Published in Proceedings of 4th International Conference on Knowledge Discovery and Data Mining (KDD-98)

Algorithms for Characterization and Trend Detection in Spatial Databases

Martin Ester, Alexander Frommelt, Hans-Peter Kriegel, Jörg Sander

Institute for Computer Science, University of Munich
Oettingenstr. 67, D-80538 München, Germany
{ester | frommelt | kriegel | sander}@informatik.uni-muenchen.de

Abstract

The number and the size of spatial databases, e.g. for geo-marketing, traffic control or environmental studies, are rapidly growing which results in an increasing need for spatial data mining. In this paper, we present new algorithms for spatial characterization and spatial trend analysis. For spatial characterization it is important that class membership of a database object is not only determined by its non-spatial attributes but also by the attributes of objects in its neighborhood. In spatial trend analysis, patterns of change of some non-spatial attributes in the neighborhood of a database object are determined. We present several algorithms for these tasks. These algorithms were implemented within a general framework for spatial data mining providing a small set of database primitives on top of a commercial spatial database management system. A performance evaluation using a real geographic database demonstrates the effectiveness of the proposed algorithms. Furthermore, we show how the algorithms can be combined to discover even more interesting spatial knowledge.

Keywords: Data Mining Algorithms, Database Primitives, Spatial Data, Characterization, Trend Detection.

1. Introduction

Spatial Database Systems (*SDBS*) (Gueting 1994) are database systems for the management of spatial data. To find implicit regularities, rules or patterns hidden in large spatial databases, e.g. for geo-marketing, traffic control or environmental studies, spatial data mining algorithms are very important. A variety of data mining algorithms for mining in relational as well as spatial databases have been proposed in the literature (Fayyad *et al.* 1996, Chen *et al.* 1996., Koperski *et al.* 1996, for overviews).

In this paper, we present new algorithms for characterization and trend detection in *spatial* databases. These tasks, especially characterization in spatial databases were also studied in (Lu *et al.* 1993), (Koperski. & Han 1995), (Ng 1996) and (Knorr & Ng 1996). For methods of spatial statistics including regression methods for trend detection see e.g. (Isaaks & Srivastava 1989). A simple approach for spatial trend detection, based on a generalized clustering algorithm, is presented in (Ester *et al.* 1997b).

In (Lu *et al.* 1993), attribute-oriented induction is performed using spatial and non-spatial concept hierarchies to discover relationships between spatial and non-spatial attributes. The data is generalized along these concept hierarchies. This process yields abstractions of the data from low concept levels to higher ones which can be used to summarize or characterize the data in more general terms.

(Koperski. & Han 1995) introduces spatial association rules which describe associations between objects based on different spatial *neighborhood relations*. They present an algorithm to discover spatial rules of the form $X \rightarrow Y$ (c%), where X and Y are sets of spatial or non-spatial predicates and c is the confidence of the rule.

(Ng 1996) and (Knorr & Ng 1996) study characteristic properties of clusters of points using reference maps and thematic maps in a spatial database. For instance, a cluster may be explained by the existence of certain *neighboring objects* which may "cause" the existence of the cluster. For a given cluster of points, they give an algorithm which can efficiently find the "top-k" polygons that are "closest" to the cluster. For n given clusters of points, an algorithm is presented which can find common polygons or classes of polygons that are nearest to most, if not all, of the clusters.

Our algorithms for spatial characterization and trend detection are presented within a general framework based on database primitives for spatial data mining. Most spatial data mining algorithms make use of explicit or implicit neighborhood relations. We argue that spatial data mining algorithms heavily depend on an efficient processing of neighborhood relationships since the neighbors of many objects have to be investigated in a single run of a data mining algorithm. Therefore, the extension of an SDBS by data structures and operations for efficient processing of neighborhood relations is proposed in (Ester *et al.* 1997a).

The rest of the paper is organized as follows. We briefly introduce database primitives for spatial data mining in section 2. In section 3 and section 4, new algorithms for spatial characterization and spatial trend detection are presented. The performance of the algorithms is evaluated in section 5 using real data from a geographic information system. Section 6 concludes the paper.

2. Database Primitives for Spatial Data Mining

Our database primitives for spatial data mining (Ester *et al.* 1997a) are based on the concepts of neighborhood graphs and neighborhood paths which in turn are defined with respect to neighborhood relations between objects.

There are three basic types of spatial relations: topological, distance and direction relations which may be combined by logical operators to express a more complex neighborhood relation. We will only mention the direction relations for the 2-dimensional case because they are explicitly needed for our filter predicates. To define the direction relations, e.g. O_2 *south* O_1, we consider one representative point of the object O_1 as the origin of a virtual coordinate system whose quadrants and half-planes define the directions. To fulfil the direction predicate, *all points* of O_2 have to be located in the respective area of the plane. Figure 1 illustrates the definition of some direction relations using 2D polygons.

Obviously, the directions are not uniquely defined but there is always a smallest direction relation for two objects *A* and *B*, called the *exact direction relation* of *A* and *B*, which is uniquely determined. For instance, in figure 1 *A* and *B* satisfy the direction relations *northeast* and *east* but the exact direction relation of *A* and *B* is *northeast*.

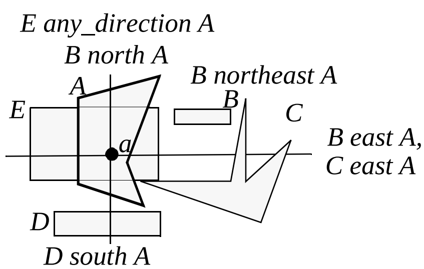


figure 1: Illustration of some direction relations

Definition 1: (***neighborhood graphs and paths***) Let *neighbor* be a neighborhood relation and *DB* be a database of spatial objects. A *neighborhood graph* $G^{DB}_{neighbor} = (N, E)$ is a graph with nodes $N = DB$ and edges $E \subseteq N \times N$ where an edge e = (n_1, n_2) exists iff *neighbor*(n_1,n_2) holds. A *neighborhood path* of length *k* is defined as a sequence of nodes $[n_1, n_2, \ldots, n_k]$, where *neighbor*(n_i, n_{i+1}) holds for all $n_i \in N, 1 \le i < k$.

We assume the standard operations from relational algebra like *selection*, *union*, *intersection* and *difference* to be available for sets of objects and sets of neighborhood paths (e.g., the operation *selection(set, predicate)* returns the set of all elements of a *set* satisfying the predicate *predicate*). Only the following important operations are briefly described:

- neighbors: Graphs × Objects × Predicates --> Sets_of_objects
- paths: Sets_of_objects --> Sets_of_paths;
- extensions: Graphs × Sets_of_paths × Integer × Predicates -> Sets_of_paths

The operation *neighbors(graph, object, predicate)* returns the set of all objects connected to *object* in *graph* satisfying the conditions expressed by the predicate *predicate*.

The operation *paths(objects)* creates all paths of length 1 formed by a single element of *objects* and the operation *extensions(graph, paths, max, predicate)* returns the set of all paths extending one of the elements of *paths* by at most *max* nodes of graph. The extended paths must satisfy the predicate *predicate*. The elements of *paths* are not contained in the result implying that an empty result indicates that none of the elements of *paths* could be extended.

Because the number of neighborhood paths may become very large, the argument *predicate* in the operations *neighbors* and *extensions* acts as a filter to restrict the number of neighbors and paths to certain types of neighbors or paths. The definition of *predicate* may use spatial as well as non-spatial attributes of the objects or paths.

For the purpose of KDD, we are mostly interested in paths "leading *away*" from the start object. We conjecture that a spatial KDD algorithm using a set of paths which are crossing the space in arbitrary ways will not produce useful patterns. The reason is that spatial patterns are most often the effect of some kind of influence of an object on other objects in its neighborhood. Furthermore, this influence typically decreases or increases more or less continuously with increasing or decreasing distance. To create only relevant paths, we introduce special filter predicates which select only a subset of all paths, thus also significantly reducing the runtime of data mining algorithms.

There are many possibilities to define "starlike" filters. The filter *starlike*, e.g., requires that, when extending a path $p = [n_1,n_2,\ldots,n_k]$ with a node n_{k+1}, the exact "final" direction of *p* may not be generalized. For instance, a path with final direction *northeast* can only be extended by a node of an edge with the exact direction *northeast*. The filter *variable-starlike* requires only that, when extending *p* the edge (n_k, n_{k+1}) has to fulfil *at least* the exact "initial" direction of *p*. For instance, a neighborhood path with initial direction *north* can be extended such that the direction *north* or the more special direction *northeast* is satisfied. Figure 2 illustrates these filters when extending the paths from a given start object. The figure also depicts another filter *vertical starlike* which is less restrictive in vertical than in horizontal direction.

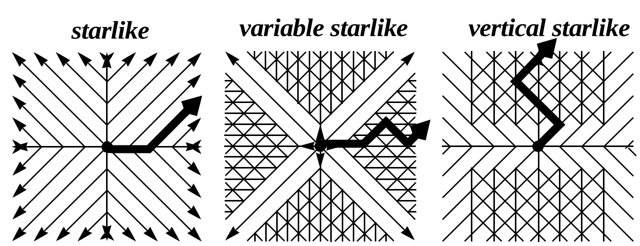
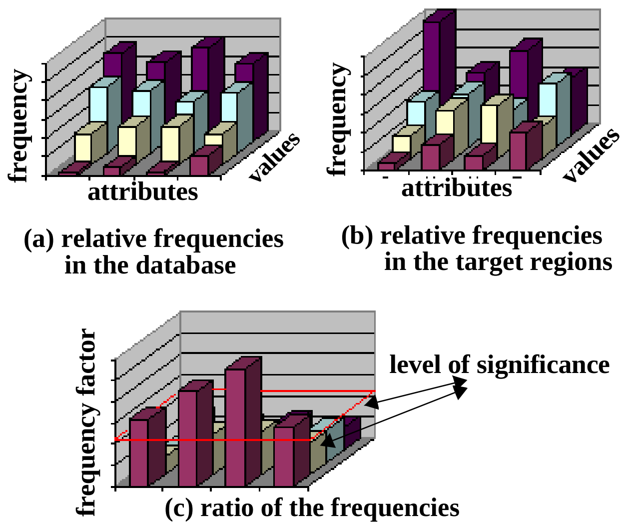


figure 2: Illustration of some filter predicates

3. Spatial Characterization

We define a *spatial characterization* of a given set of target objects with respect to the database containing these targets as a description of the spatial and non-spatial properties which are typical for the target objects but not for the whole database. We use the relative frequencies of the non-spatial attribute values and the relative frequencies of the different object types as the interesting properties. For instance, different object types in a geographic database are communities, mountains, lakes, highways, railroads etc. To obtain a *spatial* characterization, we consider not only the properties of the target objects, but also the properties of their neighbors up to a given maximum number of edges in the neighborhood graph. Figure 3 depicts an example for the relative frequencies in the database as well as in the target regions and the ratio of these frequencies in comparison with the specified level of significance.



(a) relative frequencies in the database

(b) relative frequencies in the target regions

(c) ratio of the frequencies

figure 3: Sample frequencies and differences

The task of spatial characterization is to discover the set of all tuples (attribute, value) and the set of all objects types for which the relative frequency in a set *targets*, extended by its neighbors, is significantly different from the relative frequency in *DB*. A very frequent property present only in the neighborhood of very few of the *targets* would create misleading results. Therefore, we require that such a property must also have a significantly larger relative frequency in the neighborhood of many *targets*.

Definition 2: (***spatial characterization***): Let $G^{DB}_{neighbor}$ be a neighborhood graph and *targets* be a subset of *DB*. Let *freq*^{s}*(prop)* denote the number of occurrences of the property *prop* in the set *s* and let *card(s)* denote the cardinality of *s*. The *frequency factor* of *prop* with respect to *targets* and *DB*, denoted by $f^{DB}_{targets}(prop)$, is defined as follows:

$$f^{DB}_{targets}(prop) = \frac{freq^{targets}(prop)}{card(targets)} \Big/ \frac{freq^{DB}(prop)}{card(DB)}$$

Let *significance* and *proportion* be real numbers and let *max-neighbors* be a natural number. Let $neighbors^{i}_{G}(s)$ denote the set of all objects reachable from one of the elements of *s* by traversing at most *i* of the edges of the neighborhood graph *G*. Then, the task of *spatial characterization* is to discover each property *prop* and each natural number $n \leq$ *max-neighbors* such that (1) the set *objects* = $neighbors^{n}_{G}(targets)$ as well as (2) the sets *objects* = $neighbors^{n}_{G}(\{t\})$ for at least *proportion* many $t \in$ *targets* satisfy the condition:

$$f^{DB}_{objects}(prop) \quad \begin{array}{l} \geq significance \\ \textbf{or} \\ \leq \dfrac{1}{significance} \end{array}$$

In point (1) the union of the neighborhood of all target objects is considered simultaneously, whereas in point (2) the

```
characterization(graph G_r^DB ; set of objects targets; real significance, proportion; integer max-neighbors)
initialize the set of characterizations as empty;
initialize the set of regions to targets;
initialize n to 0;
calculate frequency^DB(prop) for all properties prop = (attribute, value);
while n ≤ max-neighbors do
    for each attribute of DB and for the special attribute object type do
        for each value of attribute do
            calculate frequency^regions(prop) for property prop = (attribute, value);
            if f^DB_regions(prop) ≥ significance or f^DB_regions(prop) ≤ 1 / significance then
                add (prop, n, f^DB_regions(prop) ) to the set characterizations;
    if n < max-neighbors then
        for each object in regions do
            add neighbors( G_r^DB , object, TRUE) to regions;
    increment n by 1;
extract all tuples (prop, n, f(prop)) from characterizations which are significant in at least proportion of the
  regions with n extensions;
return the rule generated from these characterizations;
```

figure 4: Algorithm spatial characterization

neighborhood of each target object is considered separately. The parameter *proportion* specifies the minimum confidence required for the characterization rules and the frequency factors of the properties provide a measure of their interestingness with respect to the given target objects.

Figure 4 presents the algorithm for discovering spatial characterizations. The parameter *proportion* is relevant only for the last step of the algorithm, i.e. for the generation of a rule. Note the importance of the parameter *n* (that is, the maximum number of edges of the neighborhood graph traversed starting from a target object) in the resulting characterizations. For example, a property may be significant when considering all neighbors which are reachable from one of the target objects via 2 edges of the neighborhood graph. However, the same property may not be significant when considering further neighbors if then the target regions are extended by objects for which the property is not frequent. The generated rule has the following format:

$target \Rightarrow p_1\,(n_1, freq\text{-}fac_1) \wedge ... \wedge p_k\,(n_k, freq\text{-}fac_k)$.

This rule means that for the set of all targets extended by n_i neighbors, the property p_i is $freq\text{-}fac_i$ times more (or less) frequent than in the database.

4. Spatial Trend Detection

We define a *spatial trend* as a regular change of one or more non-spatial attributes when moving away from a given start object *o*. We use neighborhood paths starting from *o* to model the movement and we perform a regression analysis on the respective attribute values for the objects of a neighborhood path to describe the regularity of change. Since we are interested in trends with respect to *o*, we use the distance from *o* as the independent variable and the difference of the attribute values as the dependent variable(s) for the regression. The correlation of the observed attribute values with the values predicted by the regression function yields a measure of confidence for the discovered trend.

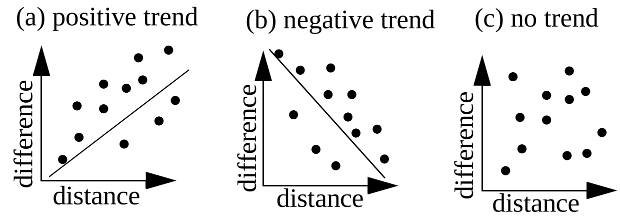


figure 5: Sample trends

In the following, we will use linear regression, since it is efficient and often the influence of some phenomenon to its neighborhood is either linear or may be transformed into a linear model, e.g. exponential regression. Figure 5 illustrates a positive and a negative (linear) trend as well as a situation where no significant (linear) trend is observed.

Definition 3: (***spatial trend detection***): Let *g* be a neighborhood graph, *o* an object (node) in *g*. and *a* be a subset of all non-spatial attributes. Let *t* be a type of function, e.g. linear or exponential, used for the regression and let *filter* be one of the filters for neighborhood paths. Let *min-conf* be a real number and let *min-length* as well as *max-length* be natural numbers. The task of *spatial trend detection* is to discover the set of all neighborhood paths in g starting from *o* and having a trend of type *t* in attributes *a* with a correlation of at least *min-conf*. The paths have to satisfy the *filter* and their length must be between *min-length* and *max-length.*

```
global-trend(graph g; object o; attribute a; type t; real min-conf, integer min-length,max-length; filter f)
initialize a list of paths to the set extensions(g, path(o), min-length, f);
initialize an empty set of observations;
initialize the last-correlation and last-paths as empty;
initialize first-pos to 1;
initialize last-pos to min-length;
while paths is not empty do
    for each path in paths do
        for object from first-pos of path to last-pos of path do
            calculate diff as a(object) - a(o) and calculate dist as dist(object,o);
            insert the tuple (diff, dist) into the set of observations;
    perform a regression of type t on the set of observations;
    if abs(correlation) of the resulting regression function ≥ min-conf then
        set last-correlation to correlation and last-paths to paths;
        if the length of the paths < max-length then
            replace the paths by the set extensions(g,paths,1, f);
            increment last-pos by 1;
            set first-pos to last-pos;
        else set paths to the empty list;
    else return last-correlation and last-paths;
return the last correlation and last-paths;
```

figure 6: Algorithm global-trend

```
local-trends(graph g; object o; attribute a; type t; real min-conf, integer min-length,max-length; filter f)
initialize a list of paths to the set extensions(g, path(o), min-length, f);
initialize two empty sets of positive and negative trends;
while paths is not empty do
    initialize the set of observations as empty;
    remove the first element of paths and take it as path;
    for object from min-length-th object of path to last object of path do
        calculate diff as a(object) - a(o) and calculate dist as dist(object,o);
        insert the tuple (diff,dist) into the set of observations;
    perform a regression of type t on the set of observations;
    if abs(correlation) of the resulting regression function ≥ min-conf then
        if correlation > 0 then
            insert the tuple (path, correlation) into the set of positive trends;
        else insert the tuple (path, correlation) into the set of negative trends;
        if the length of path < max-length then
            add the extensions(g,path,1, f) to the head of paths;
return positive-trends and negative-trends;
```

figure 7: Algorithm local-trends

Definition 3 allows different specializations. Either the set of all discovered neighborhood paths (global trend) or each of its elements (local trend) must have a trend of the specified type.

Algorithm *global-trend* is depicted in figure 6. Beginning from *o*, it creates all neighborhood paths of the same length simultaneously - starting with *min-length* and continuing until *max-length*. The regression is performed once for each of these sets of all paths of the same length. If no trend of length *l* with correlation ≥ *min-conf* is detected, then the path extensions of length *l+1*, *l+2*, . . ., *max-length* are not created. The algorithm returns the significant spatial trend with the maximum length.

Algorithm *local-trends* is outlined in figure 7. This algorithm performs a regression once for each of the neighborhood paths with length ≥ *min-length* and a path is only extended further if it has a significant trend. The algorithm returns two sets of paths showing a significant spatial trend, a set of positive trends and a set of negative trends.

Relation Bavarian Communities:

name	popula-tion	unem-ployment	for-eigners	. . .	spatial
Munich	1.300.000	0.06	0.15	. . .	
. . .	. . .	. . .	. . .	. . .	. . .

figure 8: Spatial and non-spatial attributes

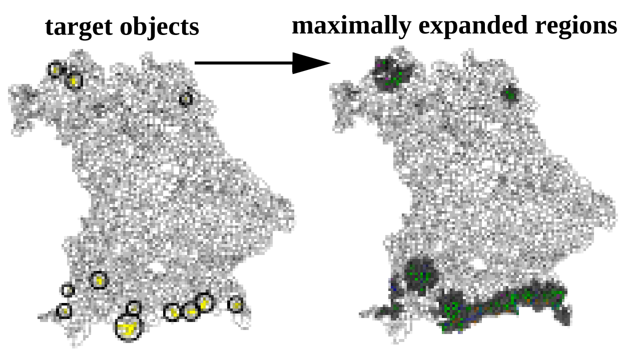
5. Performance Evaluation

We implemented the database primitives on top of the commercial DBMS Illustra (Illustra 1997) using its 2D spatial data blade which offers R-trees. The advantage of this approach is an easy and rather portable implementation. The disadvantage is that we cannot reduce the relatively large system overhead imposed by the underlying DBMS.

A geographic database on Bavaria was used for the experimental performance evaluation of our algorithms. The database contains the ATKIS 500 data (Bavarian State Bureau of Topography and Geodesy 1996) and the Bavarian part of the statistical data obtained by the German census of 1987, i.e. 2043 Bavarian communities with one spatial attribute (polygon) and 52 non-spatial attributes (such as average rent or rate of unemployment). Also included are spatial objects representing natural object like mountains or rivers and infrastructure such as highways or railroads. The total number of spatial objects in the database then amounts to 6924. The relation communities is sketched in figure 8. This geographic database may be used, e.g., by economic geographers to discover spatial rules on the economic power of communities. We performed several sets of experiments to measure the performance of our characterization and spatial trend detection algorithms. Note, that the runtime of the algorithms is in general not dependent on the database size but on the size of the input and on the number of neighborhood operations performed for the considered objects. This number depends on the average number of neighbors per object in the database which is an application dependent parameter. In our geographic information system the average number of neighboring communities is approximately six.

5.1 Characterization

The characterization algorithm usually starts with a small set of target objects, selected for instance by a condition on some non-spatial attribute(s) such as “rate of retired people = HIGH” (see figure 9, left). Then, the algorithms expands regions around the target objects, simultaneously selecting those attributes of the regions for which the distribution of values differs significantly from the distribution in the whole database (figure 9, right). In the last step of the algorithm, a characterization rule is generated describing the target regions (figure 9, bottom). In this example, not only some non-spatial attributes but also the neighborhood of mountains (after three extensions) are significant for the characterization of the target regions.



rule characterizing the target objects

object has high rate of retired people ⇒
apartments per building = very low (0, 9.1) ∧
rate of foreigners = very low (0, 8.9) ∧
rate of academics = medium (0, 6.3) ∧
average size of enterprises = very low (0, 5.8) ∧
object type = mountain (3, 4.1)

figure 9: Characterizing wrt. high rate of retired people

Table 1 reports the efficiency of our spatial characterization algorithm. The numbers are calculated as the average over all start objects for several experiments with different target sets.

characterization of target sets			
number of neighbors (*n*)	Ø *neighbors* operations	Ø no. objects in expanded region	Ø runtime per start object (min:sec)
1	**1**	**8.7**	**0:59**
2	8.8	28.7	1:73
3	**28.7**	**59.8**	**3:59**
4	**60.2**	**92.8**	**5:61**
5	**92.8**	**123.9**	**7:31**

Table 1: Performance of spatial characterization

5.2 Trend Detection

*Spatial trend*s describe a regular change of non-spatial attributes when moving away from a start object *o*. The two algorithms above may produce different patterns of change for the same start object *o*.

The existence of a global trend for a start object *o* indicates that if considering all objects on all paths starting from *o* the values for the specified attribute(s) *in general* tend to increase (decrease) with increasing distance. Figure 10 (left) depicts the result of algorithm global-trend for the attribute “average rent” and the city of Regensburg as a start object.

Algorithm *local-trends* detects single paths starting from an object *o* and having a certain trend. The paths starting from *o* may show different pattern of change, e.g., some trends may be positive while the others may be negative. Figure 10 (right) illustrates this case for the attribute “average rent” and the city of Regensburg as a start object.

The spatial objects within a trend region, i.e. either the start objects or the objects forming the paths, may be the subject of further analysis. For instance, algorithm *global-trend* may detect regions showing a certain global trend, and algorithm *local-trends* then finds within these regions some paths having the inverse trend (see figure 10). Then, we may try to find an explanation for those “inverse” paths. Another possibility is to detect “centers” for a given attribute first (using algorithm global-trend) and then apply our characterization algorithm to the centers to find their common properties. An example for this approach is presented in more detail in section 5.3.

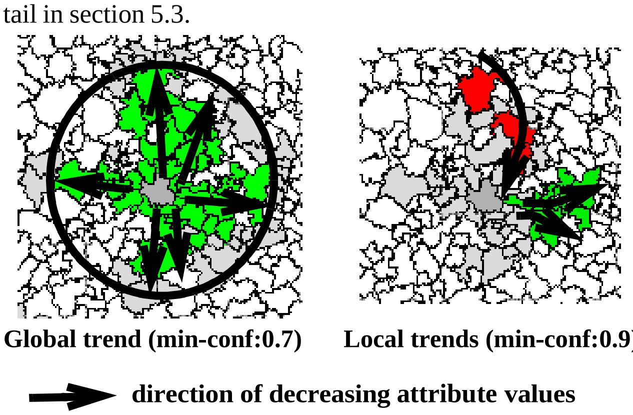


figure 10: Visualization of trends for attribute “average rent” starting from the city of Regensburg

For our performance test we applied both algorithms to the Bavaria database varying *min-confidence* from 0.6 to 0.8 for the attribute “average rent” and linear type of regression. The predicate *intersects* was used as the neighborhood relation to define the graph. The filter *vertical starlike* for paths was used because due to our domain knowledge we expected the most significant trends in north-south direction. The length of the paths was restricted by *min-length* = 4 and *max-length* = 7.

Table 2 reports the performance results for the algorithms *global-trend* and *local-trends*. The average numbers shown were calculated over all start objects.

	Algorithm ***global-trend***	
correlation	Ø *neighbors* operations	Ø runtime (sec.)
0.60	**56.6**	**91**
0.70	**55.0**	**90**
0.80	**54.3**	**85**
0.90	**53.7**	**84**

	Algorithm ***local-trends***	
correlation	Ø *neighbors* operations	Ø runtime (sec.)
0.60	**8.4**	**14.3**
0.70	**8.3**	**13.8**
0.80	**8.1**	**12.9**
0.90	**7.1**	**11.3**

Table 2: Performance of both trend algorithm

5.3 Combining Trend Detection and Characterization

In our last set of experiments, we combined trend detection and characterization. In a first step, we detected centers for attribute "average rent" using algorithm *global-trend*: minimum correlation was set to 0.7 and we selected only those communities where the slope of the trend was less than -10^{-4} and the path length was not smaller than 5, i.e. we were only interested in linear trends that are noticeably decreasing.

With this definition, we found 24 centers out of the 2043 communities. The characterization rule discovered for these centers contains the following properties:

rate of academics = high (1, 9.1),
average number of persons per household = low (1, 2.5),
rate of foreigners = low (1, 2.8).

Note that no attribute was significant for $n = 0$, i.e. without considering the neighborhood of the target object. Only if we extend the target regions by one neighbor, we can see some characteristic properties. Thus, this result could not be found by a non-spatial characterization algorithm.

6. Conclusions

In this paper, we presented new algorithms for spatial characterization and spatial trend detection. To obtain a *spatial* characterization, we consider not only the properties of the target objects but also the properties of their neighbors. The goal of spatial trend analysis is to discover patterns of change of some non-spatial attribute(s) in the *neighborhood* of a start object. The algorithms were implemented within a general framework for spatial data mining providing a small set of database primitives on top of a commercial spatial database management system. The effectiveness of the proposed algorithms was demonstrated by a performance evaluation using a real geographic database.

Future research will have to consider the following issues. The algorithm for spatial characterization might be extended to discover not only summarizing characterization rules but also discriminating rules. Furthermore, neighborhood paths may also be used as input for the well-known relational data mining algorithms such as decision tree classifiers. Alternatively, new spatial data mining algorithms operating directly on neighborhood graphs and paths will be investigated.

Acknowledgments

We thank Stefan Gundlach for fruitful discussions and his implementation of the database primitives for spatial data mining.

References

Chen M.-S., Han J., Yu P.S. 1996. Data Mining: An Overview from a Database Perspective. *IEEE Transactions on Knowledge and Data Engineering*, 8(6): 866-883.

Ester M., Kriegel H.-P. & Sander J. 1997a. Spatial Data Mining: A Database Approach. *Proc. 5th Int. Symp. on Large Spatial Databases*, 47-66, Berlin: Springer.

Ester M., Kriegel H-P, Sander J. & Xu X. 1997b. Density-Connected Sets and their Application for Trend Detection in Spatial Databases. *Proc. 3rd Int. Conf. on Knowledge Discovery and Data Mining*, 10-15, Menlo Park, CA: AAAI Press.

Fayyad U., Piatetsky-Shapiro G. & Smyth P. 1996. Knowledge Discovery and Data Mining: Towards a Unifying Framework. *Proc. 2nd Int. Conf. on Knowledge Discovery and Data Mining*, 82-88, Menlo Park, CA: AAAI Press.

Gueting, R.H. 1994. An Introduction to Spatial Database Systems. *VLDB Journal* 3(4).

Illustra Information Technologies, Inc. 1997. *Illustra User's Guide*. Release 3.3.1 for HP UX 10.1.

Isaaks E.H. & Srivastava R.M. 1989. *Applied Geostatistics*. New York: Oxford University Press.

Koperski K. & Han J. 1995. Discovery of Spatial Association Rules in Geographic Information Databases. *Proc. 4th Int. Symp. on Large Spatial Databases*, 47-66. Berlin: Springer.

Koperski K., Adhikary J., Han J. 1996. Knowledge Discovery in Spatial Databases: Progress and Challenges. Proc. SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery, Technical Report 96-08, University of British Columbia, Vancouver, Canada.

Knorr E.M. & Ng R.T. 1996. Finding Aggregate Proximity Relationships and Commonalities in Spatial Data Mining. *IEEE Transactions on Knowledge and Data Engineering*, 8(6):884-897.

Lu W., Han J., Ooi B.C. 1993. Discovery of General Knowledge in Large Spatial Databases. Proc. Far East Workshop on Geographic Information Systems, Singapore, 275-289.

Ng, R.T. 1996. Spatial Data Mining: Discovering Knowledge of Clusters from Maps. Proc. SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery, Technical Report 96-08, University of British Columbia, Vancouver, Canada.