The BANG-clustering system: Grid-based data analysis

Article *in* Lecture Notes in Computer Science · June 1997

DOI: 10.1007/BFb0052867 · Source: CiteSeer

2 authors, including:

Erich Schikuta
University of Vienna
257 PUBLICATIONS **1,971** CITATIONS

The BANG-Clustering System: Grid-Based Data Analysis

Erich Schikuta and Martin Erhart

Institute of Applied Computer Science and Information Systems, University of Vienna
Rathausstr. 19/4, A-1010 Vienna, AUSTRIA
Email: schiki@ifs.univie.ac.at

Abstract. For the analysis of large images the clustering of the data set is a common technique to identify correlation characteristics of the underlying value space. In this paper a new approach to hierarchical clustering of very large data sets is presented. The BANG-Clustering system presented in this paper is a novel approach to hierarchical data analysis. It is based on the BANG-Clustering method ([Sch96]) and uses a multi-dimensional grid data structure to organize the value space surrounding the pattern values. The patterns are grouped into blocks and clustered with respect to the blocks by a topological neighbor search algorithm.

1 Introduction

Clustering methods are extremely important for explorative data analysis, which is an important approach for the analysis of images. Previously presented algorithms can be divided into hierarchical algorithms, e.g. single-linkage, complete-linkage, etc. and partitional algorithms, e.g. K-MEANS, ISODATA, etc. (see [DJ80]). All of these methods suffer from specific drawbacks, when handling large numbers of patterns. The hierarchical methods provide structural information, as dendrograms, but are suitable only for a small number of patterns. With growing numbers the computational expense increases, due to the calculation of a dissimilarity matrix, where each pattern is compared to all others. The partitional methods are to some less resource consuming. However they lack methodical freedom because of the prerequisite of a "good guess" of structural information, e.g. the numbers and the positions of the initial cluster centers. If the choice of the initial clustering is not appropriate, the partitional methods also become very calculation intensive in computing new cluster centers.

A number of different algorithms have been proposed to overcome these problems (e.g. [Bru88, CCM92, CD91, Kur91, IK89, VR92, ZWB91]). Most of the algorithms compare the patterns to each other or to predefined cluster center. Via a calculated distance metric they organize the patterns by combining them into clusters. Another alternative is to cluster the patterns according to the structure of the embedding space, as first presented by Warnekar et al. [WK79]. They proposed a heuristic, hierarchical clustering algorithm based on detecting clusters by overlapping pattern cells. As Warnekar et al. pointed out, however, the algorithm suffered from the problem that all pattern cells were of the same

size and so did not adapt to the real distribution of the patterns. This leads to a complex and quite costly algorithm, which decides how to combine the cells by an expensive distance calculation between possible neighbor cells. Broder [Bro90] solved this problem by means of using of an adaptable data structure, the kd-B-tree. His algorithm calculates the m nearest neighbors to a specified pattern in a k-dimensional value space. The algorithm is problematic for cluster analysis, because it needs a given cluster center as input, additionally the performance is low due to the necessary recursive traversal of the kd-B-tree index.

The hierarchical Grid-Clustering algorithm [Sch96] combines and refines both ideas. It introduces a density index to compare and evaluate the possible pattern cells to find cluster centers and to combine neighbor cells. Further, a new Grid-Structure to organize the value space surrounding the patterns was implemented, which overcomes the problems of the kd-B-tree. This algorithm achieves an appealing performance gain in comparison to all conventional algorithms.

2 BANG-Clustering

Conventional cluster algorithms calculate a distance based on a dissimilarity metric (e.g. Euclidean distance, etc.) between patterns or cluster centers. The patterns are clustered accordingly to the resulting dissimilarity index. The BANG-Clustering algorithm presented here uses the idea of Warnekar [WK79] to organize the value space containing the patterns. For that reason we use the BANG-Structure. The patterns are treated as points in a k-dimensional value space and are randomly inserted into the BANG-Structure. These points are stored accordingly to their pattern values, preserving the topological distribution. The BANG-Structure partitions the value space (shown in figure 1) and administrates the points by a set of surrounding rectangular shaped *blocks* (figure 2).

A block is a rectangular shaped cube containing up to a maximum of p_{max} patterns. X = $(x_1, x_2, ..., x_n)$ is a set of n patterns and x_i is a pattern consisting of a tuple of k describing features $(p_{i_1}, p_{i_2}, ..., p_{i_k})$, where k is the number of dimensions of the underlying value space of the data set.

The pictures shown in the figures are actual screen shots. In the example 50000 patterns, which are clustered into 3 groups with about 20 percent noise, were analyzed by the BANG-Clustering algorithm.

The BANG-Clustering algorithm uses the block information of the BANG-Structure and clusters the patterns accordingly to their surrounding blocks creating a respective dendrogram in turn (see figure 8).

2.1 The BANG-Structure

The BANG-Structure stores the patterns of the underlying value space by a grid structure, which is called *grid directory* (similarly to the Grid-File). This structure (see figure 3), which is administrated by *scales*, partitionates the k-dimensional value space into *grid regions* (rectangular shaped subspaces). Each

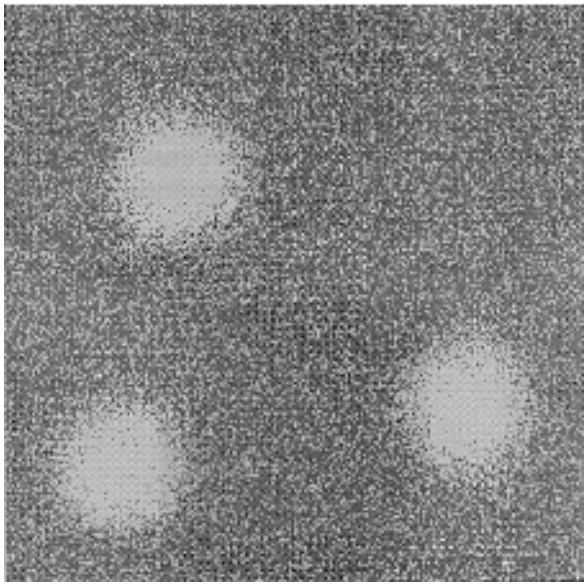

Fig. 1. 2-dimensional pattern set

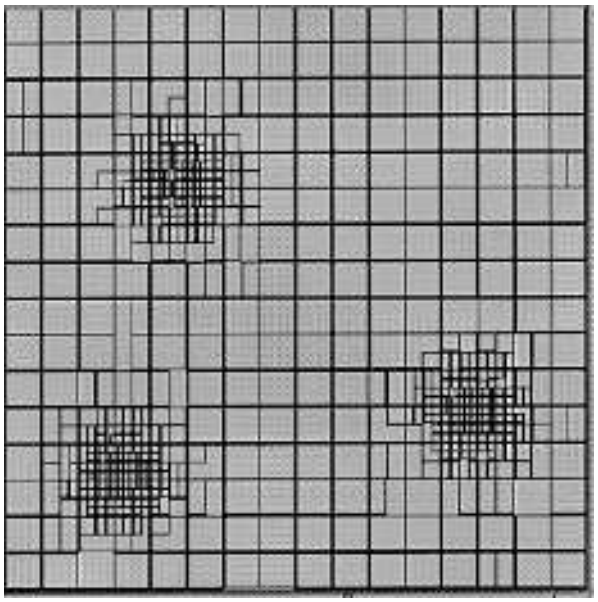

Fig. 2. BANG-Clustering

scale represents one pattern attribute, and each scale entry resembles a (k-1) dimensional hyperspace partitioning the value space into two.

Each grid region is mapped to one *data block* containing the patterns, but a data block can be mapped by more than one grid region (1:m mapping). The union of these grid regions (mapping to the same data block) is called *block region*. The value space spanned by a block region is rectangular shaped (convex).

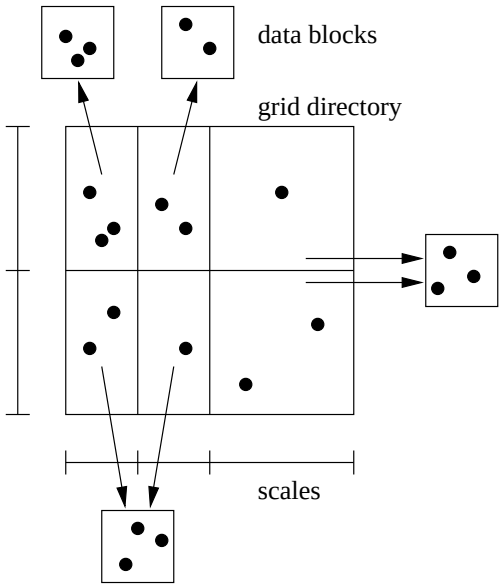

Fig. 3. BANG-Structure

The value space is partitioned into a *hierarchical* set of grid regions. Each region is uniquely identified by a pair of keys (r, l), where r is the region number and l is the level number. The partitioning is binary (a region is split into two equally shaped regions) in each dimension. The sequence of the split dimensions has to be uniquely defined. Region (0,0) comprises the whole value space and is partitioned according the defined scheme into subregions.

The structure of the block regions is defined by the following two axioms [Fre87]

- The union of all subregions into which the value space has been partitioned must span the whole value space.
- If two subregions intersect, then one of these subregions completely encloses the other.

The second axiom allows nested regions, which is shown in figure 4. To reach compact structures algorithms are defined [Fre87], which guarantee a balance between the data blocks by redistribution. This proved extremely useful for clustered value sets.

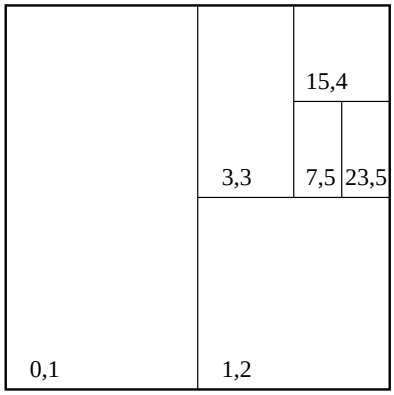

Fig. 4. BANG-Structure block regions

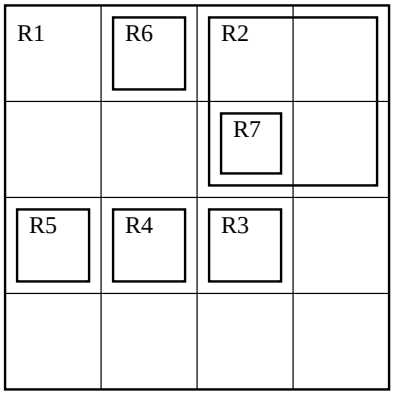

Fig. 5. Grid directory - Neighborhood

The BANG-Clustering algorithm uses the block information of the grid directory and clusters the patterns to their blocks accordingly.

2.2 Density Index

The algorithm calculates a *density index* of each block via the numbers of patterns and the spatial volume of the block (analogously to the Grid-Clustering algorithm [Sch96]). The spatial volume V_B of a block B is the Cartesian product of the extents e of block B in each dimension, i.e.

$$V_B = \prod e_{B_i}, \\ i = 1, ..., k.$$

The density index D_B of block B is defined as the ratio of the actual number of patterns p_B contained in block B to the spatial volume V_B of B, i.e.

$$D_B = \frac{p_B}{V_B}$$

The blocks are sorted accordingly to their density indices. Blocks with the highest density index (obviously with highest pattern correlation) become clustering centers. The remaining blocks are then clustered iteratively in order of their density index, thereby building new cluster centers or merging with existing clusters. Only blocks adjacent to a cluster, i.e. *neighbors*, can be merged.

2.3 Neighbors

Two types of neighborhood can be distinguished in the BANG-Structure, *normal neighborhood*, i.e. neighbors respective block regions, and *refined neighborhood*, i.e. neighbors respective logical regions. Further a *neighbor degree* can be defined by the dimensionality of the "touching" area between 2 regions. Generally the dimensionality can vary between 0 (an point) and k-1 (a k-1 dimensional hyperplane). For the example shown in figure 5 (2-dimensional case) the level of dimensionality is 0 (a point) and 1 (an edge). A normal neighborhood exists e.g. between regions R2 and R1, R6, and R7, and a refined neighborhood between regions R2 and R1, R6, and R7.

Neighbors are found by comparison of the scale values of the grid directory. If regions are at the same level, the differences can be determined directly. If the levels are not at the same level, the lower level region has to be transformed to the higher level region and the comparison has to be done appropriately. In the example of figure 6 the regions and their identifiers are R1 = (0,0), R2 = (3,2), R3 = (9,4), R4 = (12,4), R5 = (8,4), R6 = (14,4), and R7 = (3,4). R6 and R2 are neighbors but on different region-levels (R6 on level 4, with x =1 and y = 3, and R2 on level 2, with x =1 and y = 1). Therefore we have to transform R2 to level 4, which yields in $x_{min} = 2$, $x_{max} = 3$, and $y_{min} = 2$, $y_{max} = 3$.

The region identifier are an ordered set of tuples. To find possible neighbor regions the algorithm accesses these tuples. To support this step efficiently we designed a novel administration structure for the region identifiers. Because of the numbering scheme of the grid regions we chose a binary tree for storing the grid structure. The basic scheme is shown in figure 6.

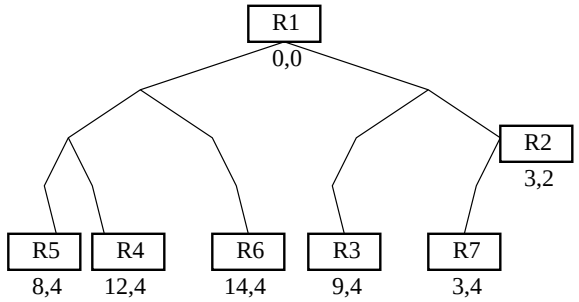

Fig. 6. Binary tree storing the grid directory of figure 5

The partitioning of a region is directly supported by the tree scheme, and a comprising region is simply found by backtracking the path to the root (representing the whole value space). The height of the tree is defined by the number of region levels.

2.4 Dendrogram

The dendrogram is calculated directly by the clustering algorithm, as depicted in figure 7. The density indices of all regions are calculated and sorted in decreasing order.

Starting with the first region (with the highest density index) all neighbor regions are determined and classified in decreasing order (step 1). The neighbor search is repeated for each processed region. The found regions are placed in the dendrogram to the right of the original regions (step 2), respective to the following rules,

- is R1 neighbor of R2 and R2 neighbor of R3 and R1 > R2 > R3, then build with R1, R2, and R3 a cluster (neighbor search starting from R3), and
- is R1 neighbor of R2 and R2 neighbor of R3 and R1 > R2 < R3, then build with R1, R2, and R3 a cluster (neighbor search starting from R2).

A calculated dendrogram based on the example depicted by figure 2 is shown in figure 8.

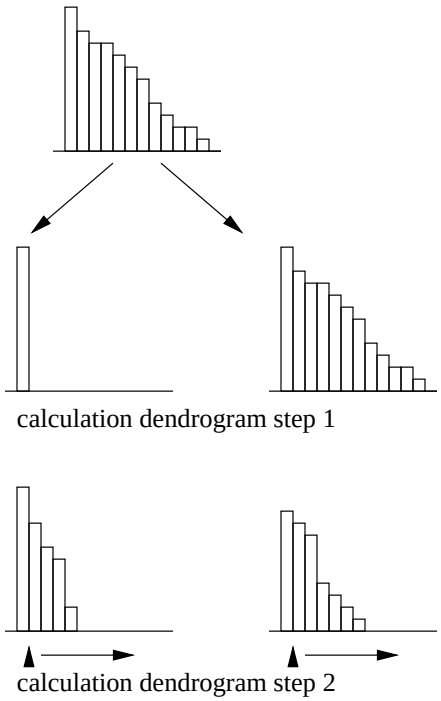

Fig. 7. Dendrogram creation

Fig. 8. Calculated dendrogram for example of figures 1 and 2

3 The BANG-Clustering System

The BANG-Clustering algorithm was implemented under the Unix operating systems using X11 and the Motif libraries providing a user friendly environment [Erh95]. Running versions are available for Linux, Sun and HP workstations. Basically the system provides windows for

- the control of the program and the visual analysis of the data set,
- the graphical representation of the BANG structure and the state of the clustering process,
- the number and positions of detected cluster centers, and
- the layout of the dendrogram.

The system allows to control any parameter of the clustering process and to follow iteratively each clustering step. The design of the system tries to maximize the user's flexibility for the analysis of the data set. In figure 9 a screen shot of the BANG-system is shown, with a sample of the most common windows. The control, clustering, dendrogram, and data view window are easily identifiable in the figure.

In figure 10 the iterative clustering approach is shown for a 2-dimensional data set consisting of 10000 patterns. The user has the possibility to control the clustering process at his will (he can advance or retrace in the process), by positioning the scroll bar at the bottom of the main window showing the pattern lay-out. For higher dimensional data sets the projection dimensions can be chosen appropriately. In figure 10 three clustering states are depicted, for 0%, 50%, and 100% of the data set clustered. Interdependently the window showing the BANG-file structure shadows the clustered block regions and blacks out the last clustered region (the region with the lowest density index until now). Thus, if the whole data set is clustered, all but one block regions are shadowed and the last region is blacked out.

Accordingly to the clustering process also the dendrogram window is updated showing the clustered data patterns in black and the remaining patterns in grey. In figure 11 the situation is shown for 50% of the data set clustered.

4 Performance Analysis

For our analysis we compare the BANG-Clustering to several well known conventional clustering algorithms and to the Grid-Clustering [Sch96] as well. For the conventional algorithms we used commercial statistical packages on workstations (WS) and, due to memory exhaustion and exceeding execution times in the workstation environment, also on mainframes (MF). The times shown in the charts are pure processing times for the completion of the algorithms (no data set loading or display of information). The investigated algorithms in figure 12 were BANG-Clustering (workstation), GRID-Clustering (workstation), single linkage (mainframe), and quick cluster (workstation).

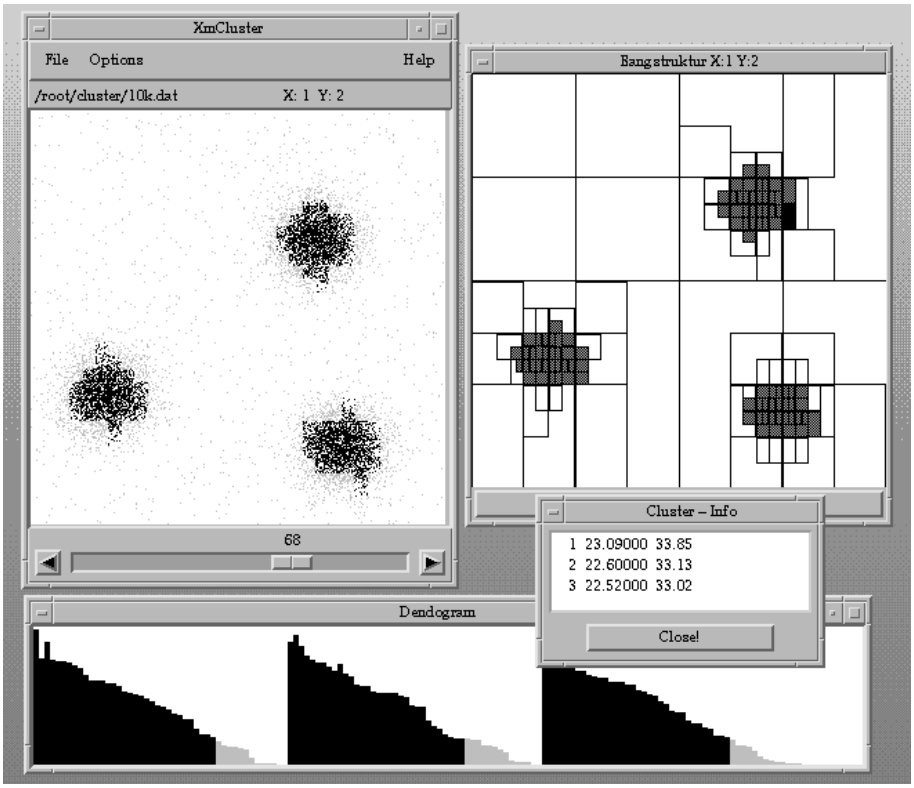

Fig. 9. BANG system screen layout

BANG-Clustering and Grid-Clustering outperformed the conventional methods by far. Both performed even better on workstations than any of the conventional methods on mainframes. More importantly they delivered results where the other algorithms failed because of exceeding runtime behavior or memory exhaustion.

Further we compared BANG-Clustering to Grid-Clustering directly. The block size (i.e. the number of patterns per region) is an influencing factor for grid structures, as shown in [Sch96]. Therefore we measured the execution times for constant block size (100 patterns) and for dynamic block sizes (10% of the data set). Due to the situation that both grid methods allow very large data sets, we present the execution times for data set sizes up to 100000 patterns (see figure 13).

For large data sets BANG-Clustering outperformed Grid-Clustering due to the linear growth rate of the BANG-structure size (one logical region per data block) compared to the over-linear growth of the Grid-structure size (often many

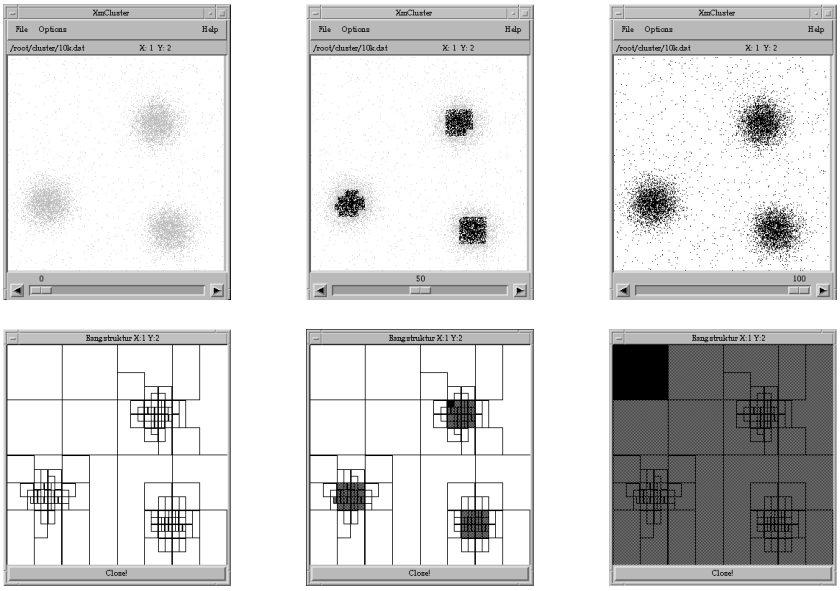

Fig. 10. Iterative data clustering process

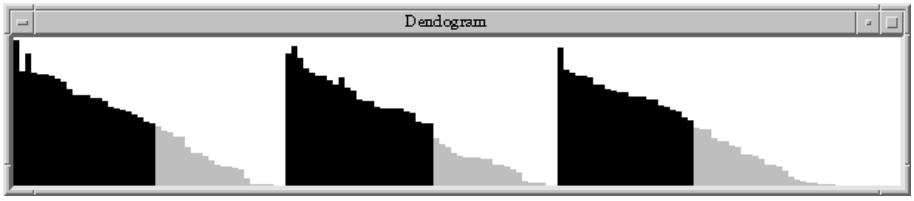

Fig. 11. Dendrogram screen

regions per data block). BANG-Clustering is even capable to cluster data sets of up to 1 million patterns, which is beyond the possibilities of Grid-Clustering. The only restriction are the memory requirements for storing such huge data sets. Test showed that for 1 million 2-dimensional patterns (2 double values) a BANG-Structure of 48 MByte was created.

5 Summary

We presented a novel hierarchical clustering method, BANG-Clustering, which organizes the space comprising the pattern set. This method is an extension of the Grid-Clustering algorithm presented in [Sch96], and is capable to cluster even larger pattern sets more efficiently. It outperforms all conventional hierarchical and partitional algorithms and the Grid-Clustering method as well.

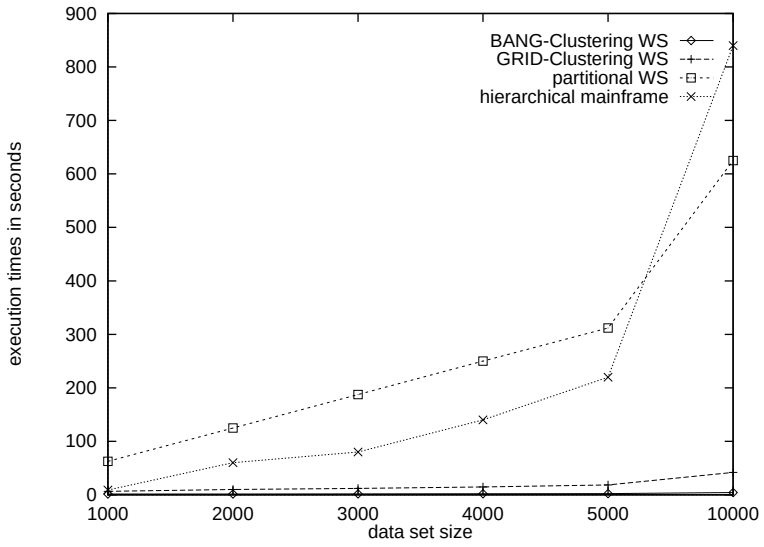

Fig. 12. Runtime comparison of different clustering algorithms

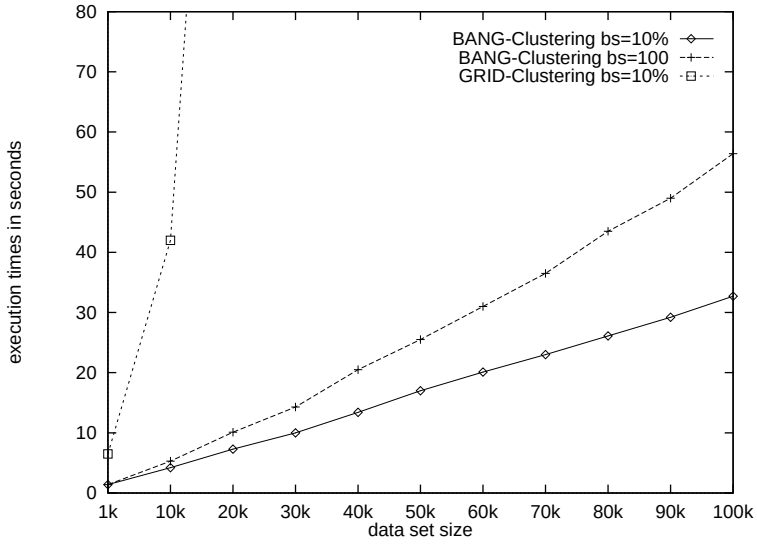

Fig. 13. Runtime comparison of BANG-Clustering for different block sizes (bs)

BANG-Clustering is therefore capable to analyze data sets, which were previously not tractable due to their size and/or dimensionality without any supplemental input information.

References

[Bro90] A.J. Broder. Strategies for efficient incremental nearest neighbor search. *Pattern Recognition*, 23:171–178, 1990.

[Bru88] M. Bruynooghe. A very efficient strategy for very large data sets clustering. In *Proc. 9th Int. Conf. on Pattern Recognition*, pages 623–627. IEEE Computer Society, 1988.

[CCM92] D. Chaudhuri, B.B. Chaudhuri, and C.A. Murthy. A new split-and-merge clustering technique. *Pattern Recognition Letters*, 3:399–409, 1992.

[CD91] Gowda K. Chidananda and E. Diday. Symbolic clustering using a new dissimilarity measure. *Pattern Recognition*, 24:567–578, 1991.

[DJ80] R. Dubes and A.K. Jain. *Clustering methodologies in exploratory data analysis*, volume 19, pages 113–228. Academia Press, 1980.

[Erh95] Martin Erhart. Entwurf und Implementation eines BANG-File-basierten Clusteranalyseverfahrens. Master's thesis, University of Vienna, September 1995.

[Fre87] M.W. Freestone. The bang file: A new kind of grid file. In *Proc. Special Interest Group on Management of Data*, pages 260–269. ACM, May 1987.

[IK89] M.A. Ismail and M.S. Kamel. Multidimensional data clustering utilizing hybrid search strategies. *Pattern Recognition*, 22:75–89, 1989.

[Kur91] T. Kurita. An efficient agglomerative clustering algorithm using a heap. *Pattern Recognition*, 24:205–209, 1991.

[Sch96] E. Schikuta. Grid clustering: An efficient hierarchical clustering method for very large data sets. In *Proc. 13th Int. Conf. on Pattern Recognition*, volume 2, pages 101–105. IEEE Computer Society, 1996.

[VR92] N.B. Venkateswarlu and P.S.V.S.K. Raju. Fast isodata clustering algorithms. *Pattern Recognition*, 25:335–342, 1992.

[WK79] C.S. Warnekar and G. Krishna. A heuristic clustering algorithm using union of overlapping pattern-cells. *Pattern Recognition*, 11:85–93, 1979.

[ZWB91] Q. Zhang, Q.R. Wang, and R. Boyle. A clustering algorithm for data-sets with a large number of classes. *Pattern Recognition*, 24:331–340, 1991.

This article was processed using the LaTeX macro package with LLNCS style