Proc. 5th Int. Conf. on Data Warehousing and Knowledge Discovery (DaWaK'03), Prague, Czech Republic, 2003

Similarity Search in Structured Data

Hans-Peter Kriegel and Stefan Schönauer

University of Munich
Institute for Computer Science
{kriegel, schoenauer}@informatik.uni-muenchen.de

Abstract. Recently, structured data is getting more and more important in database applications, such as molecular biology, image retrieval or XML document retrieval. Attributed graphs are a natural model for the structured data in those applications. For the clustering and classification of such structured data, a similarity measure for attributed graphs is necessary. All known similarity measures for attributed graphs are either limited to a special type of graph or computationally extremely complex, i.e. NP-complete, and are, therefore, unsuitable for data mining in large databases. In this paper, we present a new similarity measure for attributed graphs, called matching distance. We demonstrate, how the matching distance can be used for efficient similarity search in attributed graphs. Furthermore, we propose a filter-refinement architecture and an accompanying set of filter methods to reduce the number of necessary distance calculations during similarity search. Our experiments show that the matching distance is a meaningful similarity measure for attributed graphs and that it enables efficient clustering of structured data.

1 Introduction

Modern database applications, like molecular biology, image retrieval or XML document retrieval, are mainly based on complex structured objects. Those objects have an internal structure that is usually modeled using graphs or trees, which are then enriched with attribute information (cf. figure 1). In addition to the data objects, those modern database applications can also be characterized by their most improtant operations, which are extracting new knowledge from the database, or in other words data mining. The data mining tasks in this context require some notion of similarity or dissimilarity of objects in the database.

A common approach is to extract a vector of features from the database objects and then use the Euclidean distance or some other L_p-norm between those feature vectors as similarity measure. But often this results in very high-dimensional feature vectors, which even index structures for high-dimensional feature vectors like the X-tree [1], the IQ-tree [2] or the VA-file [3], can no longer handle efficiently due to a number of effects usually described by the term 'curse of dimensionality'.

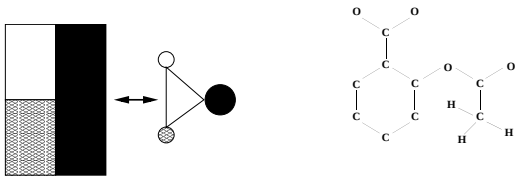


Fig. 1. Examples of attributed graphs: an image together with its graph and the graph of a molecule.

Especially for graph modeled data, the additional problem arises how to include the structural information into the feature vector. As the structure of a graph cannot be modeled by a low-dimensional feature vector, the dimensionality problem gets even worse. A way out of this dilemma is to define similarity directly for attributed graphs. Consequently, there is a strong need for similarity measures for attributed graphs. Several approaches to this problem have been proposed in recent time. Unfortunately, all of them have certain drawbacks, like being restricted to special graph types or having NP-complete time complexity, which makes them unusable for data mining applications. Therefore, we present a new similarity measure for attributed graphs, called the edge matching distance, which is not restricted to special graph types and can be evaluated efficiently. Additionally, we propose a filter-refinement architecture for efficient query processing and provide a set of filter methods for the edge matching distance.

The paper is organized as follows: In the next section, we describe the existing similarity measures for attributed graphs and discuss their strengths and weaknesses. The edge matching distance and its properties are presented in section 3, before the query architecture and the filter methods are introduced in section 4. In section 5, the effectiveness and efficiency of our methods is demonstrated in experiments with real data from the domain of image retrieval, before we finish with a short conclusion.

2 Related Work

As graphs are a very general object model, graph similarity has been studied in many fields. Similarity measures for graphs have been used in systems for shape retrieval [4], object recognition [5] or face recognition [6]. For all those measures, graph features specific to the graphs in the application, are exploited in order to define graph similarity. Examples of such features are a given one-to-one mapping between the vertices of different graphs or the requirement that all graphs are of the same order.

A very common similarity measure for graphs is the edit distance. It uses the same principle as the well known edit distance for strings [7, 8]. The idea is to determine the minimal number of insertion and deletions of vertices and edges

to make the compared graphs isomorphic. In [9] Sanfeliu and Fu extended this principle to attributed graphs, by introducing vertex relabeling as a third basic operation beside insertions and deletions. In [10] this measure is used for data mining in a graph.

Unfortunately, the edit distance is a very time-complex measure. Zhang, Statman and Shasha proved in [11] that the edit distance for unordered labeled trees is NP-complete. Consequently, in [12] a restricted edit distance for connected acyclic graphs, i.e. trees, was introduced.

Papadopoulos and Manulopoulos presented another similarity measure for graphs in [13]. Their measure is based on histograms of the degree sequence of graphs and can be computed in linear time, but does not take the attribute information of vertices and edges into account.

In the field of image retrieval, similarity of attributed graphs is sometimes described as an assignment problem [14], where the similarity distance between two graphs is defined as the minimal cost for mapping the vertices of one graph to those of another graph. With an appropriate cost function for the assignment of vertices, this measure takes the vertex attributes into account and can be evaluated in polynomial time. This asssignment measure, which we will call vertex matching distance in the rest of the paper, obviously completely ignores the structure of graphs, i.e. they are just treated as sets of vertices.

3 The Edge Matching Distance

As we just described, all the known similarity measures for attributed graphs have certain drawbacks. Starting from the edit distance and the vertex matching distance we propose a new method to measure the similarity of attributed graphs. This method solves the problems mentioned above and is useful in the context of large databases of structured objects.

3.1 Similarity of Structured Data

The similarity of attributed graphs has several major aspects. The first one is the structural similarity of graphs and the second one is the similarity of the attributes. Additionally, the weighting of the two just mentioned aspects is significant, because it is highly application dependent, to what extent the structural similarity determines the object similarity and to what extent the attribute similarity has to be considered.

With the edit distance between attributed graphs there exists a similarity measure that fulfills all those conditions. Unfortunately, the computational complexity of this measure is too high to use it for clustering databases of arbitrary size. The vertex matching distance on the other hand can be evaluated in polynomial time, but this similarity measure does not take the structural relationships between the vertices into account, which results in a too coarse model for the similarity of attributed graph. For our similarity measure, called the edge matching

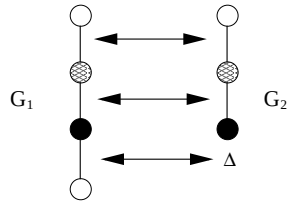


Fig. 2. An example of an edge matching between the graphs G_1 and G_2.

distance, we also rely on the principle of graph matching. But instead of matching the vertices of two graphs, we propose a cost function for the matching of edges and then derive a minimal weight maximal matching between the edge sets of two graphs. This way not only the attribute distribution, but also the structural relationships of the vertices are taken into account. Figure 2 illustrates the idea behind our measure, while the formal definition of the edge matching distance is as follows:

Definition 1. *(edge matching, edge matching distance)*
Let $G_1(V_1, E_1)$ and $G_2(V_2, E_2)$ be two attributed graphs. Without loss of generality, we assume that $|E_1| \geq |E_2|$. The complete bipartite graph $G_{em}(V_{em} = E1 \cup E_2 \cup \Delta, E_1 \times (E_2 \cup \Delta))$, where Δ represents an empty dummy edge, is called the edge matching graph of G_1 and G_2. An edge matching between G_1 and G_2 is defined as a maximal matching in G_{em}. Let there be a non-negative metric cost function $c : E_1 \times (E_2 \cup \Delta) \rightarrow \mathbb{R}_0^+$. We define the matching distance between G_1 and G_2, denoted by $d_{match}(G_1, G_2)$, as the cost of the minimum-weight edge matching between G_1 and G_2 with respect to the cost function c.

Through the use of an appropriate cost function, it is possible to adapt the edge matching distance to the particular application needs. This implies how individual attributes are weighted or how the structural similarity is weighted relative to the attribute similarity.

3.2 Properties of the Edge Matching Distance

In order to use the edge matching distance for the clustering of attributed graphs, we need to investigate a few of the properties of this measure. The time complexity of the measure is of great importance for the applicability of the measure in data mining applications. Additionally, the proof of the following theorem also provides an algorithm how the matching distance can be computed efficiently.

Theorem 1. *The matching distance can be calculated in $O(n^3)$ time in the worst case.*

Proof. To calculate the matching distance between two attributed graphs G_1 and G_2, a minimum-weight edge matching between the two graphs has to be determined. This is equivalent to determining a minimum-weight maximal matching

in the edge matching graph of G_1 and G_2. To achieve this, the method of Kuhn [15] and Munkres [16] can be used. This algorithm, also known as the Hungarian method, has a worst case complexity of $O(n^3)$, where n is the number of edges in the larger one of the two graphs. □

Apart from the complexity of the edge matching distance itself, it is also important that there are efficient search algorithms and index structures to support the use in large databases. In the context of similarity search two query types are most important, which are range queries and (k)-nearest-neighbor queries. Especially for k-nearest-neighbor search, Roussopoulos, Kelley and Vincent[17] and Hjaltason and Samet [18] proposed efficient algorithms. Both of these require that the similarity measure is a metric. Additionally, those algorithms rely on an index structure for the metric objects, such as the M-tree [19]. Therefore, the following theorem is of great importance for the practical application of the edge matching distance.

Theorem 2. *The edge matching distance for attributed graphs is a metric.*

Proof. To show that the edge matching distance is a metric, we have to prove the three metric properties for this similarity measure.

1. $d_{match}(G_1, G_2) \geq 0$
   The edge matching distance between two graphs is the sum of the cost for each edge matching. As the cost function is non-negative, any sum of cost values is also non-negative.
2. $d_{match}(G_1, G_2) = d_{match}(G_2, G_1)$
   The minimum-weight maximal matching in a bipartite graph is symmetric, if the edges in the bipartite graph are undirected. This is equivalent to the cost function being symmetric. As the cost function is a metric, the cost for matching two edges is symmetric. Therefore, the edge matching distance is symmetric.
3. $d_{match}(G_1, G_3) \leq d_{match}(G_1, G_2) + d_{match}(G_2, G_3)$
   As the cost function is a metric, the triangle inequality holds for each triple of edges in G_1, G_2 and G_3 and for those edges that are mapped to an empty edge. The edge matching distance is the sum of the cost of the matching of individual edges. Therefore, the triangle inequality also holds for the edge matching distance.

□

Definition 1 does not require that the two graphs are isomorphic in order to have a matching distance of zero. But the matching of the edges together with an appropriate cost function ensures that graphs with a matching distance of zero have a very high structural similarity. But even if the application requires that only isomorphic graphs are considered identical, the matching distance is still of great use. The following lemma allows to use the matching distance between two graphs as filter for the edit distance in a filter refinement architecture as will be described in section 4.1. This way, the number of expensive edit distance calculations during query processing can be greatly reduced.

Lemma 1. *Given a cost function for the edge matching which is always less than or equal to the cost for editing an edge, the matching distance between attributed graphs is a lower bound for the edit distance between attributed graphs:*

$$\forall G_1, G_2 : d_{match}(G_1, G_2) \leq d_{ED}(G_1, G_2)$$

Proof. The edit distance between two graphs is the number of edit operations which are necessary to make those graphs isomorphic. To be isomorphic, the two graphs have to have identical edge sets. Additionally, the vertex sets have to be identical, too. As the cost function for the edge matching distance is always less than or equal to the cost to transform two edges into each other through an edit operation, the edge matching distance is a lower bound for the number of edit operations, which are necessary to make the two edge sets identical. As the cost for making the vertex sets identical is not covered by the edge matching distance, it follows that the edge matching distance is a lower bound for the edit distance between attributed graphs. □

4 Efficient Query Processing Using the Edge Matching Distance

While the edge matching distance already has polynomial time complexity as compared to the exponential time complexity of the edit distance, a matching distance calculation is still a complex operation. Therefore, it makes sense to try to reduce the number of distance calculations during query processing. This goal can be achieved by using a filter-refinement architecture.

4.1 Multi-Step Query Processing

Query processing in a filter-refinement architecture is performed in two or more steps, where the first steps are filter steps that return a number of candidate objects from the database. For those candidate objects the exact similarity distance is determined in the refinement step and the objects fulfilling the query predicate are reported. To reduce the overall search time, the filter steps have to be easy to perform and a substantial part of the database objects has to be filtered out.

Additionally, the completeness of the filter step is essential, i.e. there must be no false drops during the filter steps. Available similarity search algorithms guarantee completeness if the distance function in the filter step fulfills the lower-bounding property. This means that the filter distance between two objects must always be less than or equal to their exact distance.

Using a multi-step query architecture requires efficient algorithms which actually make use of the filter step. Agrawal, Faloutsos and Swami proposed such an algorithm for range search [20]. In [21] and [22] multi-step algorithms for k-nearest-neighbor search were presented, which are optimal in the number of exact distance calculations neccessary during query processing. Therefore, we employ the latter algorithms in our experiments.

4.2 A Filter for the Edge Matching Distance

To employ a filter-refinement architecture we need filters for the edge matching distance, which cover the structural as well as the attribute properties of the graphs in order to be effective.

A way to derive a filter for a similarity measure is to approximate the database objects and then determine the similarity of those approximations. As an approximation for the structure of a graph G we use the size of that graph, denoted by $s(G)$, i.e. the number of edges in the graph. We define the following similarity measure for our structural approximation of attributed graphs:

$$d_{struct}(G_1, G_2) = |s(G_1) - s(G_2)| \cdot w_{mismatch}$$

Here $w_{mismatch}$ is the cost for matching an edge with the empty edge Δ. When the edge matching distance between two graphs is determined, all edges of the larger graph, which are not mapped onto an edge of the smaller graph, are mapped onto an empty dummy edge Δ. Therefore, the above measure fulfills the lower bounding property, i.e. $\forall G_1, G_2 : d_{struct}(G_1, G_2) \leq d_{match}(G_1, G_2)$.

Our filters for the attribute part of graphs are based on the observation that the difference between the attribute distributions of two graphs influences their edge matching distance. This is due to the fact, that during the distance calculation, edges of the two graphs are assigned to each other. Consequently, the edge matching distance between two graphs is the smaller, the more edges with the same attribute values the two graphs have, i.e. the more similar their attribute value distributions are. Obviously, it is too complex to determine the exact difference of the attribute distributions of two graphs in order to use this as a filter and an approximation of those distributions is, therefore, needed.

We propose a filter for the attribute part of graphs, which exploits the fact that $|x - y| \geq ||x| - |y||$. For attributes which are associated with edges, we add all the absolute values for an attribute in a graph. For two graphs G_1 and G_2 with $s(G_1) = s(G_2)$, the difference between those sums, denoted by $d_a(G_1, G_2)$, is the minimum total difference between G_1 and G_2 for the respective attribute. Weighted appropriately according to the cost function that is used, this is a lower bound for the edge matching distance. For graphs of different size, this is no longer true, as an edge causing the attribute difference could also be assigned to an empty edge. Therefore, the difference in size of the graphs multiplied with the maximum cost for this attribute has to be substracted from $d_a(G_1, G_2)$ in order to be lower bounding in all cases.

When considering attributes that are associated with vertices in the graphs,we have to take into account that during the distance calculation a vertex v is compared with several vertices of the second graph, namely exactly $degree(v)$ many vertices. To take care of this effect, the absolute attribute value for a vertex attribute has to be multiplied with the degree of the vertex, which carries this attribute value, before the attribute values are added in the same manner as for edge attributes. Obviously, the appropriately weighted size difference has to be substracted in order to achieve a lower bounding filter value for a node attribute.

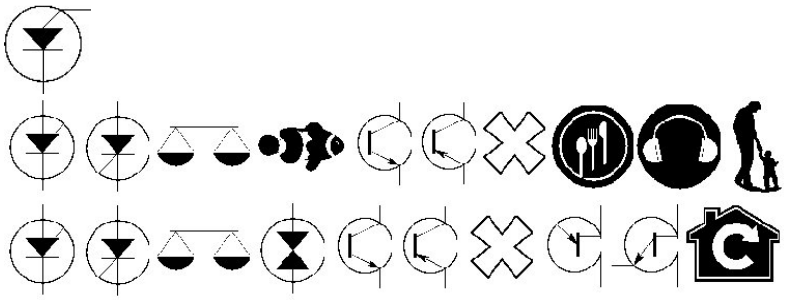


Fig. 3. Result of a 10-nearest-neighbor query for the pictograph dataset. The query object is shown on top, the result for the vertex matching distance is in the middle row and the result for the edge matching distance is in the bottom row.

With the above methods it is ensured that the sum of the structural filter distance plus all attribute filter distances is still a lower bound for the edge matching distance between two graphs. Furthermore, it is possible to precompute the structural and all attribute filter values and store them in a single vector. This supports efficient filtering during query processing.

5 Experimental Evaluation

To evaluate our new methods, we chose an image retrieval application and ran tests on a number of real world data sets:

- 705 black-and-white pictographs
- 9818 full-color TV images

To extract graphs from the images, they were segmented with a region growing technique and neighboring segments were connected by edges to represent the neighborhood relationship. Each segment was assigned four attribute values, which are the size, the height and width of the bounding box and the color of the segment. The values of the first three attributes were expressed as a percentage relative to the image size, height and width in order to make the measure invariant to scaling. We implemented all methods in Java 1.4 and performed our tests on a workstation with a 2.4GHz Xeon processor and 4GB RAM.

To calculate the cost for matching two edges, we add the difference between the values of the attributes of the corresponding terminal vertices of the two edges divided by the maximal possible difference for the respective attribute. This way, relatively small differences in the attribute values of the vertices result in a small matching cost for the compared edges. The cost for matching an edge with an empty edge is equal to the maximal cost for matching two edges. This results in a cost function, which fulfills the metric properties.

Fig. 4. A cluster of portraits in the TV-images.

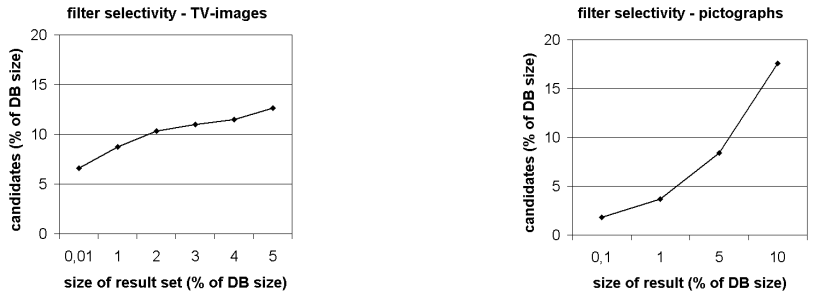
Figure 3 shows a comparison between the results of a 10-nearest-neighbor query in the pictograph dataset with the edge matching distance and the vertex matching distance. As one can see, the result obtained with the edge matching distance contains less false positives due to the fact that the structural properties of the images are considered more using this measure. It is important to note that this better result was obtained, even though the runtime of the query processing increases by as little as 5%.

To demonstrate the usefullness of the edge matching distance for data mining tasks, we determined clusterings of the TV-images by using the density-based clustering algorithm DBSCAN [23]. In figure 4 one cluster found with the edge matching distance is depicted. Although, the cluster contains some other objects, it clearly consist mainly of portraits. When clustering with the vertex matching distance, we found no comparable cluster, i.e. this cluster could only be found with the edge matching distance as similarity measure.

To measure the selectivity of our filter method, we implemented a filter refinement architecture as described in [21]. For each of our datasets, we measured the average filter selectivity for 100 queries which retrieved various fractions of the database. The results for the experiment when using the full-color TV-images are depicted in figure 5(a). It shows that the selectivity of our filter is very good, as e.g. for a query result which is 5% of the database size, more than 87% of the database objects are filtered out. The results for the pictograph dataset, as shown in figure 5(b), underline the good selectivity of the filter method. Even for a quite large result size of 10%, more than 82% of the database objects are removed by the filter. As the calculation of the edge matching distance is far more complex than that of the filter distance, it is not surprising that the reduction in runtime resulting from filter use was proportional to the number of database objects, which were filtered out.

6 Conclusions

In this paper, we presented a new similarity measure for data modeled as attributed graphs. Starting from the vertex matching distance, well known from the field of image retrieval, we developed the so called edge matching distance, which



(a) (b)

Fig. 5. Average filter selectivity for the TV-image dataset (a) and the pictograph dataset (b).

is based on minimum-weight maximum matching of the edge sets of graphs. This measure takes the structural and the attribute properties of the attributed graphs into account and can be calculated in $O(n^3)$ time in the worst case, which allows to use it in data mining applications, unlike the common edit distance. In our experiments, we demonstrate that the edge matching distance reflects the similarity of graph modeled objects better than the similar vertex matching distance, while having an almost identical runtime. Furthermore, we devised a filter refinement architecture and a filter method for the edge matching distance. Our experiments show that this architecture reduces the number of necessary distance calculations during query processing between 87% and 93%.

In our future work, we will investigate different cost functions for the edge matching distance as well as their usefullness for different applications. This includes especially, the field of molecular biology, where we plan to apply our methods to the problem of similarity search in protein databases.

7 Acknowledgement

Finally let us acknowledge the help of Stefan Brecheisen, who implemented part of our code.

References

1. Berchtold, S., Keim, D., Kriegel, H.P.: The X-tree: An index structure for high-dimensional data. In: Proc. 22nd VLDB Conf., Bombay, India (1996) 28–39
2. Berchtold, S., Böhm, C., Jagadish, H., Kriegel, H.P., Sander, J.: Independent quantization: An index compression technique for high-dimensional data spaces. In: Proc. of the 16th ICDE. (2000) 577–588

3. Weber, R., Schek, H.J., Blott, S.: A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. In: Proc. 24th VLDB Conf. (1998) 194–205
4. Huet, B., Cross, A., Hancock, E.: Shape retrieval by inexact graph matching. In: Proc. IEEE Int. Conf. on Multimedia Computing Systems. Volume 2., IEEE Computer Society Press (1999) 40–44
5. Kubicka, E., Kubicki, G., Vakalis, I.: Using graph distance in object recognition. In: Proc. ACM Computer Science Conference. (1990) 43–48
6. Wiskott, L., Fellous, J.M., Krüger, N., von der Malsburg, C.: Face recognition by elastic bunch graph matching. IEEE PAMI **19** (1997) 775–779
7. Levenshtein, V.: Binary codes capable of correcting deletions, insertions and reversals. Soviet Physics-Doklady **10** (1966) 707–710
8. Wagner, R.A., Fisher, M.J.: The string-to-string correction problem. Journal of the ACM **21** (1974) 168–173
9. Sanfeliu, A., Fu, K.S.: A distance measure between attributed relational graphs for pattern recognition. IEEE Transactions on Systems, Man and Cybernetics **13** (1983) 353–362
10. Cook, D.J., Holder, L.B.: Graph-based data mining. IEEE Intelligent Systems **15** (2000) 32–41
11. Zhang, K., Statman, R., Shasha, D.: On the editing distance between unordered labeled trees. Information Processing Letters **42** (1992) 133–139
12. Zhang, K., Wang, J., Shasha, D.: On the editing distance between undirected acyclic graphs. International Journal of Foundations of Computer Science **7** (1996) 43–57
13. Papadopoulos, A., Manolopoulos, Y.: Structure-based similarity search with graph histograms. In: Proc. DEXA/IWOSS Int. Workshop on Similarity Search, IEEE Computer Society Press (1999) 174–178
14. Petrakis, E.: Design and evaluation of spatial similarity approaches for image retrieval. Image and Vision Computing **20** (2002) 59–76
15. Kuhn, H.: The hungarian method for the assignment problem. Nval Research Logistics Quarterly **2** (1955) 83–97
16. Munkres, J.: Algorithms for the assignment and transportation problems. Journal of the SIAM **6** (1957) 32–38
17. Roussopoulos, N., Kelley, S., Vincent, F.: Nearest neighbor queries. In: Proc. ACM SIGMOD, ACM Press (1995) 71–79
18. Hjaltason, G.R., Samet, H.: Ranking in spatial databases. In: Advances in Spatial Databases, 4th International Symposium, SSD'95, Portland, Maine. Volume 951 of Lecture Notes in Computer Science., Springer (1995) 83–95
19. Ciaccia, P., Patella, M., Zezula, P.: M-tree: An efficient access method for similarity search in metric spaces. In: Proc. of 23rd VLDB Conf. (1997) 426–435
20. Agrawal, R., Faloutsos, C., Swami, A.N.: Efficient similarity search in sequence databases. In: Proc. of the 4th Int. Conf. of Foundations of Data Organization and Algorithms (FODO), Springer Verlag (1993) 69–84
21. Seidl, T., Kriegel, H.P.: Optimal multi-step k-nearest neighbor search. In: Proc. ACM SIGMOD, ACM Press (1998) 154–165
22. Korn, F., Sidiropoulos, N., Faloutsos, C., Siegel, E., Protopapas, Z.: Fast and effective retrieval of medical tumor shapes. IEEE TKDE **10** (1998) 889–904
23. Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: 2nd Int. Conf. on Knowledge Discovery and Data Mining, Portland, Oregon, AAAI Press (1996) 226–231