Data Reduction Method for Categorical Data Clustering

Eréndira Rendón[1], J. Salvador Sánchez[2], Rene A. Garcia[1], Itzel Abundez[1], Citlalih Gutierrez[1], and Eduardo Gasca[1]

[1] Lab. Reconocimiento de Patrones, Instituto Tecnológico de Toluca
Av. Tecnológico s/n, 52140 Metepec (México)
erendon@ittoluca.edu.mx
[2] Dept. Llenguatges i Sistemes Informtics, Universitat Jaume I
Av. Sos Baynat s/n, E-12071 Castell de la Plana (Spain)
sanchez@uji.es

Abstract. Categorical data clustering constitutes an important part of data mining; its relevance has recently drawn attention from several researchers. As a step in data mining, however, clustering encounters the problem of large amount of data to be processed. This article offers a solution for categorical clustering algorithms when working with high volumes of data by means of a method that summarizes the database. This is done using a structure called CM-tree. In order to test our method, the K-Modes and Click clustering algorithms were used with several databases. Experiments demonstrate that the proposed summarization method improves execution time, without losing clustering quality.

Keywords: Categorical Attributes, K-modes Clustering Algorithm, Reduced database.

1 Introduction

Finding a database object homogeneous partition (tuple) constitutes a fundamental data mining task. This task is referred to as clustering, defined by [1] [2] as a method to partition a set of objects in clusters so that objects in the same cluster are more similar between them than objects from other clusters, according to a previously established criterion, and in order to maximize intra-cluster similarity and minimize inter-cluster similarity. Most of the existing clustering algorithms can be classified into two main categories, namely hierarchical(agglomerative or divisive) and partitioning algorithms [1]. Most clustering algorithms in literature must keep the set of data in the main memory, so not every clustering algorithm can be applied directly when the set of data cannot be stored in the memory. This constitutes a frequent problem in data mining applications, which work with high volumes of data. The presence of categorical data is also frequent.There are clustering algorithms [3] [4] [5] that work with

H. Geffner et al. (Eds.): IBERAMIA 2008, LNAI 5290, pp. 143–152, 2008.

large databases and categorical data, like ROCK [6] clustering algorithm, which deals with the size of databases by working with a database random sample. However, the algorithm is highly impacted by size of the sample and randomness. In this paper, we offer a solution that consists in reducing the size of a categorical-type database, therefore it possible to used any clustering algorithm. In our case, we use the clustering algorithm K-Modes and CLICK [8]. Several scalable clustering algorithms have been recently reported [8] [3] [4]. One of them is the K-Modes algorithm [4], which is the version of the K-means algorithm for categorical data. Although K-Modes is a scalable algorithm, some problems still arise from it. The data mining community has carried out efforts to develop quick and scalable clustering algorithms for large databases and to make them work with categorical data; some of them are described as follows: The ROCK [6] algorithm is an adaptation of an agglomerative hierarchical algorithm. The algorithm attempts to maximize a goodness measure that favors merging pairs with a large number of Links. Two objects are called neighbors if their similarity exceeds a certain threshold θ given by the user. The number of Links between two objects is the number of common neighbors. The Rock algorithm selects a random sample from the databases after a clustering algorithm employing Links is applied to the sample. Finally, the resulting clusters are used to assign the remaining objects on the disk to the appropriate clusters. CACTUS [5] is an agglomerative algorithm that uses data summarization to achieve linear scaling in the number of rows. It requires only two scans of the data. The CACTUS algorithm is based on the idea of co-occurrence for attributes-values pairs and consists of three phases: summarization, clustering and validation. The COOLCAT clustering algorithm presented by Barbará in [7] is very similar to the K-means algorithm. Unlike K-means, it includes a step that has the purpose of selecting the most adequate representative clusters. It uses entropy as a measure to compute similarity between objects and, as criterion function, it uses minimization of entropy between clusters. The COOLCAT algorithm depends on a sample and it is a hierarchical algorithm. It assumes features are independent from each other. COOLCAT defines a cluster's entropy and starts with a sample of objects; it uses a heuristic method to identify the number of initial clusters, which are the objects with higher entropy. COOLCAT requires as entrance parameters the number of clusters to be formed and the sample size.The CLICK clustering algorithm [8] finds clustering in categorical datasets using a seach method for K- partite maximal cliques. This paper is organized as follows: section 2 provides definitions used by the proposed method; section 3 contains a brief explanation of the K-Modes algorithm; section 4 has the proposed model to summarize the database. Section 5 offers results obtained and, finally, section 6 contains the analysis of results. This paper is organized as follows: section 2 provides definitions used by the method proposed; section 3 contains a brief explanation of the K-Modes algorithm; section 4 has the model proposed to summarize the database. Section 5 offers results obtained and, finally, section 6 contains the analysis of results.

2 Definitions

This section defines concepts that will be used in the rest of this paper. Let D be a database described by a set of records (Tuples), where each tuple $t : t \in D_1 \times \ldots \times D_n$ and a tuple t is represented by $t.A_1, t.A_2, \ldots, t.A_m$; each feature $t.A_i$ takes values in domain D. In clustering, data may be of different types, numeric and non-numeric; non-numeric are frequently referred to as categorical. The first use continuous values and, with categorical data, their domain D_i takes values in a finite, disarranged set; therefore, a feature $t.A_i$ only takes one value from D . Like in [9] an object is a logical conjunction of events. An event is a pair relating features with a value, as follows:

$[t.A_1 = x_1] \wedge \ldots \wedge [t.A_2 = x_2] \wedge \ldots \wedge [t.A_m = x_m]$

Where $x_j \in D_j$ for $1 \leq j \leq m$. Then, we define a categorical object as a logical conjunction of events as follows:

$X = [t.A_1 = x_1] \wedge \ldots \wedge [t.A_2 = x_2] \wedge \ldots \wedge [t.A_m = x_m]$

For simplicity purposes, an event will be referred to as feature to an event.

2.1 Dissimilarity Measure

The similarity measure used in this paper is described in [4]. Let us take two categorical objects X and Y described in m features. The dissimilarity measure between the objects X and Y is defined by the number of feature-to-feature non-coincidences of objects, formally defined as follows:

$$d_1(X, Y) = \sum_{j=1}^{m} \delta(x_j, y_j) \tag{1}$$

where:

$$\delta(x_j, y_j) = \begin{cases} 1 \; if \; x_j \neq y_j \\ 0 \; if \; x_j = y_j \end{cases} \tag{2}$$

2.2 Frecuency and Mode Vectors

Definition 1. *Let* $C = (t.A_i, t.A_{i+1}, \ldots, t.A_m)$ *be a frecuency vector, where* $i = 1, \ldots, m$*;* $m =$ *number of attributes and* $t.A_i$ *is defined as follows:*
$t.A_i = (x_{i,j}|f_{i,j}| \rightarrow, x_{i,j+1}|f_{i,j+1}| \rightarrow, \ldots, x_{i,d}|f_{i,d}|null)$ *; where* $d = |D_i|$,$x_{i,j}$ *is some value of* A_i*;* $f_{i,j}$ *is the relative frecuency of* $x_{i,j}$ *and* $\rightarrow$ *is a link to* $x_{i,j+1}$

Definition 2. *Let* $\tilde{FC} = \left(\tilde{f}_j.A_1, \tilde{f}_j.A_2, \ldots, \tilde{f}_j.A_m\right)$ *be a mode vector from* C*, where* $\tilde{f}_i.A_i$ *is* $x_{i,j}$ *value from* A_i *with the highest relative frecuency, which we calculated as follows:* $\tilde{f}_j.Ai = Max\,[x_{i,j}|f_{i,j}| \rightarrow, x_{i,j+1}|f_{i,j+2}| \rightarrow, \ldots, x_{i,d}|f_d|Null]$.

2.3 *CM* Structure

A CM structure is a pair for summarizing the information that we maintain about a subcluster of objects.

Definition 3. *Given N $m-Dimensional$ categorical objects in a cluster:$[N_i]$ where $i = 1, 2, \ldots, N$, the structure CM entry of the cluster is defined as a pair $CM = \left(C, \tilde{FC}\right)$, where C is a frecuency vector and $\tilde{FC}$ is a mode vector from C.*

2.4 Definition of $CM - Tree$

A $CM - tree$ is a tree balanced by height (see Fig. 1); it is formed by two types of nodes: non-leaf nodes and leaf nodes. These nodes have, in turn, a number of entries, B for the non-leaf nodes and L for leaf nodes, which determine the tree expansion factor, and an absorption threshold T that should meet entries of leaf nodes. Leaf and non-leaf node entries store CM's. At a certain point of the tree construction, leaf nodes contain vector $\tilde{FC}$ representing the cluster of objects that have been absorbed in this entry. Non-leaf nodes store CMs formed by CMs of children nodes (leaf or non-leaf nodes). The construction algorithm is described hereinbelow. Each entry of a leaf and non-leaf node (see Fig.3) stores CMs

2.5 Calculation of Threshold T

To compute T, a heuristic function is proposed to go from T_i to T_{i+1} , which depends on the number of rebuilding and features:

$$T_{i+1} = \frac{m - dec}{m} \tag{3}$$

where m is the number of features, dec is a constant taking values between 0 and m, which may be increased in 1 at each rebuilding to obtain a more compact $CM - tree$.

3 K-Modes Algorithm

The K-Modes [3] algorithm is an extension of the K- means algorithm for categorical data.

General description: The K-Modes algorithm was designed to group large sets of categorical data and its purpose is to obtain K-modes representing the data set and minimizing the criterion function. Three modifications are carried out in the K-modes algorithm with respect to the K-means algorithm:

1. It uses different dissimilarity measures,
2. It substitutes K means by K modes to form centers and
3. It uses a method based in frequencies of attribute values to update modes.

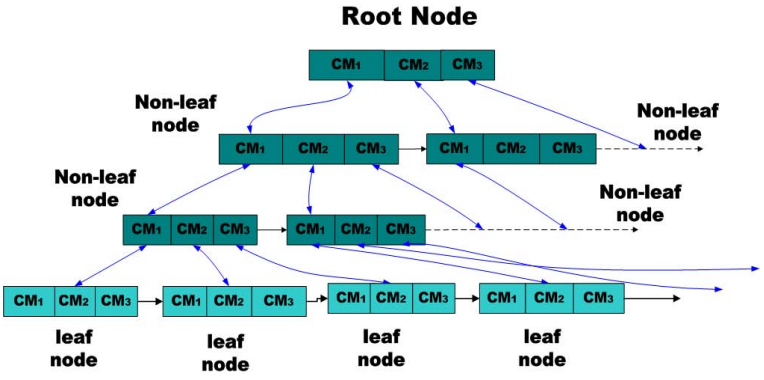

Fig. 1. CM-Tree with L=3 and B=3

Mode update is carried out at each allotment of an object to its set. The K-Modes algorithm produces local optimum solutions depending on initial modes and the objects' entrance order.

4 Proposed Method

The method proposed to reduce the database consists in building a CM-tree that allows obtaining a database summarization.

4.1 Method Overview

Figure 2 shows the details of the summary method. Assume that we know values L, B, T_i and M bytes of memory available for building the $CM-tree$. Summary starts with a $CM-tree$ of a small initial threshold value, say 1, scans the data and inserts objects into the $CM-tree\ t_i$ using the insertion algorithm described in Section 4.3. If it runs out of memory before it finishes scanning the data, it then increases the threshold value (T_i) and rebuilds a new $CM-tree\ t_{i+1}$ from $CM-tree\ t_i$ and T_{i+1}, using the rebuilding process described in Section 4.2. When all objects in the database have been scanned, it has the reduced database into the entry leaves ($\tilde{FC}$) of the last formed $CM-tree$.

4.2 Rebuilding Process

Assume t_i is a $CM-tree$ of threshold T_i. Its height is h and its size (number of nodes) is S_i. Given $T_{i+1} \geq T_i$, we use all the leaf node entries of $CM-tree\ t_i$ to rebuild a $CM-tree\ t_{i+1}$ of threshold T_{i+1} such that the size of t_{i+1} should not be larger than S_i. The building process obtains a $CM-tree\ t_{i+1}$ from $CM-tree$ t_i using the CM structure from entry leaf nodes of $CM-tree\ t_{i+1}$

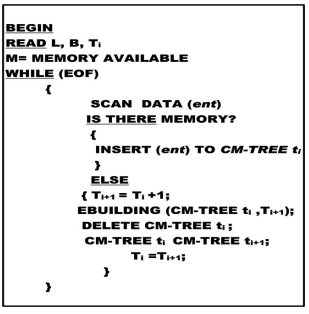

Fig. 2. Overview of reduced method

4.3 Insertion Algorithm

The insertion algorithm offered here is very similar to the one presented in [10]; the main difference is node entries.

1. Identifying the appropriate leaf: starting from the root, it recursively descends the CM-tree by choosing the closest child node according to the similarity metric given in Sect. 2.1.
2. Updating the leaf: when a leaf node is reached, it finds the closest leaf entry, say L_i, and then tests whether L_i can absorb *ent* without violating the threshold condition. If so, the new entry is absorbed by L_i. If not, it tests whether there is space for this new entry on the leaf. If so, a new entry for *ent* is added to the leaf. Otherwise, the leaf node must be split. Splitting is done by choosing the farthest pair of entries as seeds and redistributing the remaining entries based on the closest criteria.
3. Updating the path to the leaf: after inserting *ent* into a leaf, the information for each non-leaf entry must be updated on the path from the root to the leaf. If no splitting was done, this simply involves the leaf entry that absorbed *ent*. Conversely, a leaf split means that a new non-leaf entry has to be inserted into the parent. If the parent has space for this new entry, at all higher levels, it is only necessary to update the corresponding entries to reflect the addition of *ent*. However, it may be required to split the parent as well, and so on up to the root. If the root is split, the tree height increases by one.

5 Experimental Evaluation

This section offers the results of applying our database reduction method, for which we selected the K-Modes and Click clustering algorithms. Experiments were carried out to test that, when reducing the database, clustering quality is not impacted and execution time is improved. The main task of clustering algorithms is to discover the grouping structures inherent in data. For this purpose, the assumption that a certain structure may exist in a given database is first made and, then, a clustering algorithm is used to verify the assumption

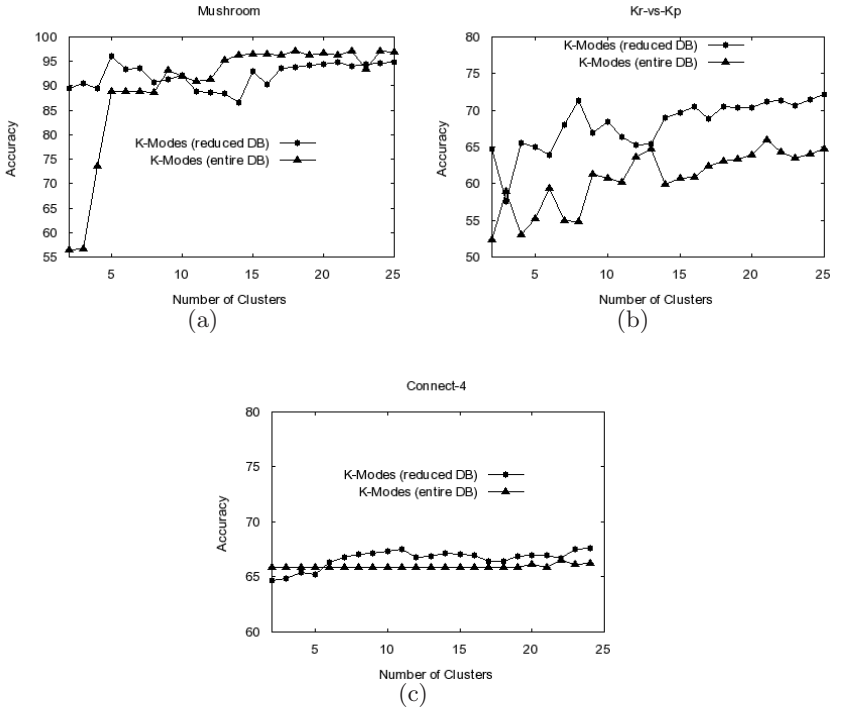

Fig. 3. Accuracy vs Number of clusters (K-Modes)

and recover the structure. [2] presents three types of criteria used to evaluate the performance of clustering algorithms. In evaluating our database reduced method, we chose an external criterion, which measures the degree of correspondence between the clusters obtained from clustering algorithms and the classes assigned a priori. We have used the clustering accurancy as a measure of a clustering result. Clustering algorithms were run in two ways: first, we ran them throughout the entire database; the clustering results obtained are called *Entire DB*; when we ran algorithms throughout the reduced database (using our method), the clustering results are called *Reduced DB*.

5.1 Set of Data Used

Our experiments were carried out with data sets taken from UCI achina Learning Repository: http://www.ics.uci.edu/ mlearn/MLRepository.html. The data sets used were Kr-vs-Kp with 3198 objects, Connect-4 with 67557 and Mushroom with 8124, all of them with categorical data.

5.2 Tests

To measure the results of our model, the k-Modes and Click clustering algorithms were both executed with the reduced and the entire database for different values

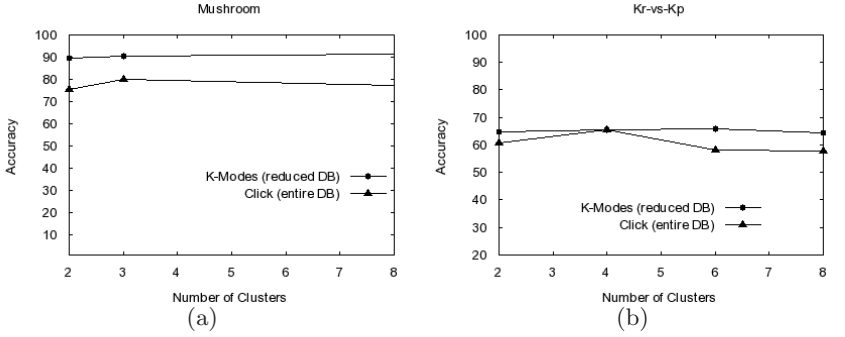

Fig. 4. Accuracy vs Number of clusters (Click)

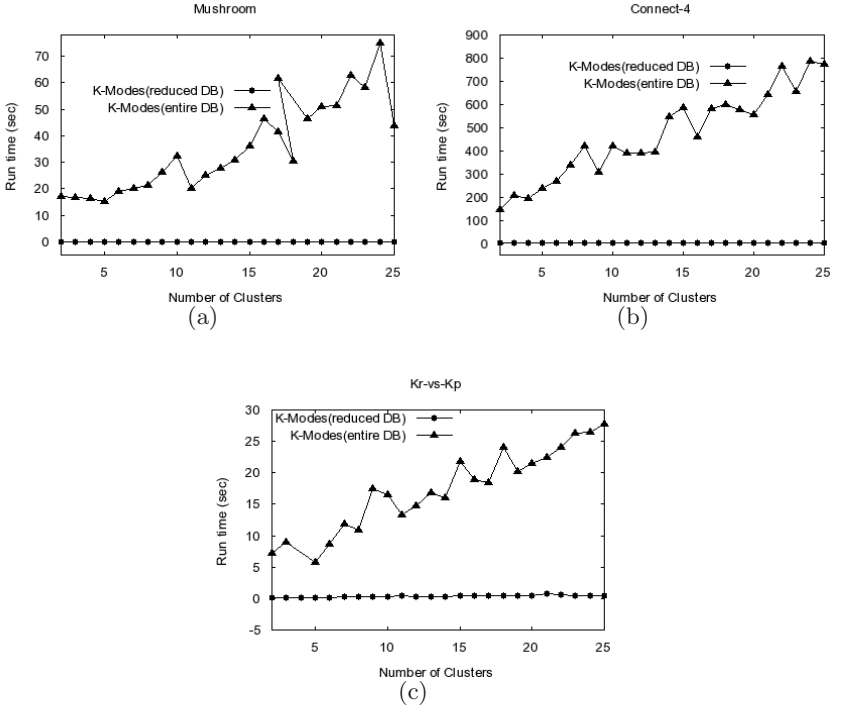

Fig. 5. Run time vs Number of clusters (K-Modes)

of K. The execution time and clustering quality (accuracy) were compared, as shown in Figs. 3 and 5.

5.3 Discussion

– About the K-Modes Algorithm: Figure 3.a presents both clustering results, with the mushroom entire and reduced database for different values of K.

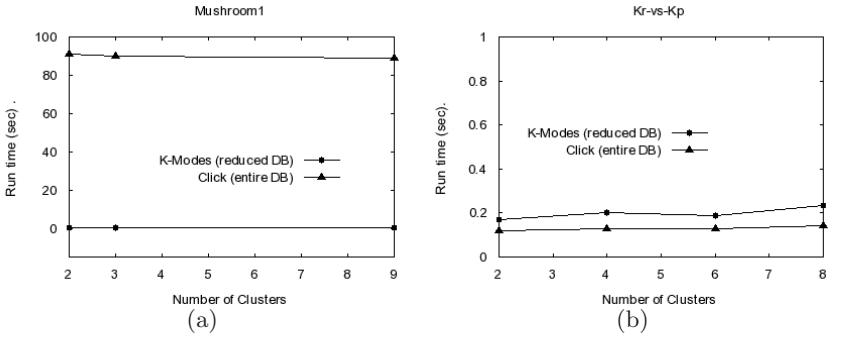

Fig. 6. Run time vs Number of clusters (Click)

On the Reduced DB clustering results, we can see that accuracy was better when K took values between 2-10; however, when K took values between 11 to 25, both clustering results were almost similar. Figure 3.b shows clustering results of the Kr-vs-Kp database, with the Reduced DB clustering results accuracy being slightly better than with the Entire DB clustering results. Figure 3.c shows the Entire DB and Reduced DB clustering results of the Connect-4 database. In this case, accuracy reported on the Reduced DB clustering results was better than with the Entire DB clustering results when K took values between 6 to 23. Figure 5 presents execution times obtained by means of the K-Modes algorithm for both the entire and reduced database. When the K-Modes algorithm was run with the reduced database, better run times were obtained than with the K-modes algorithm and the entire database. We can therefore state that the proposed database reduction method considerably improves the efficiency of the K-Modes algorithm due to the reduction of execution times and because the clustering quality is not affected by this reduction.

– About the Click Algorithm: Figure 4 presents the results obtained by the Click and K-Modes algorithms. The first was run with the reduced DB and the second with the entire DB, using different values of K. For the mushroom dataset, accuracy obtained with K-Modes (Reduced DB) was better than with the Click algorithm (Entire DB). Results obtained with the Kr-vs-Kp dataset were similar (see Figure 4a and Figure 4b). Figure 6 shows the time run, evidencing its decrease when using the reduced DB.

6 Conclusions and Future Works

We offer a solution for categorical data clustering algorithms considering the size of data sets. This solution consists in a database reduction method using a tree-type structure balanced by the height of the CM-tree; it adjusts to the memory availability and, as it reduces the data set, it also carries out semi-clustering. When reducing the data set, tests demonstrated that clustering quality is not

affected and execution time decreases considerably. We consider applying this method to other clustering algorithms in the future.

References

1. Kaufman, L., Rousseeuw, P.J.: Finding Groups in Data: An Introduction to Cluster Analysis. John Wiley and Sons, New York (1990)
2. Jain, A.K., Dubes, R.C.: Algorithm for Clustering Data. Prentice-Hall, Englewood Cliffs (1988)
3. Andritsos, P., Tsaparas, P., Miller, R.J., Sevcik, K.C.: LIMBO: A scalable Algorithm to Cluster Categorical Data. Technical report, University of Toronto, Department of Computer Science, CSRG-467 (2004)
4. Huang, Z.: A Fast Clustering Algorithm to Cluster Very Large Categorical Data Sets in Data Mining. In: Sigmod Workshop on Research Issues on Data Mining and Knowledge Discovery, pp. 1–8 (1997)
5. Ganti, V., Gehrkeand, J., Ramakrishanan, R.: CACTUS-Clustering Categorical Data Using Summaries. In: Proceeding of the 5th ACM Sigmod International Conference on Knowledge Discovery in Databases, San Diego, California, pp. 73–83 (1999)
6. Guha, S., Rastogi, R., Shim, K.: Rock: A robust clustering algorithm for categorical attributes. In: Proceeding of the 15th International Conference on Data Engineering (ICDE), Sydney, pp. 512–521 (1999)
7. Barbará, D., Li, Y., Couto, J.: Coolcat: an entropy-based algorithm for categorical clustering, pp. 582–589. ACM Press, New York (2002)
8. Zaki, M.J., Peters, M., Assent, I., Seidl, T.: CLICK: An Effective algorithm for Mining Subspace Clustering in categorical datasets. In: Proceeding of the eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, pp. 733–742 (2005)
9. Gowda, K., Diday, E.: Symbolic Clustering Using a New Dissimilarity Measure. Pattern Recognition 24(6), 567–578 (1991)
10. Rendón, E., Sánchez, J.S.: Clustering Based on Compressed Data for Categorical and Mixed Attibutes. In: Yeung, D.-Y., Kwok, J.T., Fred, A., Roli, F., de Ridder, D. (eds.) SSPR 2006 and SPR 2006. LNCS, vol. 4109, pp. 817–825. Springer, Heidelberg (2006)