part. Figure 8 shows the convex and non-convex sample solutions in Pareto of Figure 7. S1-S3 are part of the Pareto solutions returned by MOCK, and all these solutions are obtained when $k = 3$. S1 and S3 are in a partly non-convex part of Pareto and S2 is in a convex part.

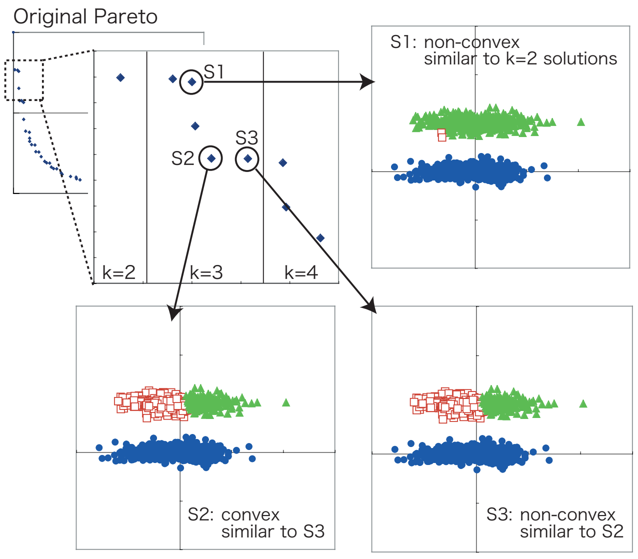


Figure 8: Non-convex and convex part of solutions

In S1, almost all data are divided into the two largest clusters, and a few data are divided into the smallest cluster. This solution is similar to those obtained when k=2. Such a solution is worse than the k=2 solutions on the left, because the smallest cluster contains too few data and has little information. Solutions S2 and S3 seem to be the same. However, connectivity is not the same, so the border of the above two clusters is more efficient in S2 than in S3. Thus, S1 and S2 are lower-potential solutions than S2. The flow of this step is as follows.

1. Normalize overall deviation and connectivity to [0, 1] - [1, 0].
2. Sort solutions by connectivity.
3. Calculate adjacent angles. When it is non-convex, remove the solution. This step is iterated until no more solutions are removed.

3.2.2 *k-determination step: first scheme*

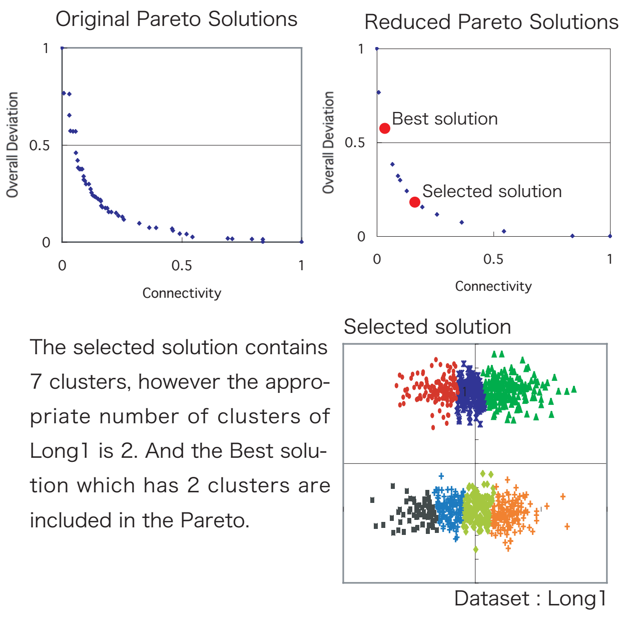
Our k-determination scheme uses the information of the Pareto curve, $|\delta D/\delta C|$. We determine the solution on the *knee* of the Pareto curve as the final solution. This is not as powerful as the original MOCK scheme[6]. However, this is a very simple and efficient method to determine the appropriate k, which does not require the random data clustering step. The process is described below.

1. Reduction step.
2. Calculate adjacent angles and determine the *knee*.

3.2.3 *k-determination step: second scheme*

Figure 9 shows the failure pattern of the first sub-scheme. This dataset is composed of non-circular clusters.

[6]This scheme makes errors when the typical *knee* is not shown in the Pareto, although the reduced Pareto contains the appropriate k solution.



The selected solution contains 7 clusters, however the appropriate number of clusters of Long1 is 2. And the Best solution which has 2 clusters are included in the Pareto.

Figure 9: Failure pattern of the first sub-scheme

Such non-circular clusters foil the first sub-scheme for the following two reasons.

1. Overall deviation decreases any time separating clusters, and circular clusters are more efficient than non-circular ones using the Euclidean distance.
2. When cutting such a wide cluster, penalized data in connectivity function are fewer than circular clusters.

The following steps are our basic solution to solve this problem.

- When separating a wide non-circular cluster, keep $|\delta D/\delta C|$ the same as when it is circular.
- This adjustment is done in overall deviation function, because the connectivity function uses ordinal numbers and it is difficult to change the value.

We accept the scaling filter to the overall deviation function, after obtaining the Pareto solutions. The flow below is our filter method.

1. Reduction step.
2. Make each cluster of each solution non-correlated by PCA.
3. Calculate variance of each dimension of each cluster, and scale down each dimension to the minimum-variance dimension.
4. Re-valuate overall deviation of each solution, and create new Pareto.
5. Determine the new *knee*.

We can treat non-circular clusters in this sub-scheme.

4. ASSESSMENT EXPERIMENT

4.1 Proposal of the experiment

We performed two experiments. The first experiment was performed to investigate the performance of k-determination of our new scheme. For this purpose, we used 9 characteristic test datasets: basic circulate clusters, overlapping clusters, non-circular clusters, circular and non-circular clusters, different distribution in clusters. We applied our first and second sub-scheme to each dataset, and investigated whether the appropriate k solution was selected by each scheme, and the size of the reduced Pareto. Notice that, in all test datasets used, the appropriate k solution was always created by MOCK and these were in the Pareto. In the second experiment, we applied our scheme to a large-scale dataset for investigating the scalability.

4.2 Dataset

We used 9 test datasets employed in [1]. Their names, size[7] and appropriate k are listed in Table 1, and Figure 10 shows an overview. They are all 2D continuous-valued datasets.

Table 1: Test dataset

Dataset	*Size*	*Appropriatenumberofclusters*
Square1	1000	4
Square4	1000	4
Sizes5	1000	4
Triangle1	1000	4
Long1	1000	2
Twenty	1000	20
D2C10	2882	10
Spiral	1000	2
LongSquare	900	6

Square1 and Square4 consist of four circular clusters of equal size and spread. They differ in the overlapping of clusters. The sizes of clusters in Sizes5 are not equal. Triangle1 consists of a non-circular cluster. Long1 consists of only non-circular clusters. Twenty has twenty clusters of equal size, circularity, and spread. D2C10 consists of ten non-circular and circular clusters of different size and spread. LongSquare consists of Long1 and Square1. We also use Square1, which contains 100,000 data for the second experiment.

4.3 Parameter settings

The parameters used in this experiment are listed in Table 2. These are recommended values for MOCK. (See [10]) In our MOCK, the internal population size is equal to the external population size, because we used archive copy selection and neighborhood crossover, proposed by Watanabe[14], as the internal population selection and for crossover of MOCK, respectively. We also used SPEA2[11] as the base multiobjective genetic algorithm of MOCK. C++ package of SPEA2[12] is utilized for implementation.

[7] *i.e.*, number of data

Table 2: Parameter settings for MOCK

Parameter	*Setting*
Gene length(N)	dataset size
Number of generations	200
Internal population size	min(50, 2*k_{max})
External population size	min(50, 2*k_{max})
Initialization	Minimum Spanning Tree
Crossover	Uniform and neighborhood crossover
Crossover rate	0.7
Mutation type	L nearest neighbors($L = 20$)
Mutation rate	$1/N + (l/N)^2$
Constraints	$k \in \{1, .., 25\}$, cluster size>2

4.4 Experimental results

In the first experiment, we investigated the performance of auto k-determination and reduction rate of Pareto size when applying our proposed scheme to MOCK. Figure 11 shows the k of the selected solution by our two sub-schemes. The thick line is the appropriate k for the dataset, which is described in Table 1. In Square1, Square4, Sizes5, Triangle1, and Twenty, the first scheme found the appropriate k perfectly. However, in Long1 and Spiral, the first scheme did not find appropriate k. In these cases, the second sub-scheme covered the first. In this result, either of our two sub-schemes can find the appropriate k.

Comparison of this result and [1] indicated that the performance of our scheme is worse than that of the original scheme. However, our scheme can be used without random data clustering, which can reduce the calculation costs over 50% as compared with the original MOCK. This is one of the most important factors in web mining.

Table 3 shows the reduced Pareto size (average of 10 trials). Over the whole dataset, about 70% of solutions are removed in the first sub-scheme, and 50% more are removed in the second. In addition, in the case in which the auto k-determination scheme makes a mistake in identifying k, the appropriate k solutions remained in the reduced Pareto. This result indicated that our scheme reduced the Pareto size by about 50-70%. It is efficient in cases in which an expert selects the last solution when they are not satisfied with the selected solution.

Table 3: Reduction rate of Pareto size

Dataset	*1stscheme*	*2ndscheme*
Square1	76.4%	60.8%
Square4	77.8%	49.2%
Sizes5	74.2%	60.4%
Triangle1	73.0%	74.6%
Long1	70.4%	69.6%
Twenty	84.7%	72.5%
D2C10	74.4%	57.6%
Spiral	74.4%	74.2%
LongSquare	68.0%	58.8%

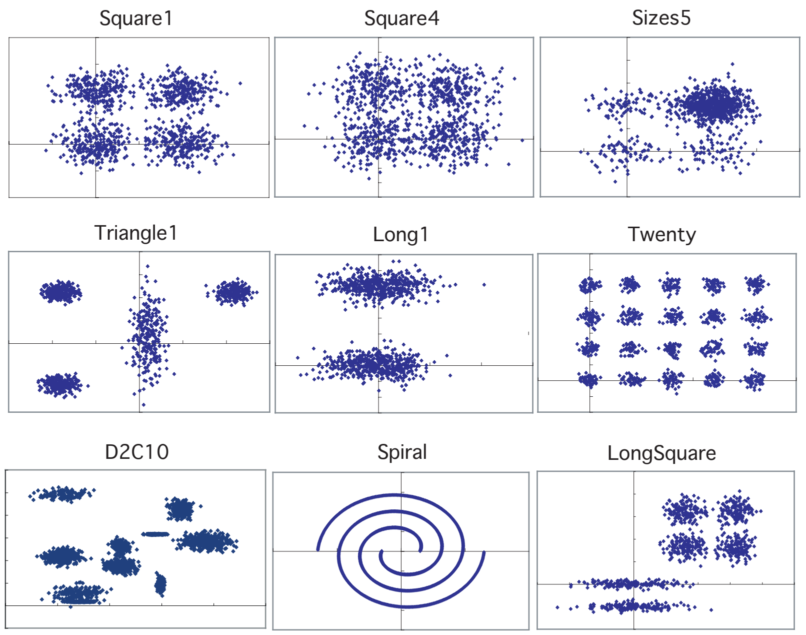


Figure 10: Test dataset overview

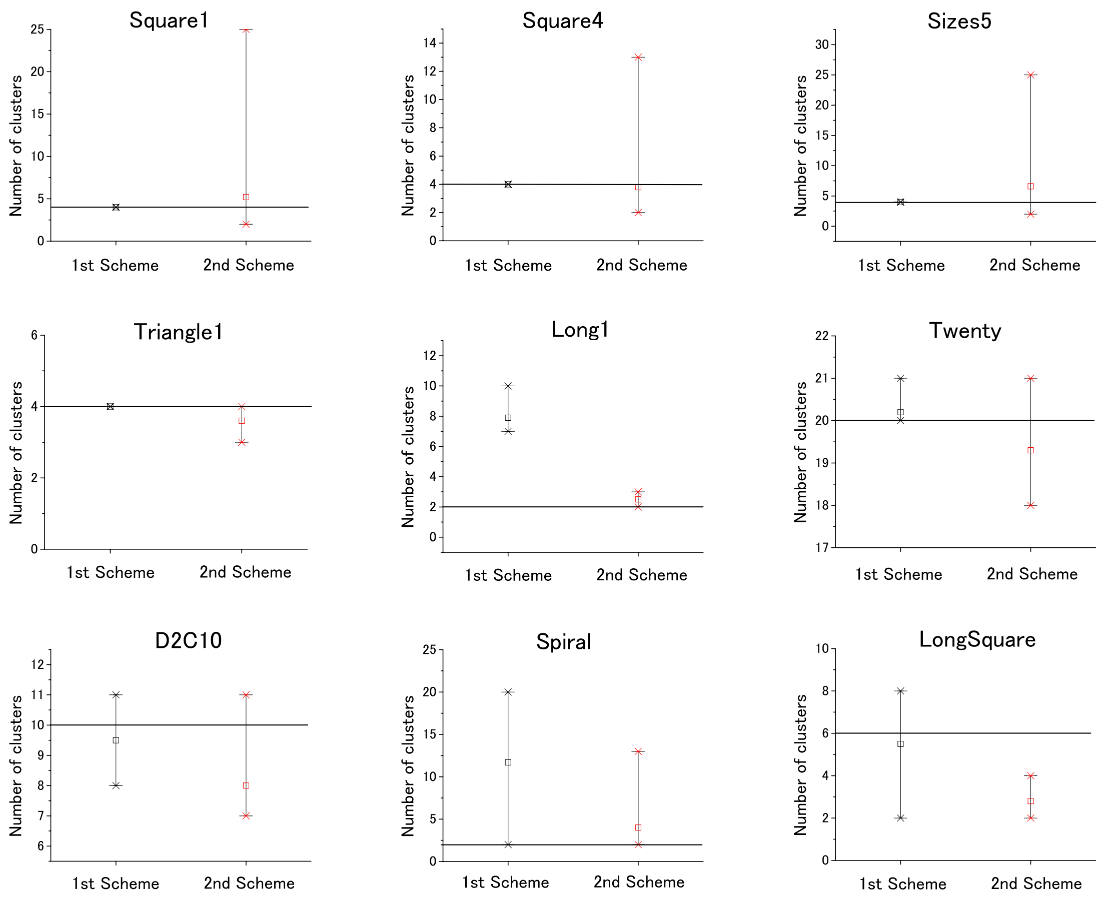


Figure 11: Selected k in each dataset

In the second experiment, we investigated the scalability of our scheme. Figure 12 shows the calculating time of our MOCK. The original MOCK using Gap statistic needs the same time with this total time to determine k, because Gap statistic needs the same calculation (initialization, GA operation etc.) for random data clustering. On the other hand, the k-determination time of our MOCK is a few seconds, and the determined k is collect in almost trials (9 times in 10 trials). Our k-determination scheme doesn't need calculation for random data, so it is much faster than original MOCK.

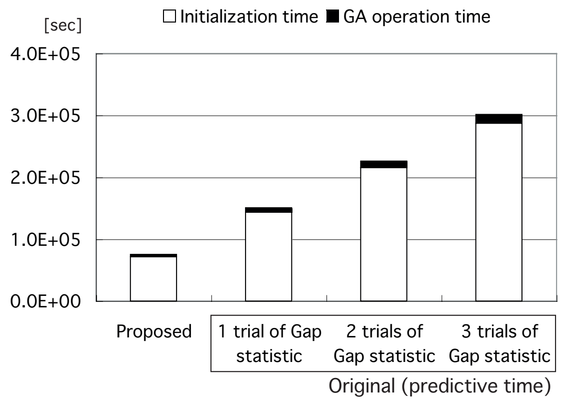


Figure 12: Running time of our MOCK (Square1: 100,000 data)

5. CONCLUSIONS

Data clustering is an important technique used in the preparation step of web mining. Many methods are available to do this, each of which has both merits and demerits. We used a multiobjective clustering algorithm, MOCK, for clustering performance. It has been reported that MOCK is better than k-means and agglutination methods. However, the calculation costs associated with this algorithm are too high for use with large-scale data, such as web access data. Especially, automatic k-determination scheme has high calculation cost. We attempted to resolve this problem.

We proposed a new automatic k-determination scheme of MOCK, which includes two sub-schemes. The first sub-scheme using the information of the Pareto curve is effective but produces fatal errors under some conditions. The second sub-scheme covers this drawback. Our new scheme is able to determine the appropriate k at low cost, although the performance is poorer than the original algorithm. Our scheme can also reduce the Pareto size by about 50-70%, its useful when selected the final solution by expert. In addition, our scheme doesn't need random data clustering, therefore the k-determination is much faster than original. Using this scheme, MOCK can be applied to large-scale data. k-determination scheme is the only different operation from the original MOCK. The Initialization and GA operation of our MOCK are the same with those of the original MOCK. Therefore the quality of Pareto solutions is the same with those of original.

In the future work, the proposed schemes will be applied to realistic web data clustering and behavior mining. The scalable MST creation is also needed to fasten initialization.

APPENDIX

A. REFERENCES

[1] Julia Handl and Joshua Knowles. Multiobjective clustering with automatic determination of the number of clusters, Technical Report No. TR-COMPSYSBIO-2004-02, UMIST, Department of Chemistry, August 2004.

[2] Julia Handl and Joshua Knowles. Evolutionary Multiobjective Clustering, in Xin Yao et al. (editors), Parallel Problem Solving from Nature (PPSN VIII), pp. 1081–1091, Springer-Verlag, Lecture Notes in Computer Science, Vol. 3242, Berlin, September 2004.

[3] A.K.Jain, M.N.Murty, and P.J.Flynn. Data clustering: A review. ACM Computing Surveys, 31:264-323, 1999.

[4] L.Kaufman and P.J.Rousseeuw. Finding Groups in Data. John Wiley & and Sons Inc., New York, NY, 1990.

[5] Ujjiwal Maulik, Sandhamitra Bandypadhyay. Genetic algorithm-based clustering technique. Pattern Recognition 33, pages 1455-1465, 2000.

[6] Julia Handl and Jushua Knowles. An investigation of representations and operators for evolutionary data clustering with a variable number of clusters. Parallel Problem Solving from Nature, 2006.

[7] R. Tibshirani, G. Walther, and T. Hastie. Estimating the number of clusters in a dataset via the Gap statictic. Technical report, Stanford University, 2000.

[8] R. C. Prim. Shortest connection networks and some generalizations. Bell System Technical Journal, 36:1389-1401, 1957.

[9] J. Branke, K. Deb, H. Dierof, and M. Osswald. Finding knees in multi-objective optimization. Proceedings of the Eighth International Conference on Parallel Problem Solving from Nature, Birmingham, UK, 2004.

[10] Julia Handle, Jusua Knowles. Improvements to the scalability of multiobjective clustering. In Proceedings of the 2005 IEEE Congress on Evolutionary Computation, pages 2372-2379. IEEE Press, Anaheim, CA, 2005.

[11] E. Zitzler and M. Laumanns and L. Thiele. SPEA2: Improving the Performance of the Strength Pareto Evolutionary Algorithm. Tech- nical Report 103, Computer Engineering and Communication Networks Lab(TLK), Swiss Federal Institute of Technology(ETH)Zurich(2001), May 2001.

[12] SPEA2 C++ Package, `http://mikilab.doshisha.ac.jp/dia/research/mop\_ga/archive/index.html`

[13] David W. Corne, Nick R. Jerram, Joshua D, Knowles, Martin J. Oates. PESA-II: Region-based Selection in Evolutionary Multiobjective Optimization. Proceedings of the Genetic and Evolutionary Computation Conference, 2001.

[14] S. Watanabe. Genetic Algorithm Based on neighborhood Crossover for Multi-Objective Optimization. Doctoral thesis, Doshisha University, JPN, 2003.