INFORMATICA, 2014, Vol. 25, No. 4, 563–580

DOI: http://dx.doi.org/10.15388/Informatica.2014.30

Sweep-Hyperplane Clustering Algorithm Using Dynamic Model

Niko LUKAČ*, Borut ŽALIK, Krista Rizman ŽALIK,
Faculty of Electrical Engineering and Computer Science, University of Maribor
Smetanova ulica 17, SI-2000 Maribor, Slovenia
e-mail: niko.lukac@uni-mb.si

Received: May 2013; accepted: March 2014

Abstract. Clustering is one of the better known unsupervised learning methods with the aim of discovering structures in the data. This paper presents a distance-based Sweep-Hyperplane Clustering Algorithm (SHCA), which uses sweep-hyperplanes to quickly locate each point's approximate nearest neighbourhood. Furthermore, a new distance-based dynamic model that is based on 2^N-tree hierarchical space partitioning, extends SHCA's capability for finding clusters that are not well-separated, with arbitrary shape and density. Experimental results on different synthetic and real multidimensional datasets that are large and noisy demonstrate the effectiveness of the proposed algorithm.

Key words: clustering, sweeping paradigm, dynamic model.

1. Introduction

Clustering is an often used technique in data mining, by which the unknown relations and knowledge contained within the datasets can be obtained. In the N-dimensional Euclidean space R^N, it is a process of finding groups (i.e. clusters) of points. Clustering has been successfully applied in many areas, such as gene research (Jiang *et al*., 2004), image segmentation (Shi and Malik, 2000), document organisation and filtering (Steinbach *et al*., 2000), spatial data analysis (Han *et al*., 2001). Unsupervised clustering also presents an alternative to supervised data classification (Kiriş, 2013), when the relations in the data are unknown and the training set is difficult to obtain. Although, no common definition of a cluster has been agreed upon, it is widely accepted that the points in the considered cluster are more similar (i.e. nearer) to each other than the points in the remaining clusters. Over the past decade, many algorithms and approaches have been contributed for solving the problem of cluster identification. Some clustering methods are specialised for finding globular clusters, whilst other methods find contiguous (i.e. connected) type of clusters. Other methods perform density-based clustering by separating regions of high density from regions with lower densities, and are generally robust to noise and outliers. The notion of noise represents points that do not have any meaning or structure, whilst the

*Corresponding author.

outliers are isolated points that are considerably distant from the rest of the data. Methods designed for finding contiguous clusters can also find globular type of clusters, as well as clusters of arbitrary shape and size, as long as they are well-separated. Generally, distance-based clustering methods designed for finding specific types of clusters, use a static clustering model (Theodoridis and Koutroumbas, 2008), whilst dynamic clustering models (Karypis *et al*., 1999) provide the ability for finding arbitrary (i.e. natural) types of clusters that can have arbitrary sizes, shapes and densities. A dynamic model should also be capable of detecting not well-separated clusters. Over recent years, the sizes of datasets have increased exponentially, where datasets can contain millions of points with the presence of noise and outliers. High performance clustering is required to process such datasets within reasonable time, whilst quality results are still expected. Generally, a trade-off exists between finding arbitrary type of clusters, and processing large datasets within a reasonable time. Most of the known clustering methods are designed around the former ability. Although some approaches consider the compression of larger datasets, or using a small representative subset from the data, it is never guaranteed that all the clusters will be found (Steinbach *et al*., 2003). Distance-based clustering methods attempt to locate the nearest neighbour to a given point, and perform clustering being based on the proximity of the points. Exact nearest neighbour search is infeasible within higher dimensional spaces. In most cases, an approximate nearest neighbourhood (ANN) search is good enough (Nene and Nayar, 1997), because the acquired solution is often exact (Chang *et al*., 2002), and is obtained at several magnitudes faster.

This paper proposes a new distance-based Sweep-Hyperplane Clustering Algorithm (SHCA). The proposed algorithm uses sweeping hyperplanes that provide fast searches of the ANN for each point in the R^N space. The found ANN enables faster local clustering decisions. SHCA has been extended with a newly proposed distance-based dynamic model, based on a 2^N-tree data structure, in order to find arbitrary type of clusters that do not have to be well-separated.

The remainder of this paper is structured in five sections. An overview of the better known clustering methods is given in Section 2. The proposed algorithm and dynamic model are presented in Section 3. The effectiveness of the algorithm is demonstrated by the experiments in Section 4. The paper is concluded in Section 5.

2. Related Work

Many approaches have been proposed for distance-based clustering. They can be broadly divided into partitional, hierarchical, grid-based, graph-based, and model-based clustering.

Partitional clustering methods use iterative optimisation, where the relocation of different points to different clusters (i.e. partitions) is performed. In general, it requires a user specified number of clusters k; an inappropriate choice of k may yield unexpected results. The most known algorithm is k-means (MacQueen, 1967) that is based on centre-based clustering and if directly applied can only find globular clusters (Wu *et al*., 2008).

It is significantly sensitive to noise, outliers, and the initialisation of k clusters' centres. The method's time complexity is $O(knt)$, where n is the number of points and t is the number of iterations. Other partition-based representatives are k-medoids (Kaufman and Rousseeuw, 2001) and CLARANS (Ng and Han, 2002), which consider a cluster's centre as one of the points.

The DBSCAN (Ester *et al*., 1996) is a density-based clustering method that constructs core points, where their ϵ-neighbourhoods contain at least *MinPts* points. The method then iteratively finds density-reachable points from the core points. DBSCAN is robust to noise, and can discover arbitrary type of clusters as long the dataset does not contain large fluctuations in densities. This method has $O(n\log(n))$ time complexity if a suitable indexing data structure is used. Other popular density-based methods with the same time complexities are OPTICS (Ankerst *et al*., 1999) and DENCLUE (Hinneburg and Keim, 2003).

Hierarchical clustering algorithms divide the dataset into multiple clusters (i.e. divisive clustering), or merge the dataset elements in a bottom-up manner (i.e. agglomerative clustering). The result of such clustering is a hierarchical tree of clusters that can be visually represented in a tree diagramme-dendrogram. The traditional hierarchical single-linkage method is capable of detecting well-separated clusters of arbitrary shape, whilst complete-linkage and average-linkage cannot, and are more suitable for separating "touching" clusters. Their main drawback is their time complexity (i.e. single-linkage $O(n^2)$, complete-linkage and average-linkage $O(n^2\log(n))$. BIRCH (Zhang *et al*., 1996), CURE (Guha *et al*., 1998), LSH-link (Koga *et al*., 2007), and adaptive hierarchical clustering (Şerban and Câmpan, 2008) are all examples of hierarchical clustering algorithms. A sweep-line hierarchical clustering algorithm (Žalik and Žalik, 2009) can detect well-separated clusters of arbitrary shape within large datasets. However, it is limited to R^2 space, and cannot cope with noisy data. It has expected time complexity of $O(n\log(n))$. CHAMELEON (Karypis *et al*., 1999) is an agglomerative hierarchical clustering algorithm based on a dynamic model that merges clusters by evaluating the interconnectivity and closeness between each pair of clusters from the k-nearest neighbour's graph. It is suitable for discovering arbitrary type of clusters and has $O(n^2)$ time complexity. Similarly, the Mitosis (Yousri *et al*., 2009) clustering algorithm is based on a dynamic model. SPARCL (Chaoji *et al*., 2009) combines partitional and hierarchical clustering by merging seed clustered partitions in a hierarchical manner.

Spectral clustering is a branch of graph-based clustering. A fully connected weighted graph is constructed from the dataset, and partitioned using a cut off criterion. Shi and Malik (2000) approximated the normalised min-cut problem by extracting the eigenvectors from the graph's Laplacian matrix. The time complexity of this method is $O(n^3)$, which is unsuitable for clustering larger datasets. Many spectral clustering methods have been developed, where each handles eigenvectors differently (for an overview see Verma and Meila, 2003).

Grid-based clustering methods hierarchically divide space into a finite number of cells. The main disadvantage is that the quality of clusters depends on the cells' coarseness. Unfortunately, the amount of cells increases exponentially with the number of dimen-

sions. Known grid-based clustering methods are STING (Wang *et al*., 1997), WaveCluster (Sheikholeslami *et al*., 1998), and CLIQUE (Agrawal*et al*., 1998).

Model-based clustering attempts to optimise the fitting of a certain mathematical model over the data (Fraley and Raftery, 1998). It is assumed that the data is composed of a mixture of probability distributions. One of the better known model-based clustering methods is based on a mixture of Gaussians, where the EM algorithm is used for the likelihood function maximisation (Dempster *et al*., 1977). Although this can be used for clustering multivariate datasets, EM can converge into local extremes without careful initialisation (Kavaliauskas and Rudzkis, 2005). Therefore, more advanced solutions were developed (Rudzkis *et al*., 2013).

Nowadays, multidimensional datasets are typically large and can contain noise. Good quality clustering algorithms are computationally expensive and are, therefore, inappropriate for fast clustering of larger datasets. Some algorithms have low time complexities overall, but are inefficient for identifying arbitrary type of clusters, whilst other algorithms cannot handle multiple dimensions. This paper proposes a new distance-based clustering method SHCA using a novel dynamic model, which is described in the next section.

3. SHCA: Sweep-Hyperplane Clustering Algorithm

The proposed clustering method SHCA exploits the sweeping paradigm, a widely used method in computational geometry. Sweeping was introduced by Shamos and Hoey (1976) for solving the line-segment intersection problem. The word sweep is used because it can be imagined that a line in R^2, a plane in R^3, or a hyperplane in R^N $(N > 3)$ space, iteratively moves (i.e. sweeps) across the space. It briefly stops when an event occurs, i.e. when the sweep-line/plane/hyperplane collides with a point. During the stop, the algorithm updates the adequate data structure. The proposed SHCA uses hyperplanes that stop at each point and enable fast ANN detection. The sweeping paradigm has been already applied for clustering (Žalik and Žalik, 2009), in order to find well-separated clusters of arbitrary shape within the R^2, where the exact neighbourhood can easily be detected. SHCA generalises the sweep-based clustering to support higher dimensions.

When considering the distance threshold parameter d for d-nearest neighbourhood around a given point, the ANN search using sweep-hyperplanes within R^N, consists of the following steps:

1. Sweep-hyperplane s_1 iteratively sweeps through the data sorted by the 1-st dimension and stops at unvisited point q. Initialise $i = 1$.
2. Sweep-hyperplane $s_{(i+1)}$ sweeps the subspace around s_i with boundary d (i.e. $S_i = [s_i - d, s_i + d]$), where the data is sorted by an arbitrary $(i+1)$-th dimension. While $i \neq (N+1)$ increases i and repeats the 2nd step.
3. When $i = (N+1)$, the subspace hypercube $I = S_i \cap S_{(i+1)} \cap \cdots \cap S_N$, $I \in R^N$ corresponds to the intersection of the subspaces around the hyperplanes, which defines the ANN of point q. Points within subspace I are in close proximity, where the local clustering can be performed (see Fig. 1).

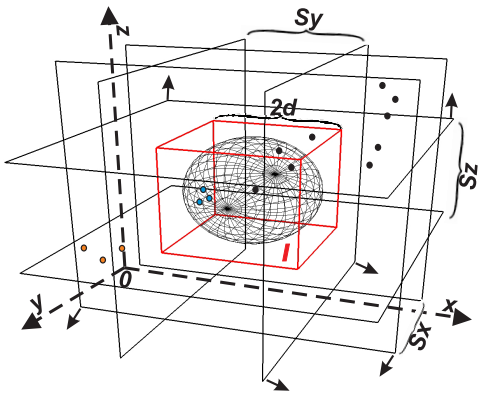


Fig. 1. Illustration of the ANN around point q using the sweep-hyperplanes approach in R^3. Considering the L_2-norm, the proximity around point q is defined with a unit sphere (i.e. d-ANN) within the unit cube (i.e. L_∞-norm proximity) representing the subspace I. The black points are those yet to be clustered. The radius of the unit sphere is d and the side length of the unit cube is $2d$.

4. If all points have been swept, the algorithm terminates, otherwise the steps are repeated by moving the hyperplanes ahead, thus visiting the remaining points.

Essentially, the sweep-hyperplanes construct a hypercube I around a given point. The amount of sweep-hyperplanes increases as the number of dimensions increase, hence 2 sweep-lines are necessary in R^2, 3 sweep-planes in R^3 (see Fig. 1), and N sweep-hyperplanes in R^N $(N > 3)$ space. SHCA performs the sweeping by using advancing fronts (*AFs*) for points' storage and fast detection of the ANNs. When the sweep-hyperplane visits a given point, it is virtually projected onto the advancing front AF_i, where the virtual points are ordered by the i-th dimension. The AF can be imagined as a form of polyline (see Fig. 3). The projection onto the AF polyline is realised as an insertion of the point into the AF data structure. SHCA uses the skip-list data structure for AF, where the details about its implementation are given in Section 3.1.

The overview of SHCA is given in Fig. 2. The method takes as input the data consisting of points in R^N space, distance threshold parameter d, and optionally the *MinPts* threshold parameter for handling noise. Parameter d defines the maximum distance the neighbouring points can be separated in order to belong to the same cluster, whilst *MinPts* defines the minimum amount of points within ANN to be considered when forming a cluster. By default $MinPts = 2$ in order to avoid outliers, unless otherwise specified. At first the data is sorted into an ascending order according to the first dimension N_1 (line 2). After the initial sorting, the incremental sweeping phase begins. The point q is visited by the sweep-hyperplane of the 1st dimension, where q is inserted into the AF_1 (line 4). The points already included in the *AFs* are further considered if they are located within the subspace I, where q represents I's centre. This is done in an incremental way; if a given point $p \in AF_i$ lies within the subspace S_i (line 7), then it is re-projected to AF for one dimension higher (i.e. insertion into $AF_{(i+1)}$) (line 8). Otherwise the points outside I are removed from AF_k with k within the range from i to N (lines 10–11). The $AF_N = I$ contains the $d\sqrt{N}$-ANN of point q. The points located within I guarantee the

```
1:  SHCA(data, d, [MinPts = 2])
2:    data_S ← sort ∀ p ∈ data by N_1
3:    for each q ∈ data_S do
4:      AF_1.add(q)
5:      for i ← 1 to N do
6:        for each p ∈ AF_i do
7:          if p^i ≥ q^i − d ∧ p^i ≤ q^i + d then
8:            if i ≠ N ∧ p ∉ AF_(i+1) then AF_(i+1).add(p)
9:          else
10:           for k ← i to N do
11:             if p ∈ AF_k then AF_k.remove(p)
12:   if |AF_N| ≥ MinPts then
13:       for each p ∈ ANN do
14:         if ||p − q|| > d then ANN.remove(p)
15:         else if p ∈ C then C_M ← max(|C|, |C_M|)
16:       if |C_M| = 0 then
17:         clusters.add(C_new)
18:         for each p ∈ ANN do p ∈ C_new
19:       else
20:         for each p ∈ ANN do
21:           if p ∈ C then
22:             C_M ← merge(C_M, C)
23:           else
24:             p ∈ C_M
25:   return clusters
```

Fig. 2. Pseudocode for SHCA.

exact neighbourhood of q only when using the L_∞-norm (i.e. Chebyshev distance metric), which is seldom used for clustering. Therefore, a new dynamic model is considered in the R^N space, in order to find d-ANN within I, where d changes dynamically based on the relative densities of neighbouring points. This allows detection of clusters of arbitrary shape, size and density, as will be described in Section 3.2. If the size of AF_N is less than *MinPts*, the points are discarded as potential noise, unless they are in d-ANN of any other point having more than *MinPts* neighbours. In the case where no point within d-ANN belongs to any cluster from the previous sweep iteration, a new cluster is created (lines 16–18). Otherwise, if at least one of the points belongs to a cluster, then the largest cluster C_M is found. All the non-clustered points and smaller clusters inside cd-ANN are then merged to C_M (lines 20–24).

The SHCA can be categorised as an agglomerative hierarchical clustering algorithm, because smaller clusters are created at first, and later merged in a bottom-up manner regarding to the distances from neighbouring clusters. The reason for the points being sorted at the beginning is to enable a faster incremental sweeping phase. The high performance of SHCA lies in the use of advancing fronts implemented with skip list data structure, as described in the next section.

3.1. *Advancing Front*

SHCA's high performance lies in the reduction of the high-dimensional ANN search into one-dimensional searches on advancing fronts. During the sweeping phase, points within a given subspace S_i are projected to AF_i (see Fig. 3(a)). The subspace S_i is likely to change

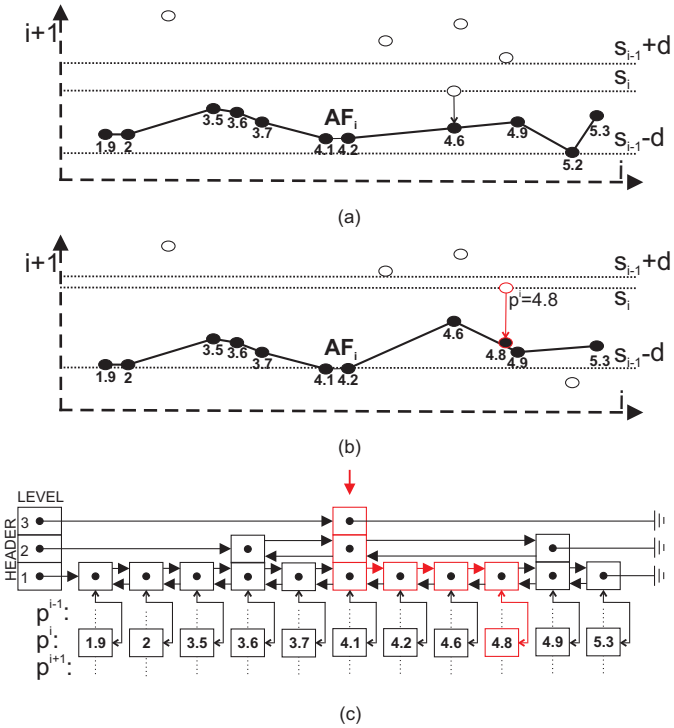


Fig. 3. Illustration of AF_i for a given subspace S_i where (a) a new point is being projected to the advancing front within S_i; (b) the next subspace S_i for unvisited point q is swept by the hyperplane s_i; (c) the implementation of 1–3 DSL for the example in (b), where the points within S_i are sorted according to i-th dimension. The path of insertion for the newly projected point is highlighted in red.

when a new unvisited point q is visited by s_1. Hence, some older points on AF_i might be removed, as shown in Fig. 3(b), where the point $p^i = 5.2$ is being removed.

The points are ordered on insertion into the AF_i in ascending order, by the i-th dimension. The AF is implemented as a skip list – a self-balanced data structure, introduced by Pugh (1990). The skip list has the same time complexity as a self-balanced tree (e.g. red-black tree) but, in practice, it has turned out to be considerably faster. The points inside the skip list are stored in a sorted single-linked list at the bottom (considered as level 1), and are accessible from the higher levels' linked lists. The number of levels is defined by the height h. The skip list used by SHCA is a deterministic skip list (DSL), proposed by Munro *et al.* (1992). The DSL used in SHCA consists of doubly-linked lists that is particularly efficient for the range search operation in AF, where the points within the vicinity of the newly visited point inside subspace I have to be found (e.g. similarly as applied for the fast nearest neighbour search in R^2, Zadravec *et al.*, 2008). The notion of A–B DSL defines that the skip list should have a gap of size $[A, B]$) between the elements in the list at a level higher than 1. According to Munro *et al.* (1992) the suitable values are $A = 1$ and $B = 3$. An example of AF structure using 1–3 DSL is shown in Fig. 3(c), where the path of the insertion for the newly projected point $p^i = 4.8$ is highlighted.

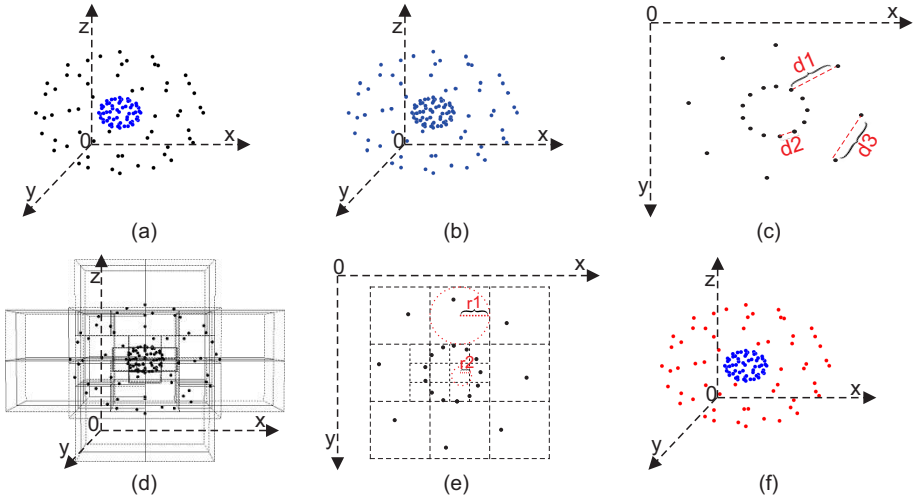


Fig. 4. Illustrative example in R^3 of undesirable scenarios when using (a) too small distance parameter d, where the black points are not clustered, or (b) too big d; (c) the XY orthogonal projection of a cross section through the middle of the data; (d) the dataset inside a 2^3-tree; (e) the same cross section as in (c), together with the 2^3-tree partitions and their bounding spheres; (f) the obtained result of SHCA using the dynamic model.

3.2. 2^N-Tree Dynamic Model

If the distance threshold parameter d is constant throughout the entire clustering process, then clusters with arbitrary densities that are not well-separated will remain undetected. Such a static model is a major weakness of most distance-based clustering algorithms, and was addressed by Karypis *et al.* (1999) who proposed a dynamic model for the CHAMELEON clustering algorithm. Clusters may be undetected, because the proximity of e.g. two clusters are smaller than the proximity of the neighbouring points within one cluster. Such clusters are not well-separated, and are impossible to find with a clustering method using a static model. Figures 4(a), 4(b) illustrates a characteristic example of nested clusters. The clearly distinguishable smaller cluster is placed inside a larger cluster. The distance between the exact neighbouring points of the larger cluster (d_3) is greater than the distance of the neighbouring points in the smaller cluster (d_2). The distance between the two clusters (d_1) is greater than d_2 and smaller than d_3 (see Fig. 4(c)). When d is small enough to detect the smaller cluster ($d \geqslant d_2 \wedge d < d_1$), the larger cluster cannot be detected (see Fig. 4(a)). If d is large enough to detect the larger cluster ($d \geqslant d_3 \wedge d > d_1$), then the smaller cluster is treated as being part of the larger cluster (see Fig. 4(b)). Without a dynamic model, a distance-based clustering algorithm will be unable to separate the two desired clusters.

A new distance-based dynamic model is proposed to be used with the SHCA, which enables adaption to the visited points together with the sweep-hyperplane movement. The dynamic model is based on the hierarchical space partitioning. It uses an unbalanced 2^N-tree (also denoted hyperoctree in Yau and Srihari, 1983) for N-dimensional spatial data partitioning. A 2^N-tree simply subdivides the space into maximum of 2^N equally

sized and separated subspaces (i.e. hypercubes), called partitions, which are further recursively subdivided. 2^N-tree represents a generalisation of quadtree (2^2-tree) and octree (2^3-tree) data structures. 2^N-tree is efficient for separating dense regions, because the distances between the points reflect the density structures (Yousri *et al.*, 2009). The 2^N-tree is constructed before SHCA is performed. The SHCA input parametre changes from d to the depth of the 2^N-tree data structure, which can be accomplished by spatial division until the tree's depth is the same as the input threshold α, or by stopping the division, when the partition contains fewer points than the input threshold β. SHCA considers both thresholds for greater flexibility. The quality of the clustering then depends on parameters α and β. Generally, less divisions of the 2^N-tree (i.e. using low α and high β) enable SHCA to find larger clusters. In contrast, more divisions of 2^N-tree yield more smaller clusters. Of course, this also highly depends on the distribution of the points, and the underlying clusters' densities.

In this way, the densities of the points can be approximated by estimating the depth of the tree. Hence the densest clusters are encapsulated within the partitions at the lowest levels of the 2^N-tree hierarchy. If considering the L_2-norm, the distance parameter d is then dynamically determined by the bounding hypersphere of the considered partition. The bounding hypersphere's diameter changes d for each visited point, hence the subspace I is variably sized around each point. Each point p located in partition P has its own distance parameter $p.d = 2r_P$, $p \in P \wedge p \in I$, where r_P is the radius of P's bounding hypersphere. Due to the approximation of the clusters' densities in a 2^N-tree, points belonging to the same cluster may be located within the spatial partitions at different tree levels. In order to solve this, the smallest distance parameter is considered from the set of the neighbouring points $d_M = \min\{p.d;\ p \in I\}$. This also efficiently separates two relatively-close clusters with different densities. The points assigned to the same cluster change their distance parameter to $p.d = d_M$. This is to ensure a continuous solution for the closer neighbouring points in different partitions. Clustering using SHCA with the dynamic model can be observed in Figs. 4(d), 4(e), and 4(f), where the desired clusters are found.

3.3. *Computational Complexity Analysis*

At first, all points are sorted by N_1 in $O(n\log(n))$ time. The insertion of n points into the *AFs* (i.e. DSLs) for each dimension is done in $O(Nn\log(n))$ (Munro *et al.*, 1992). The insertion of n points into 2^N-tree is realised in $O(n\log(n))$. For the clustering of n points in R^N, SHCA using the dynamic model has a total time complexity of $O((N+2)n\log(n))$, which is generally expected to be $O(n\log(n))$, since $N \ll n$.

4. Experimental Results

The performance of the proposed algorithm is demonstrated on different synthetic and real multidimensional datasets. SHCA, using the dynamic model, was evaluated by detecting arbitrary, nested, and noisy clusters. The results of the proposed algorithm were compared with the following clustering algorithms: single-linkage, k-means, DBSCAN,

Table 1
Clustering algorithms runtimes on different datasets (C = number of known clusters, N = number of dimensions, n = number of points).

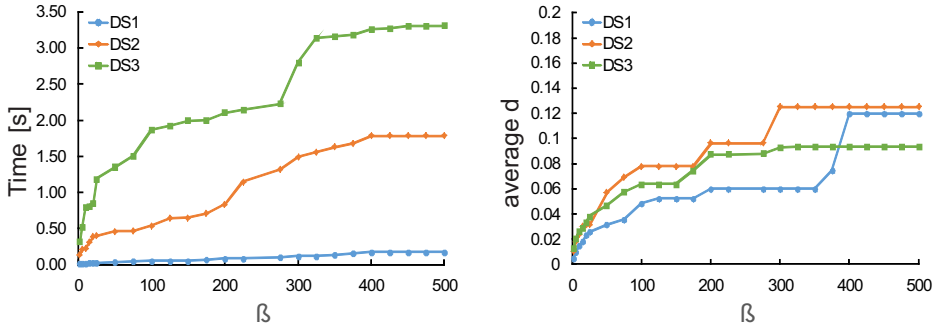
Dataset	C	N	n	Clustering runtime [s]						
				SHCA	Single-linkage	k-Means	DB-SCAN	GM-EM	CHAM-ELEON	Spectral-SHI
DS1	6	2	1512	0.065	0.036	0.019	0.125	0.112	0.145	36.214
DS2(I)	2	3	5000	0.122	0.613	0.015	1.021	0.072	1.390	610.930
DS2(II)	100	3	5000	0.116	0.596	0.791	1.031	3.685	2.824	624.356
DS3	5	3	11 260	0.835	3.552	0.126	4.247	0.373	5.655	–
Statue Lib.	5	3	106 587	162.744	–	0.626	699.443	2.453	271.767	–
Santa Barb.	28	3	2 097 152	948.189	–	100.195	–	725.832	–	–
Iris	3	4	150	0.013	0.003	0.002	0.005	0.017	0.053	0.022
Shuttle	7	9	58 000	92.056	–	3.309	520.881	4.775	189.847	–
Wine	3	13	178	0.017	0.002	0.005	0.008	0.011	0.018	0.030

Gaussian Mixture with EM (GM-EM), CHAMELEON as part of the CLUTO clustering toolkit (`http://glaros.dtc.umn.edu/gkhome`), and spectral clustering using normalised cut (Spectral-SHI). The testing environment consisted of a desktop computer with Intel i7-950 (3.06 GHz), and memory capacity of 4 GB. The L_2-norm was used in SHCA as the distance metric during the experiments. Four synthetic (DS1, DS2, DS3, and Santa Barbara Cluster) and five real (Statue of Liberty, Iris, Shuttle, and Wine) datasets were used in the experiments. The real datasets validate the efficiency of the algorithm in practice. DS1 consists of spirals as arbitrary well-separated contiguous type of clusters. DS2 consists of two interwoven tori (i.e. DS2(I)), which are constructed from smaller not well-separated clusters (i.e. DS2(II)). DS3 presents an example of touching (i.e. not well-separated) nested spheres, where each sphere was a cluster. The more nested a sphere is, the denser it is. The Statue of Liberty is a publicly available spatial dataset from Microsoft's Photosynth service (`http://photosynth.net`), under the Creative Commons license. Applying clustering on it can be useful for eliminating noise produced from the point-cloud construction process, and for simple segmentation purposes. The Santa Barbara Cluster is a publicly available cosmological spatial dataset of dark matter simulation, from The Cosmic Data ArXiv (`http://t8web.lanl.gov/people/heitmann`). The dataset consists of a large amount of dark matter particles. The particles have many attributes, but only the position of the particles was considered. With clustering, useful information could be extracted from the cosmological simulations, such as the largest known structures (filaments) in the universe. From the UCI Machine Learning Repository (`http://archive.ics.uci.edu/ml`) the following publicly available multidimensional real datasets were used: Iris, Shuttle, and Wine.

The first experiment consisted of clustering all the considered datasets using different algorithms, whilst evaluating their clustering speeds and accuracy. The datasets' properties and the clustering runtime for the used algorithms are shown in Table 1. The number of clusters for Statue of Liberty and Santa Barbara Cluster datasets was defined manually by visual inspection.

Table 2
SHCA input parameters (α and β) for different datasets, and the average distance parameter d.

Dataset	α	β	Average d
DS1	5	2	0.015
DS2(I)	6	25	0.031
DS2(II)	6	8	0.016
DS3	11	180	0.082
Statue Lib.	8	200	1.417
Santa Barb.	10	250	2.283
Iris	6	4	3.398
Shuttle	10	400	175.691
Wine	6	3	471.122



(a) (b)

Fig. 5. The increase of 2^N-tree depth influence on (a) performance and (b) average d when running SHCA over DS1, DS2, and DS3.

The user-defined input parameters for SHCA are shown in Table 2. The average dynamically-set distance parameter d is shown for each considered dataset. The depth of the tree impacts the runtime of the algorithm, which can be seen in Fig. 5(a). As can be observed, the increase of the tree depth decreases the runtime, since the considered local neighbourhoods are smaller. This is inversely proportional to the static model (e.g. single-linkage or SHCA without dynamic model), where the increase of d would also increase the runtime, as more points would have to be checked at once. A correlation between the depth of the tree (defined with parameters α and β) and average d can be seen; the deeper the tree, the smaller the average distance between the points within the obtained clusters (see Fig. 5(b)). For the experiment in Fig. 5, the α was set to 100, whilst β was gradually increasing (i.e. decreasing the tree depth).

The calculated accuracies for all considered datasets from the clustering algorithms are shown in Table 3. The clustering accuracy was verified using Rand (1971), Jaccard (1901), and Adjusted Rand (Hubert and Arabie, 1985) external cluster validation indices. Each index is within [0, 1] range, and high index's value indicates very high pair-wise points' agreements between the found clusters and the real ones that are known a priori (i.e. classes). The Rand index can be defined as the ratio of pairs of points that are correctly

Table 3
Validation indices for different clustering algorithms used on different datasets.

Dataset	SHCA	Single-linkage	k-Means	DBSCAN	GM-EM	CHAM-ELEON	Spectral-SHI
Rand							
DS1	1.000	1.000	0.721	1.000	0.721	1.000	0.689
DS2(I)	1.000	1.000	0.554	1.000	0.499	1.000	0.581
DS2(II)	1.000	0.509	0.984	1.000	0.983	1.000	0.980
DS3	0.976	–	0.747	0.878	0.721	0.983	–
Iris	0.878	0.775	0.878	0.878	0.941	0.870	0.354
Wine	0.728	0.359	0.717	0.674	0.932	0.725	0.363
Shuttle	0.863	–	0.623	0.861	0.678	0.861	–
Jaccard							
DS1	1.000	1.000	0.088	1.000	0.088	1.000	0.102
DS2(I)	1.000	1.000	0.384	1.000	0.333	1.000	0.475
DS2(II)	1.000	0.019	0.126	1.000	0.108	1.000	0.052
DS3	0.887	–	0.399	0.397	0.377	0.919	–
Iris	0.642	0.589	0.695	0.641	0.837	0.663	0.322
Wine	0.468	0.330	0.412	0.192	0.818	0.429	0.335
Shuttle	0.812	–	0.498	0.818	0.516	0.819	–
Adjusted rand							
DS1	1.000	1.000	0.003	1.000	0.003	1.000	0.002
DS2(I)	1.000	1.000	0.109	1.000	0.004	1.000	0.163
DS2(II)	1.000	0.019	0.217	1.000	0.188	1.000	0.090
DS3	0.925	–	0.417	0.512	0.381	0.947	–
Iris	0.701	0.561	0.729	0.701	0.867	0.702	0.001
Wine	0.415	0.001	0.369	0.154	0.849	0.391	0.006
Shuttle	0.679	–	0.254	0.680	0.404	0.677	–

clustered out of all possible pairs:

$$\frac{TP + TN}{TP + FP + FN + TN}, \tag{1}$$

where TP is the amount of point pairs for true positive cases (a given pair both belongs to the same cluster as the correct real class), TN for true negatives, FP for false positives, and FN for false negatives. Jaccard index is similarly defined, but neglects the use of TN. The Rand index can give a relatively high expected value for any two random groups of points. The Adjusted Rand adjusts for such scenario by reducing the expected value close to 0, and is defined as

$$\frac{2(TP * TN - FN * FP)}{(TP + FP)(FP + TN) + (TP + FN)(FN + TN)}. \tag{2}$$

The larger datasets were not tested with all the algorithms due to the runtime and memory constraints, e.g. DBSCAN and CHAMELEON detect clusters of arbitrary shape in smaller datasets, whilst due to their time complexities are inefficient for larger datasets such as Santa Barbara Cluster. Even if a larger dataset is clustered after a longer period of time, the tuning of the input parameters for a clustering algorithm would be tedious.

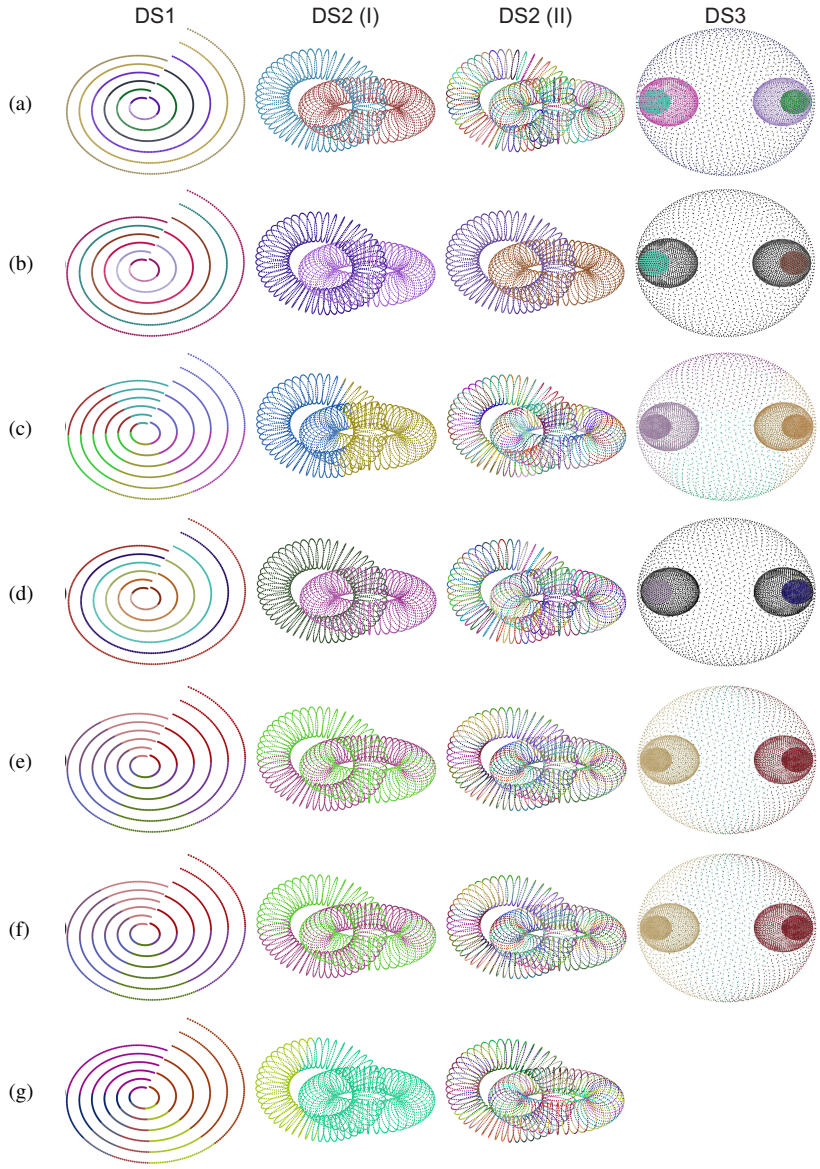


Fig. 6. Clustering results of the synthetic datasets (DS1, DS2, and DS3) using (a) SHCA, (b) single-linkage, (c) k-means, (d) DBSCAN, (e) GM-EM, (f) CHAMELEON, and (g) Spectral-SHI clustering methods. Black points indicate non-clustered points in (b), (d) for DS3.

Single-linkage can detect arbitrary well-separated clusters, if the distance matrix is created in reasonable time. However, the distance matrix's memory requirement is quadratic, hence single-linkage is unable to handle larger datasets. K-means is the fastest algorithm; however it does not detect clusters of arbitrary shape and density. Spectral-SHI clustering method suffers from high time complexity, and was tested only on the smaller-sized

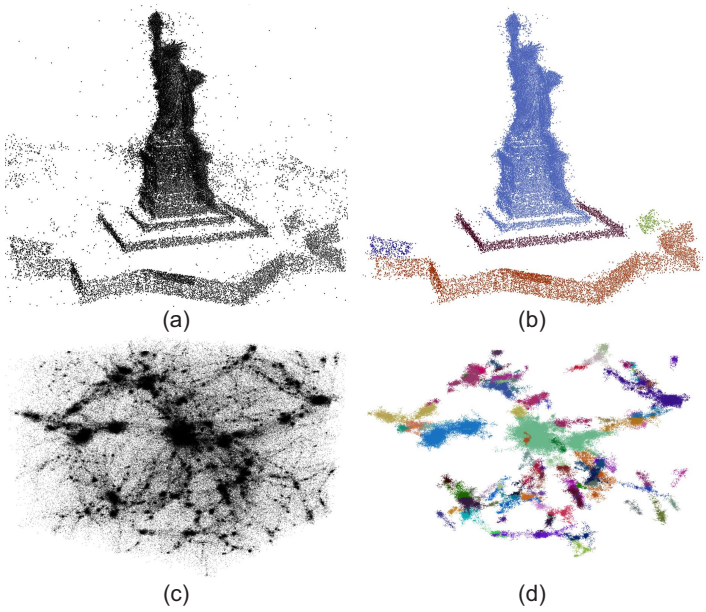


Fig. 7. Clustering larger datasets using SHCA; (a, c) raw datasets (Statue of Liberty and Santa Barbara Cluster), and (b, d) the clustering results.

datasets. The clustering results of the DS1, DS2, and DS3 datasets using different algorithms are shown in Fig. 6, whilst the results of using SHCA on larger datasets are shown in Fig. 7.

It can be observed that SHCA was considerably faster than the CHAMELEON and DBSCAN when providing similar results. The experiment confirmed that SHCA can successfully detect nested clusters of arbitrary shapes, and is also highly capable of handling larger datasets. Single-linkage was unable to find not well-separated clusters in DS2(II). DBSCAN did not detect all the spheres in DS3, as the spheres have different densities. For low dimensional datasets consisting of arbitrary type of clusters, the values of all three indices have high values, when using SHCA or CHAMELEON. For multidimensional real datasets, SHCA performed as well as CHAMELEON. The GM-EM clustering method provided similar results as k-means, although more accurate in higher dimensions. The clustering accuracies of the larger datasets were not evaluated, because the clusters' classes were unknown beforehand.

SHCA's resilience to noise was tested in the second experiment. Three synthetic datasets were used in this experiment; DS1, DS2 (II), and DS3. Randomly generated spatial Gaussian noise in four different amounts was added to the datasets (see Fig. 8 for DS2 (II)). Fig. 8(b) shows the obtained result, when *MinPts* was set to 0, where many small false clusters were produced. When the *MinPts* was set at 0.001% of the total amount of points per dataset, the false clusters were efficiently removed as shown in Fig. 8(c). The amount of noise present was calculated based on signal-to-noise ratio (SNR) that can be seen in Table 4, as well the change of accuracy in SHCA. The clustering accuracy was

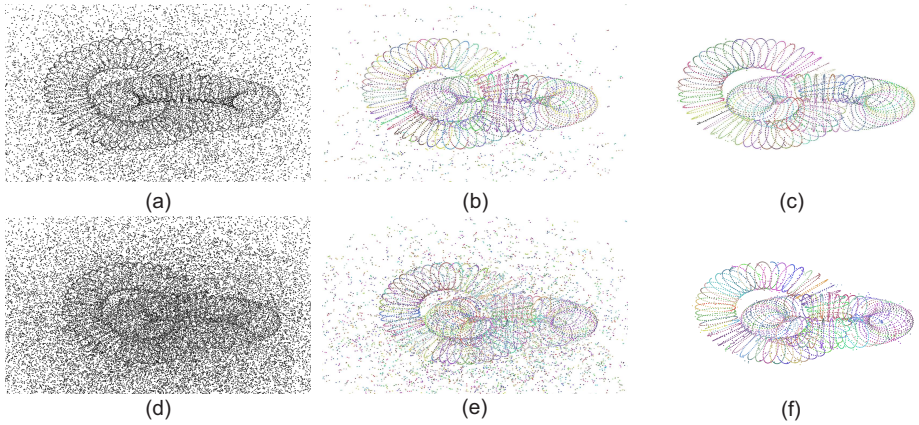


Fig. 8. (a) example of a noisy dataset DS2 (II) with SNR = 1 : 4; (b) SHCA clustering of (a) using $MinPts = 0$; (c) SHCA clustering of (a) using $MinPts = 0.001n$; (d) an example DS2 with SNR = 1 : 8; (e) SHCA clustering of d) using $MinPts = 0$; (f) SHCA clustering of d) using $MinPts = 0.001n$.

Table 4
Accuracy of SHCA for synthetic datasets with the inclusion of different amounts of noise.

SNR	1:2	1:4	1:6	1:8
Rand				
DS1	0.971	0.965	0.948	0.947
DS2(II)	0.932	0.904	0.899	0.902
DS3	0.739	0.640	0.607	0.601
Jaccard				
DS1	0.955	0.928	0.932	0.935
DS2(II)	0.862	0.864	0.874	0.886
DS3	0.701	0.545	0.408	0.568
Adjusted rand				
DS1	0.937	0.930	0.867	0.845
DS2(II)	0.863	0.788	0.732	0.670
DS3	0.350	0.214	0.203	0.175

evaluated by comparing the detected noisy clusters to the predetermined ones using the same indices as in the first experiment. This experiment confirmed that SHCA can detect clusters reasonably well within noisy environments even when the amount of noise is eight times greater than the original data. The worst recovery of the original clusters occurred in noisy DS3, which was probably because the outer spheres with low density were naturally absorbed into the noise of higher density.

5. Conclusion

A new distance-based agglomerative hierarchical clustering algorithm was proposed based on the sweeping paradigm. The SHCA uses sweep-hyperplanes to incrementally

find the ANN of each point in the dataset. This was performed efficiently with advancing fronts, implemented as deterministic skip-lists. SHCA clustering was combined with newly proposed dynamic model, based on hierarchical space division using 2^N-tree, in order to detect arbitrary clusters. SHCA requires two input parameters that define the 2^N-tree depth, whilst the number of final clusters does not need to be known a priori. Optionally, *MinPts* parameter can be used to control the robustness to noise. The experimental results demonstrated that SHCA with the dynamic model provides clustering results in reasonable time, with comparable quality to the clustering methods capable of detecting arbitrary type of clusters, even with a high presence of noise.

Acknowledgments. This work was supported by the Slovenian Research Agency under grants 1000-13-0552, J2-5479, and P2-0041. Thanks to Photosynth and Cosmic Data ArXiv for making the used datasets publicly available.

References

Agrawal, R., Gehrke, J., Gunopulos, D., Raghavan, P. (1998). Automatic subspace clustering of high dimensional data for data mining applications. In: *Proceedings of the International Conference on Management of Data*, pp. 94–105.

Ankerst, M., Breunig, M.M., Kriegel, H.P., Sander, J. (1999). OPTICS: ordering points to identify the clustering structure. In: *Proceedings of the International Conference on Management of Data*, pp. 49–60.

Chang, E., Garcia-Molina, H., Wiederhold, G. (2002). Clustering for approximate similarity search in high-dimensional spaces. *IEEE Transactions on Knowledge and Data Engineering*, 14(4), 792–808.

Chaoji, V., Al, M.H., Salem, S., Zaki, M.J. (2009). SPARCL: an effective and efficient algorithm for mining arbitrary shape-based clusters. *Knowledge and Information Systems*, 21(2), 201–229.

Dempster, A.P., Laird, N.M., Rubin, D.B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39(1), 1–38.

Ester, M., Kriegel, H.P., Sander, J., Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In: *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, pp. 226–231.

Fraley, C., Raftery, A.E. (1998). How many clusters? Which clustering method? Answers via model-based cluster analysis. *The Computer Journal*, 41(8), 578–588.

Guha, S., Rastogi, R., Shim, K. (1998). CURE: an efficient clustering algorithm for large databases. In: *Proceedings of the International Conference on Management of Data*, pp. 35–58.

Han, J., Kamber, M., Tung, A.K.H. (2001). *Spatial clustering methods in data mining: a survey*. In: Miller, H.J., Han, J. (Eds.), *Geographic Data Mining and Knowledge Discovery*. Taylor & Francis Inc., Bristol, pp. 201–231.

Hinneburg, A., Keim, D. A. (2003). A general approach to clustering in large databases with noise. *Knowledge and Information Systems*, 5, 387–415.

Hubert, L., Arabie, P (1985). Comparing partitions. *Journal of Classification*, 2(1), 193–218.

Jaccard, P. (1901). Étude comparative de la distribution florale dans une portion des alpes et des jura. *In Bulletin del la Société Vaudoise des Sciences Naturelles*, 37, 547–579.

Jiang, D., Tang, C., Zhang, A. (2004). Cluster analysis for gene expression data: a survey. *IEEE Transactions of Knowledge and Data Engineering*, 16(11), 1370–1386.

Karypis, G., Han, E.H., Kumar, V. (1999). CHAMELEON: hierarchical clustering using dynamic modelling. *Computer*, 32(8), 68–75.

Kaufman, L., Rousseeuw, P.J. (2001). *Finding Groups in Data: An Introduction to Clustering Analysis (Wiley Series in Probability and Statistics)*, Wiley-Interscience, New York.

Kavaliauskas, M., Rudzkis, R. (2005). Multivariate data clustering for the Gaussian mixture model. *Informatica*, 16(19), 61–74.

Kiriş, Ş. (2013). Multi-criteria inventory classification by using a fuzzy analytic network process (ANP) approach. *Informatica*, 24(2), 199–217.

Koga, H., Ishibashi, T., Watanabe, T. (2007). Fast agglomerative hierarchical clustering algorithm using locality-sensitive hashing. *Knowledge and Information Systems*, 12(1), 25–53.

MacQueen, J. (1967). Some methods for classifications and analysis of multivariate observations. In: *Proceedings of the 5th Berkeley Symposium Mathematical Statistics and Probability*, pp. 281–297.

Munro, J.I., Papadakis, T., Sedgewick, R. (1992) Deterministic skip lists. In: *Proceedings of the 3rd ACM-SIAM Symposium Discrete Algorithms*, pp. 367–375.

Nene, S.A., Nayar, S.K. (1997). A simple algorithm for nearest neighbor search in high dimensions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(9), 989–1003.

Ng, R.T., Han, J. (2002). CLARANS: a method for clustering objects for spatial data mining. *IEEE Transactions on Knowledge and Data Engineering*, 14(5), 1003–1016.

Pugh, W. (1990). Skip lists: a probabilistic alternative to balanced trees. *Communications of the ACM*, 33(6), 668–676.

Rand, W.M. (1971). Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66, 846–850.

Rudzkis, R., Bakshaev, A., Wiederhold, G. (2013). Goodness of fit tests based on kernel density estimators. *Informatica*, 24(3), 447–460.

Şerban, G., Câmpan, A. (2008). Hierarchical adaptive clustering. *Informatica*, 19(1), 101–112.

Shamos, M.I., Hoey, D. (1976). Geometric intersection problems. In: *Proceedings of the 17th Annual Symposium on Foundations of Computer Science*, pp. 208–215.

Sheikholeslami, G., Chatterjee, S., Zhang, A. (1998). Wavecluster: a multi-resolution clustering approach for very large spatial databases. In: *Proceedings of the 24rd International Conference on Very Large Data Bases*, pp. 428–439.

Shi, J., Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8), 888–905.

Steinbach, M., Karypis, G., Kumar, V. (2000). A comparison of document clustering techniques. In: *KDD Workshop on Text Mining*.

Steinbach, M., Ertöz, L., Kumar, V. (2003). The challenges of clustering high dimensional data. In: Wille, L.T. (Ed.), *New Vistas in Statistical Physics-Applications in Econophysics, Bioinformatics, and Pattern Recognition*, Springer, Berlin.

Theodoridis, S., Koutroumbas, K. (2008). *Pattern Recognition*, Academic Press, Burlington.

Verma, D., Meila, M. (2003). A comparison of spectral clustering algorithms. *Technical report*, Washington University.

Wang, W., Yang, J., Muntz, R. (1997). STING: a statistical information grid approach to spatial data mining. In: *Proceedings of the 23rd International Conference on Very Large Data Bases*, pp. 186–195.

Wu, X., Kumar, V., Ross, Q., Ghosh, J., Yang, Q., Motoda, H., McLachlan, G.J., Ng, A., Liu, B., Yu, P.S., Zhou, Z.H., Steinbach, M., Hand, D.J., Steinberg, D. (2008). Top 10 algorithms in data mining. *Knowledge and Information Systems*, 14(1), 1–37.

Yau, M.M., Srihari, S.N. (1983). A hierarchical data structure for multidimensional digital images. *Communications of the ACM*, 26(7), 504–515.

Yousri, N.A., Kamel, M.S., Ismail, M.A. (2009). A distance-relatedness dynamic model for clustering high dimensional data of arbitrary shapes and densities. *Pattern Recognition*, 42(7), 1193–1209.

Zadravec, M., Brodnik, A., Mannila, M., Wanne, M., Žalik, B. (2008). A Practical approach to the 2D incremental nearest-point problem suitable for different point distributions. *Pattern Recognition*, 41(2), 646–653.

Žalik, K.R., Žalik, B. (2009). A sweep-line algorithm for spatial clustering. *Advances Engineering Software*, 40(6), 445–451.

Zhang, T., Ramakrishnan, R., Livny, M. (1996). Birch: an efficient data clustering method for very large databases. In: *Proceedings of the international conference on Management of data*, pp. 103–114.

N. Lukač is a PhD student at University of Maribor, Slovenia. He received his BSc and MSc in computer science at University of Maribor, in 2010 and 2012, respectively. His research interests include clustering, massive data processing, general-purpose computing on GPU and visualizations.

B. Žalik is a Professor of Computer Science at University of Maribor, Slovenia. He obtained BSc in Electrical Engineering in 1985, MSc and PhD in Computer Science in 1989 and 1993, respectively. He is the head of Laboratory for Geometric Modelling and Multimedia Algorithms at Faculty of Electrical Engineering and Computer Science, University of Maribor. His research interests include processing of geometric data, scientific visualizations and geographic information systems. He is an IEEE member.

K.R. Žalik received the PhD degree in computer science from University of Maribor, in 1993 She is currently an associate professor of computer science at University of Maribor. Her research interests include machine learning, in particular unsupervised learning and feature learning. Her scientific contributions include approaches to fast clustering methods and to fuzzy and crisp clustering validation indices. She has presented her research work on conferences and in journals such as the Pattern Recognition and Pattern Recognition Letter. She has served as a reviewer for these and other journals.

Naujas hiperplokštuma pagrįstas klasterizavimo algoritmas naudojant dinaminį modelį

Niko Lukač, Borut Žalik, Krista Rizman Žalik

Klasterizavimas – vienas iš populiariausių neprižiūrimojo mokymosi metodų, kurio tikslas atrasti „užslėptas“ struktūras duomenyse. Šiame straipsnyje aprašomas atstumu pagrįstas klasterizavimo algoritmas, kuriame artimiausio kaimyno paieškai naudojamos hiperplokštumos. Naujas, atstumu pagrįstas dinaminis modelis, praplečia SHCA algoritmo galimybes. Atlikti eksperimentiniai tyrimai su realiomis ir dirbtinai sugeneruotomis duomenų aibėmis, demonstruoja pasiūlyto algoritmo pranašumą ir efektyvumą.