

An ILP Approach to Spatial Clustering

Antonio Varlaro, Annalisa Appice, Antonietta Lanza, and Donato Malerba

Dipartimento di Informatica, Università degli Studi di Bari
via Orabona, 4 - 70126 Bari - Italy

{varlaro,appice,lanza,malerba}@di.uniba.it

Abstract. Spatial clustering is a fundamental task in Spatial Data Mining where the goal is to group nearby sites and form clusters of homogeneous regions. Spatial clustering must be driven by the discrete spatial structure of data that expresses the (spatial) relational constraints between separate sites. Only similar sites (transitively) connected in the discrete spatial structure may be clustered together. CORSO is a novel clustering method that resorts to an ILP approach to mine spatial data and exploits the concept of neighborhood to capture relational constraints. Spatial data are expressed in a first-order formalism and similarity among structured objects is computed as degree of matching with respect to a common generalization. Two real-world applications are described.

1 Introduction

In recent times the growing interest in the extension of data mining methods to spatial domain as well as the application of Inductive Logic Programming (ILP) to knowledge discovery in spatial database is driven by the pressure from the public and private sectors to provide solutions to a wide range of data intensive spatial applications (e.g., environmental analysis or urban planning). We consider a specific task, that is, spatial clustering and investigate spatial clustering tools as advanced means for creating models of spatially distributed phenomena.

Many approaches to discover spatial clusters have been proposed in literature. For instance, Ng and Han [11] have extended the k -medoid partitioning algorithm [6] to group point data in a set of k clusters, while Ester et al. [5] have proposed DBSCAN to detect dense clusters of arbitrary shape from point spatial data with noise. Similarly, Wang and Hamilton [16] have proposed a density-based clustering method that estimates density within a cluster on the basis of only non spatial properties of point spatial data. Conversely, Sander et al. have proposed GDBSCAN [14] that is a generalization of DBSCAN in order to cluster not only point data but also spatially extended objects (lines or areas) taking into account both spatial and non spatial attributes. At this aim, GDBSCAN uses the notion of neighborhood, that is, any binary relation that is symmetric and reflexive (e.g. distance, meet). A neighborhood relation can be modeled as graph, namely neighborhood or proximity graph [15]. This graph imposes a discrete spatial structure on data that guides the cluster detection such that only the neighbor objects may be clustered together. However, all

these methods suffer from severe limitations due to the single-table assumption [2], that is, data to be mined are stored in a single table of a relational database, such that each row (or tuple) represents an independent unit of the sample population and columns correspond to properties of units. This is a strong restriction in representing spatial objects that have different properties and are modeled by as many data tables (relational data model) as the number of object types. To face this additional level of complexity we resort to the field of (multi-)relational data mining [2] and adopt subsets of first-order logic to express the relational nature of both data to be mined and relational patterns to be discovered. In particular, ILP approach to relational data mining supplies representation and reasoning means appropriate for the spatial domain where relations among objects play a key role and are often inferred by qualitative reasoning. However, ILP methods of clustering [7, 1] generally work in the learning from interpretation setting [13] that allows to mine examples and background knowledge stored as Prolog programs exploiting expressiveness of first-order representation during cluster detection. The interpretation corresponding to each example e given the background knowledge BK is here intended as the minimal Herbrand model of $e \wedge BK$ and the implicit assumption is that separate interpretations are independent. This leads to ignore relational constraints eventually relating separate interpretations (e.g. geographic contiguity of areal units). To overcome these deficiencies, we propose to combine a graph-based partitioning algorithm with a relational clustering method to mine both relational constraints imposing the discrete spatial structure and relational data representing structured objects (spatial unit) to be clustered.

The paper is organized as follows. In the next section we discuss a new relational clustering method working in presence of the discrete spatial structure. Two applications of spatial clustering for topographic map interpretation and geo-referenced census data analysis are reported in Section 3. Finally Section 4 presents some conclusions and future works.

2 The Method

In a quite general formulation, the problem of clustering structured objects (e.g., complex areal units), which are related by links representing persistent relations between objects (e.g., spatial correlation), can be defined as follows: *Given*: (i) a set of structured objects O , (ii) a background knowledge BK and (iii) a binary relation R expressing links among objects in O ; *Find* a set of homogeneous clusters $\mathbf{C} \subseteq \wp(O)$ that is feasible with R .

Each structured object $o_i \in O$ can be described by means of a conjunctive ground formula (conjunction of ground selectors) in a first-order formalism, while background knowledge BK is expressed with first-order clauses that support some qualitative reasoning on O . In both cases, each basic component (i.e., *selector*) is a relational statement in the form $f(t_1, \dots, t_n) = v$, where f is a function symbol or *descriptor*, t_i are constants or variables (but not functions) and v is a value taken from the categorical or numerical range of f . Structured

objects are then related by R that is a binary relation $R \subseteq O \times O$ imposing a discrete structure on O . This relation may be either purely spatial (topological, distance and directional relations) or hybrid, which mixes both spatial and non spatial properties (e.g. two regions are connected by a road). The relation R can be described by the graph $G = (N_O, A_R)$ where N_O is the set of nodes n_i representing each structured object o_i and A_R is the set of arcs $a_{i,j}$ describing links between each pair of nodes $\langle n_i, n_j \rangle$ according to the discrete structure imposed by R . This means that there is an arc from n_i to n_j only if $o_i R o_j$. Let $N_R(n_i)$ be the R -neighborhood of a node n_i such that $N_R(n_i) = \{n_j \mid \text{there is an arc linking } n_i \text{ to } n_j \text{ in } G\}$, a node n_j is R -reachable from n_i if $n_j \in N_R(n_i)$, or $\exists n_h \in N_R(n_i)$ such that n_j is R -reachable from n_h .

Algorithm 1 Top-level description of CORSO algorithm.

```

1: function CORSO( $O, BK, R, h - threshold$ )  $\rightarrow$   $CList$ ;
2:  $CList \leftarrow \emptyset$ ;  $O_{BK} \leftarrow \text{saturate}(O, BK)$ ;  $C \leftarrow \text{newCluster}()$ ;
3: for each  $seed \in O_{BK}$  do
4:   if  $seed$  is UNCLASSIFIED then
5:      $N_{seed} \leftarrow \text{neighborhood}(seed, O_{BK}, R)$ ;
6:     for each  $o \in N_{seed}$  do
7:       if  $o$  is assigned to a cluster different from  $C$  then
8:          $N_{seed} = N_{seed}/o$ ;
9:       end if
10:    end for
11:     $T_{seed} \leftarrow \text{neighborhoodModel}(N_{seed})$ ;
12:    if  $\text{homogeneity}(N_{seed}, T_{seed}) \geq h - threshold$  then
13:       $C.add(seed)$ ;  $seedList \leftarrow \emptyset$ ;
14:      for each  $o \in N_{seed}$  do
15:        if  $o$  in UNCLASSIFIED or  $o$  is NOISE then
16:           $C.add(o)$ ;  $seedList.add(o)$ ;
17:        end if
18:      end for
19:       $\langle C, T_C \rangle \leftarrow \text{expandCluster}(C, seedList, O_{BK}, R, T_{seed}, h - threshold)$ ;
20:       $CLabel = \text{clusterLabel}(T_C)$ ;  $CList.add(\langle C, CLabel \rangle)$ ;  $C \leftarrow \text{newCluster}()$ ;
21:    else
22:       $seed \leftarrow NOISE$ ;
23:    end if
24:  end if
25: end for
26: return  $CList$ ;

```

According to this graph-based formalization, a clustering $\mathbf{C} \subseteq \wp(O)$ is feasible with the discrete structure imposed by R when each cluster $C \in \mathbf{C}$ is a subgraph G_C of the graph $G(N_O, A_R)$ such that for each pair of nodes $\langle n_i, n_j \rangle$ of G_C , n_i is R -reachable from n_j , or vice-versa.

CORSO is a clustering method that both integrates a neighborhood-based graph partitioning to obtain clusters which are feasible with R discrete structure

Algorithm 2 Expand current cluster by merging homogeneous neighborhood.

```

function expandCluster( $C, seedList, O_{BK}, R, T, h - threshold$ )  $\rightarrow \langle C, T \rangle$ ;
2: while ( $seedList$  is not empty) do
     $seed \leftarrow seedList.first()$ ;  $N_{seed} \leftarrow neighborhood(seed, O_{BK}, R)$ ;
4:   for each  $o \in N_{seed}$  do
       if  $o$  is assigned to a cluster different from  $C$  then
6:          $N_{seed} = N_{seed}/o$ ;
       end if
8:   end for
     $T_{seed} \leftarrow neighborhoodModel(N_{seed})$ ;
10:  if  $homogeneity(N_{seed}, \{T, T_{seed}\}) \geq h - threshold$  then
       for each  $o \in N_{seed}$  do
12:          $C.add(o)$ ;  $seedList.add(o)$ ;
       end for
14:    $seedList.remove(seed)$ ;  $T \leftarrow T \cup T_{seed}$ ;
       end if
16: end while
return  $\langle C, T \rangle$ ;

```

and resorts to a multi-relational approach to evaluate similarity among structured objects and form homogeneous clusters (i.e., clusters grouping structured objects sharing a similar relational description according to some similarity criterion). Top-level description of the method is presented in Algorithm 1. The key idea is to exploit the R -neighborhood construction and build clusters feasible with R -discrete structure by merging partially overlapping homogeneous neighborhood units. Cluster construction starts from an arbitrary node n (seed node) of G such that the R -neighborhood $N_R(n)$ is an homogeneous cluster C whose model T_C is a generalization of C and then iteratively expands the cluster in question by including the objects of each R -neighborhood $N_R(n_i)$ of a node $n_i \in G_C$ (neighborhood expansion) when $N_R(n_i)$ is an homogeneous set (see Algorithm 2). The R -neighborhoods involved in the construction of any cluster contain only the objects not yet classified into a different cluster. Moreover, homogeneity within a neighborhood is estimated by comparing each object in the neighborhood with the model of the cluster currently built. Obviously the choice of the seed node affects clustering since neither reassignments nor multiple-cluster assignments are performed.

Although CORSO appears quite similar to GDBSCAN in detecting spatial clusters by exploiting a neighborhood-based partitioning of the graph representing the discrete spatial structure, CORSO differs from GDBSCAN in evaluating homogeneity within a neighborhood to be added to the current cluster. Indeed, GDBSCAN retrieves all spatial objects density-reachable from an arbitrary core object by building successive neighborhoods, but it checks the density within a neighborhood by ignoring the entire cluster. This yields a density-connected set, where density is efficiently estimated independently from the neighborhoods already merged in forming the current cluster. However, this approach may lead

to merge connected neighborhoods sharing some objects but modeling different phenomena. Moreover, GDBSCAN computes density within each neighborhood according to a weighted cardinality function (e.g. aggregation of non spatial values) that assumes single table data representation. CORSO overcomes these limitations by computing density within a neighborhood in terms of degree of similarity among all relationally structured objects falling in the neighborhood with respect to the model of the entire cluster currently built. In particular, following the suggestion given in [10], we evaluate homogeneity within a neighborhood $N_R(n_i)$ to be added to the cluster C as the average degree of matching between objects of $N_R(n_i)$ and the cluster model $T_{C'}$ with $C' = C \cup N_R(n_i)$. Details on cluster model determination, neighborhood homogeneity estimation and cluster labeling are reported below.

2.1 Cluster model generation

Let C be the cluster currently built by merging w neighborhood sets N_1, \dots, N_w , we assume that the cluster model T_C is a set of first-order theories $\{T_1, \dots, T_w\}$ for the concept C where T_i is a model for the neighborhood set N_i . More precisely, T_i is a set of first-order clauses:

$$T_i : \{cluster(X) = c \leftarrow H_{i1}, \dots, cluster(X) = c \leftarrow H_{iz}\},$$

where each H_{ij} is a conjunctive formula describing a sub-structure shared by one or more objects in N_i and $\forall o_i \in N_i, BK \cup T_i \models o_i$, that is, each object in N_i can be explained by the model T_i and the background knowledge BK .

Such model can be learned by resorting to the ILP system ATRE [8] that adopts a separate-and-conquer search strategy to learn a model of structured objects from a set of training examples and eventually counter-examples. In this context, ATRE learns a model for each neighborhood set without considering any counter-examples. To this aim, since only consistent clauses can be induced without negative examples and ATRE searches the hypotheses space until a fixed number of consistent clauses have been generated, the learning parameters are opportunely fixed to make the learner able to generate an adequate number of clauses and to choose the clause covering the highest number of examples and with the highest number of literals in the body. The search of a model starts with the most general clause, that is, $cluster(X) = c \leftarrow$, and proceeds top-down by adding selectors (literals) to the body according to some preference criteria according to which a clause can be considered better than another one (e.g. number of objects covered or number of literals). Selectors involving both numerical and categorical descriptors are handled in the same way, that is, they have to comply with the property of linkedness and are sorted according to preference criteria. The only difference is that selectors involving numerical descriptors are generalized by computing the closed interval that best covers positive examples and eventually discriminates from counter-examples, while selectors involving categorical descriptors with same function value are generalized by simply turning all ground arguments into the corresponding variables and assigning as function value the set containing only the value taken from the input

categorical descriptors. In both cases, the generalization of two selectors is in the form $f(X_1, \dots, X_n) \in v$, where v is an interval or a set.

2.2 Neighborhood homogeneity estimation

The homogeneity h of a neighborhood set N to be added to the cluster C is computed as follows:

$$h(N, T_{C \cup N}) = \frac{1}{\#N} \sum_i h(o_i, T_{C \cup N}) = \frac{1}{\#N} \sum_i \frac{1}{w+1} \sum_j h(o_i, T_j), \quad (1)$$

where $\#N$ is the cardinality of the neighborhood set N and $T_{C \cup N}$ is the cluster model of $C \cup N$ formed by both $\{T_1, \dots, T_w\}$, i.e., the model of C , and T_{w+1} , i.e., the model of N built as explained above. It is clear that, as defined above, the homogeneity of a neighborhood is calculated according to the entire cluster rather than just the neighborhood itself. Since $T_j = \{H_{1j}, \dots, H_{zj}\}$ ($z \geq 1$) with each H_{rj} a conjunctive formula in first-order formalism, we assume that:

$$h(o_i, T_j) = \frac{1}{z} \sum_r fm(o_i, H_{rj}), \quad (2)$$

where fm is a function returning the degree of matching of an object $o_i \in N$ against the conjunctive formula H_{rj} . In this way, the definition of homogeneity of a neighborhood set $N = \{o_1, \dots, o_n\}$ with respect to some logical theory $T_{C \cup N}$ is closely related to the problem of comparing (matching) the conjunctive formula f_i representing an object $o_i \in N^1$ with a conjunctive formula H_{rj} forming the model T_j in order to discover likenesses or differences [12]. This is a directional similarity judgment involving a *referent* R , that is the description or prototype of a class (cluster model) and a *subject* S that is the description of an instance of a class (object to be clustered).

In the classical matching paradigm, the matching of S against R corresponds to compare them just for equality. In particular, when both S and R are conjunctive formulas in first-order formalism, matching S against R corresponds to check the existence of a substitution θ for the variables in R such that $S = \theta(R)$. This last condition is generally weakened by requiring that $S \Rightarrow \theta(R)$, where \Rightarrow is the logical implication. However, the requirement of equality, even in terms of logical implication, is restrictive in presence of noise or variability of the phenomenon described by the referent of matching. Therefore a flexible definition of matching is needed that aims at comparing two descriptions and identifying similarities rather than equalities. The result of such a flexible matching is a number in the interval $[0, 1]$ that is the probability of precisely matching S against R , provided that some change described by θ is possibly made in the description R .

The problem of computing flexible matching to compare structures is not novel. Esposito et al. [4] have formalized a computation schema for flexible

¹ The conjunctive formula f_i is here intended as the description of $o_i \in N$ saturated according to the *BK*.

matching on formulas in first-order formalism whose basic components (selectors) are the relational statements, that is, $f_i(t_1, \dots, t_n) = v$, which are combined by applying different operators such as conjunction (\wedge) or disjunction (\vee) operator. In this work, we focus on the computation of flexible matching $fm(S, R)$ when both S and R are described by conjunctive formulas and $fm(S, R)$ looks for the substitution θ returning the best matching of S against R , as:

$$fm(S, R) = \max_{\theta} \prod_{i=1, \dots, k} fm_{\theta}(S, r_i). \quad (3)$$

The optimal θ that maximizes the above conditional probability is here searched by adopting the branch and bound algorithm that expands the least cost partial path by performing quickly on average [4]. According to this formulation, fm_{θ} denotes the flexible matching with the tie of the substitution fixed by θ computed on each single selector $r_i \equiv f_{r_i}(t_{r_1}, \dots, t_{r_n}) = v_{r_i}$ of the referent R where f_{r_i} is a function descriptor with either numerical (e.g. area or distance) or categorical (e.g. intersect) range. In the former case the function value v_{r_i} is an interval value ($v_{r_i} \equiv [a, b]$), while in the latter case v_{r_i} is a subset of values ($v_{r_i} \equiv \{v_1, \dots, v_M\}$) from the range of f_{r_i} . This faces with a referent R that is obtained by generalizing a neighborhood of objects in O . Conversely for the subject S , that is, the description of a single object $o \in O$, the function value w_{s_j} assigned to each selector $s_j \equiv f_{s_j}(t_{s_1}, \dots, t_{s_n}) = w_{s_j}$ is an exactly known single value from the range of f_{s_j} . In this context, the flexible matching $fm_{\theta}(S, r_i)$ evaluates the degree of similarity $fm(s_j, \theta(r_i))$ between $\theta(r_i)$ and the corresponding selector s_j in the subject S such that both r_i and s_j have the same function descriptor $f_r = f_s$ and for each pair of terms $\langle t_{r_i}, t_{s_i} \rangle$, $\theta(t_{r_i}) = t_{s_i}$. More precisely,

$$fm(s_j, \theta(r_i)) = fm(w_{s_j}, v_{r_i}) = \max_{v \in v_{r_i}} P(equal(w_{s_j}, v)). \quad (4)$$

The probability of the event $equal(w_{s_j}, v)$ is then defined as the probability that an observed w_{s_j} is a distortion of v (i.e. w_{s_j} is the observed value different from the real value v because of the presence of noise), that is:

$$P(equal(w_{s_j}, v)) = P(\delta(X, v) \geq \delta(w_{s_j}, v)) \quad (5)$$

where X is a random variable assuming value in the domain D representing the range of f_r while δ is a distance measure. The computation of $P(equal(w_{s_j}, v))$ clearly depends on the probability density function of X . For categorical descriptors, that is, D is a discrete set with cardinality $\#D$, it has been proved [4] that:

$$P(equal(w_{s_j}, v)) = \begin{cases} 1 & \text{if } w_{s_j} = v \\ (\#D - 1) / \#D & \text{otherwise} \end{cases} \quad (6)$$

when X is assumed to have a uniform probability distribution on D and $\delta(x, y) = 0$ if $x = y$, 1 otherwise. Although similar results have been reported for both linear non numerical and tree-structured domains, no result appears for numerical

domains. Therefore, we have extended definitions reported in [4] to make flexible matching able to deal with numerical descriptors and we have proved that:

$$fm(c, [a, b]) = \begin{cases} 1 & \text{if } a \leq c \leq b \\ 1 - 2(a - c)/(\beta - \alpha) & \text{if } c < a \wedge 2a - c \leq \beta \\ (c - \alpha)/(\beta - \alpha) & \text{if } c < a \wedge 2a - c > \beta \\ (\beta - c)/(\beta - \alpha) & \text{if } c > b \wedge 2b - c < \alpha \\ 1 - 2(c - b)/(\beta - \alpha) & \text{if } c > b \wedge 2b - c \geq \alpha \end{cases} \quad (7)$$

by assuming that X has uniform distribution on $D \equiv [\alpha, \beta]$ and $\delta(x, y) = |x - y|$. A proof of formula 7 is reported in the Appendix A of this paper.

2.3 Cluster labeling

A cluster C can be naturally labeled with T_C that is the set of first-order clauses obtained from the generalization of neighborhoods merged in C . Each first-order clause is in the form $C \leftarrow s_1, \dots, s_n$, where C represents the cluster label and each s_i denotes a selector in the form $f_i(X_{i_1}, \dots, X_{i_l}) \in v_i$ with v_i an interval value or a set value. In this formalization, two selectors $s_1 : f_1(X_{1_1}, \dots, X_{1_l}) \in v_1$ and $s_2 : f_2(X_{2_1}, \dots, X_{2_l}) \in v_2$ are comparable according to some substitution θ when they involve the same descriptor ($f_1 = f_2 = f$) and each pair of terms $\langle X_{1_i}, X_{2_i} \rangle$ is unifiable according to θ , i.e., $X_{1_i}\theta = X_{2_i}\theta = X_i$ ($\forall i = 1 \dots l$). In this case, the selector $s : f(X_{1_1}, \dots, X_{1_l}) \in v_1 \cup v_2$ is intended as a generalization for both s_1 and s_2 . In particular, the selectors s_1 and s_2 are equal when they are comparable and $v_1 \equiv v_2 \equiv v$. In this case the generalization of s_1 and s_2 is built as $s : f(X_{1_1}, \dots, X_{1_l}) \in v$. Similarly, the selector s_1 (s_2) is contained in the selector s_2 (s_1) when they are comparable and $v_1 \subseteq v_2$ ($v_2 \subseteq v_1$), while the generalization s is $f(X_{1_1}, \dots, X_{1_l}) \in v_2$ ($f(X_{1_1}, \dots, X_{1_l}) \in v_1$). Note that equality of selectors implies containment, but not vice-versa. Similarly, the first-order clauses $H_1 : C \leftarrow s_{1_1}, \dots, s_{1_n}$ and $H_2 : C \leftarrow s_{2_1}, \dots, s_{2_n}$ are comparable according to some substitution θ when each pair of selectors $\langle s_{1_i}, s_{2_i} \rangle$ is comparable according to θ . Hence, H_1 is equal (contained) to H_2 when s_{1_i} is equal (contained) to s_{2_i} for each $i = 1, \dots, n$. In both these cases (equality and containment condition), the pair of clauses H_1, H_2 can be replaced without loss of information with the clause H that is the generalization of H_1, H_2 built by substituting each pair of comparable selectors $\langle s_{1_i}, s_{2_i} \rangle \in \langle H_1, H_2 \rangle$ with the generalization obtained as stated above. This suggests the idea of merging a pair of comparable clauses H_1, H_2 in a single clause H by preserving the equivalence of coverage, that is:

- if $H_1, H_2, BK \models o$ then $H, BK \models o$ and vice-versa,
- if $H_1, H_2, BK \not\models o$ then $H, BK \not\models o$ and vice-versa,

where o is a structured object and BK is a set of first-order clauses. The equivalence of coverage between $\{H_1, H_2\}$ and H is obviously guaranteed when H_1 is either equal or contained in H_2 or vice-versa, but this equivalence cannot be guaranteed when H_1 and H_2 are comparable first-order clauses but neither equality condition nor containment condition are satisfied.

Example 1: Let us consider the pair of comparable first-order clauses:

$$H_1 : cluster(X_1) = c \leftarrow distance(X_1, X_2) \in [5..10], type(X_2) \in \{street\}$$

$$H_2 : cluster(X_1) = c \leftarrow distance(X_1, X_2) \in [3..7], type(X_2) \in \{river\}$$

where neither H_1 is equal to H_2 nor $H_1(H_2)$ is contained in $H_2(H_1)$. The first-order clause obtained by generalizing pairs of comparable selectors in both H_1 and H_2 , is:

$$H : cluster(X_1) = c \leftarrow distance(X_1, X_2) \in [3..10], type(X_2) \in \{street, river\}$$

where $H \models o$ with $o : distance(X_1, X_2) = 3 \wedge type(X_2) = street$, but neither $H_1 \models o$ nor $H_2 \models o$.

The requirement of equality between H_1 and H_2 can be relaxed while preserving equivalence of coverage with respect to the generalization H . Indeed, when

$$H_1 : C \leftarrow s_1(-) \in v_1, \dots, s_k(-) \in v_k, \dots, s_n(-) \in v_n$$

$$H_2 : C \leftarrow s_1(-) \in v_1, \dots, s_k(-) \in w_k, \dots, s_n(-) \in v_n$$

are comparable first-order clauses differing only in the function value of a single selector (i.e. s_k), the first-order clause:

$$H : C \leftarrow s_1(-) \in v_1, \dots, s_k(-) \in v_k \cup w_k, \dots, s_n(-) \in v_n$$

continues to preserve the equivalence of coverage with $\{H_1, H_2\}$.

Example 2: Let us consider the pair of comparable first-order clauses:

$$H_1 : cluster(X_1) = c \leftarrow distance(X_1, X_2) \in [3..7], type(X_2) \in \{street\},$$

$$length(X_2) \in [3, 5]$$

$$H_2 : cluster(X_1) = c \leftarrow distance(X_1, X_2) \in [3..7], type(X_2) \in \{street\},$$

$$length(X_2) \in [7, 10]$$

which differ only in the value of a single selector (length), the first-order clause obtained by generalizing the pairs of comparable selectors in both H_1 and H_2 is $H : cluster(X_1) = c \leftarrow distance(X_1, X_2) \in [3..7], type(X_2) \in \{street\}, length(X_2) \in [3, 5] \cup [7, 10]$ that is equivalent in coverage to the pair $\{H_1, H_2\}$.

Following this idea, it is possible to compactly describe the cluster theory T_C finally associated to a cluster C by iteratively replacing pairs of comparable first-order clauses H_1, H_2 with the generalization H , when H results equivalent in coverage to $\{H_1, H_2\}$ (see Algorithm 3).

Example 3: Let us consider T_C that is the set of first-order clauses including:

$$H_1 : cluster(X_1) = c \leftarrow distance(X_1, X_2) \in [5..10], color(X_2) \in \{red\}$$

$$H_2 : cluster(X_1) = c \leftarrow distance(X_1, X_2) \in [5..6], color(X_2) \in \{blue\}$$

$$H_3 : cluster(X_1) = c \leftarrow distance(X_1, X_2) \in [5..10], color(X_2) \in \{blue\}$$

$$H_4 : cluster(X_1) = c \leftarrow distance(X_1, X_2) \in [6..10], area(X_2) \in [30..40]$$

T_C can be transformed in the set of first-order clauses:

$$H'_1 : cluster(X_1) = c \leftarrow distance(X_1, X_2) \in [5..10], color(X_2) \in \{red, blue\}$$

$$H'_2 : cluster(X_1) = c \leftarrow distance(X_1, X_2) \in [6..10], area(X_2) \in [30..40]$$

where H'_1 results by firstly merging H_1 and H_3 , which are comparable and differ only in the function value of a selector ($color(X_2) \in \{red\}$ vs $color(X_2) \in \{blue\}$), and obtaining $H_{13} : cluster(X_1) = c \leftarrow distance(X_1, X_2) \in [5..10], color(X_2) \in \{red, blue\}$ and then merging H_{13} and H_2 since H_2 is contained in H_{13} .

Algorithm 3 Build a compact theory to describe a cluster C .

```
1: function clusterLabel( $T_C$ )  $\rightarrow T'_C$ ;  
2:  $T'_C \leftarrow \emptyset$ ;  $merge \leftarrow false$ ;  
3: while  $T_C$  is not empty do  
4:    $H$  is a first-order clause in  $T_C$ ;  $T_C = T_C/H$ ;  
5:   for each  $H' \in T_C$  do  
6:     if  $H$  and  $H'$  are generalizable without lost of information then  
7:        $H = generalize(H, H')$ ;  $T_C = T_C/H'$ ;  $merge = true$ ;  
8:     end if  
9:   end for  
10:   $T'_C = T'_C \cup H$ ;  
11: end while  
12: if  $merge$  is  $true$  then  
13:   $T'_C \leftarrow clusterLabel(T'_C)$ ;  
14: end if  
15: return  $T'_C$ ;
```

3 The Application: two case studies

In this section, we describe the application of CORSO to two distinct real-world problems, namely topographic map interpretation and geo-referenced census data analysis. In the former problem, a topographic map is treated as a grid of square cells of same size, according to a hybrid tessellation-topological model such that adjacency among cells allows map-reading from a cell to one of its neighbors in the map. For each cell, geographical data is represented by means of a geometric (or physical) representation, describing the physical entities (point, line or region) corresponding to the geographical objects, and a thematic (or logical) representation, expressing the corresponding semantics (e.g., hydrography, vegetation and so on). Spatial clustering in this case aims at identifying a mosaic of nearly homogeneous clusters (areas) including adjacent cells such that geographical data inside each cluster properly models the spatial continuity of some morphological environment, while separate clusters model spatial variation over the entire space. In the second problem, the goal is to perform a joint analysis of both socio-economic factors represented in census data and geographical factors represented in topographic maps. The discovery of homogeneous areal clusters on spatially distributed socio-economic phenomena (e.g. social and economical deprivation) can be a valuable support to good public policy. In this case, spatial objects are territorial units for which census data are collected as well as entities of geographical layers such as urban and wood areas. In both applications, running time of CORSO refers to execution performed on a 2 Ghz IBM notebook with Windows XP and 256 Mb of RAM.

3.1 Topographic map interpretation

In this study we discuss an application of spatial clustering to characterize spatial continuity of some morphological elements over the topographic map of Canosa

(Apulia, Italy). The examined area covers 45 km² and is segmented into square areal units of 1 Km² each. Thus, the problem of recognizing spatial continuity of morphological elements in the map is formulated as the problem of grouping adjacent cells resulting in a morphologically homogeneous area. The discrete spatial structure is then imposed by the relation of “adjacency” among cells.

Since several geographical objects (e.g., almond tree, olive tree, fountain, etc) are collected within each cell, we apply algorithms derived from geometrical and topological reasoning [9] to obtain cell descriptions in first-order formalism. We consider descriptions including spatial descriptors encompassing geometrical properties (e.g., *area* and *extension*) and topological relations (e.g., *regionToRegion*, *lineToLine*, *pointToRegion*) as well as non spatial descriptors (e.g., *typeOf* and *subtypeOf*). The descriptor *partOf* is used to define the physical structure of a logical object. An example is: $typeOf(f_1) = fountain \wedge partOf(f_1, x_1) = true$, where f_1 denotes a fountain which is physically represented by a point referred with the constant x_1 . Each cell is here described by a conjunction of 946.866 ground selectors in average. To support some qualitative reasoning, a BK is expressed in form of clauses. An example of BK is:

$fountainToCulture(Font, Culture) = Relation \leftarrow typeOf(Font) = fountain,$
 $partOf(Font, Point) = true, typeOf(Culture) = culture,$
 $partOf(Culture, Region) = true, pointToRegion(Point, Region) = Relation$

that allows to move from a physical to a logical level in describing the topological relation between the point that physically represents the fountain and the region that physically represents the culture and that are, respectively, referred to as the variables *Font* and *Culture*. The goal is to model the spatial continuity of some morphological environment (e.g. cultivation setting) within adjacent cells over the map. It is noteworthy that granularity of partitioning changes by varying homogeneity threshold (see Figure 1). In particular, when $h - threshold = 0.95$, CORSO clusters adjacent cells in five regions in 1821 secs. Each cluster is compactly labeled as follows:

- $C_1 : cluster(X_1) = c_1 \leftarrow containAlmondTree(X_1, X_2) \in \{true\},$
 $cultivationToCulture(X_2, X_3) \in \{outside\},$
 $areaCulture(X_3) \in [328..420112], fountainToCulture(X_4, X_3) \in \{outside\}.$
- $C_2 : cluster(X_1) = c_2 \leftarrow containAlmondTree(X_1, X_2) \in \{true\},$
 $cultivationToCulture(X_2, X_3) \in \{inside\}, areaCulture(X_3) \in [13550..$
 $187525], cultivationToCulture(X_2, X_4) \in \{outside\}.$
- $C_3 : cluster(X_1) = c_3 \leftarrow containGrapevine(X_1, X_2) \in \{true\},$
 $cultivationToCulture(X_2, X_3) \in \{inside\}, areaCulture(X_3) \in [13550..$
 $212675], cultivationToCulture(X_2, X_4) \in \{outside\}.$
 $cluster(X_1) = c_3 \leftarrow containGrapevine(X_1, X_2) \in \{true\},$
 $cultivationToCulture(X_2, X_3) \in \{outside\}, areaCulture(X_3) \in [150..$
 $212675], cultivationToCulture(X_2, X_4) \in \{outside, inside\}.$
- $C_4 : cluster(X_1) = c_4 \leftarrow containStreet(X_1, X_2) \in \{true\}$
 $streetToCulture(X_2, X_3) \in \{adjacent\}, areaCulture(X_3) \in [620..$
 $230326], cultureToCulture(X_3, X_4) \in \{outside, inside\}.$
- $C_5 : cluster(X_1) = c_5 \leftarrow containOliveTree(X_1, X_2) \in true,$

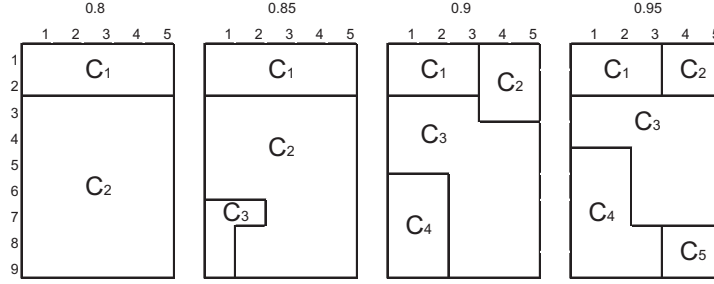


Fig. 1. Spatial clusters detected on map data from the zone of Canosa by varying h – *threshold* value in $\{0.8, 0.85, 0.9, 0.95\}$.

$cultivationToCulture(X_2, X_3) \in \{outside\}$, $areaCulture(X_3) \in [620..144787]$, $cultivationToCulture(X_2, X_4) \in \{outside\}$.

Notice that each detected cluster effectively includes adjacent cells sharing a similar morphological environment, while separate clusters describe quite different environments.

3.2 Geo-referenced census data analysis

In this application, we consider both census and digital map data concerning North West England (NWE) area that is decomposed into censual sections or wards for a total of 1011 wards. Census data is available at ward level and provides some measures of deprivation level in the ward according to index scores that combine information provided by 1998 Census. These scores are used in the evaluation of the need for primary care, in health-related analysis and in targeting urban regeneration funds. The higher the index value the more deprived a ward is. We focus attention on investigating continuity of socio-economic deprivation joined to geographical factors represented in linked topographic maps.

Both ward-referenced census data and map data are stored in Oracle Spatial 9i database as a set of spatial tables, one for each layer. Each spatial table includes a geometry attribute that allows storing the geometrical representation and the positioning of a spatial object with respect to some reference system. We adopt a topological algorithm based on the 9-intersection model [3] to detect both adjacency relation between NWE wards (i.e. wards which share some boundary) and overlapping relation between wards and urban areas (or woods). The former imposes a discrete spatial structure over NWE wards such that only adjacent wards may be grouped in the same cluster while the latter contributes to define the spatial structure embedded in each ward not only in terms of observed values of deprivation scores but also extension of urban areas and/or woods overlapping each ward. No BK is defined for this problem.

Granularity of partitioning changes when varying the value of h – *threshold*, that is, CORSO detects 79 clusters with h – *threshold* = 0.80, 89 clusters with h – *threshold* = 0.85, 122 clusters with h – *threshold* = 0.90 and 163 clusters

with $h - threshold = 0.95$. In particular, when $h - threshold = 0.95$, CORSO clusters NWE area in 2160 secs and identifies adjacent regions modeling differently relational patterns involving deprivation and geographical environment. For instance, by analyzing these spatial clusters, we discover three adjacent areas, namely C_1 , C_2 and C_3 compactly labeled as follows:

C_1 : $cluster(X_1) = c_1 \leftarrow townsend(X_1) \in [-4.7.. - 0.6]$,
 $doe(X_1) \in [-12.4..2.7]$, $carstairs(X_1) \in [-4.5.. - 0.9]$,
 $jarman(X_1) \in [-32.7..7.5]$, $overlapped_by_wood(X1, X2) \in \{true\}$.
 $cluster(X_1) = c_1 \leftarrow townsend(X_1) \in [-5.4.. - 2.3]$,
 $doe(X_1) \in [-10.9.. - 0.5]$, $carstairs(X_1) \in [-4.2.. - 1.6]$,
 $jarman(X_1) \in [-22.8..0.6]$, $overlapped_by_wood(X1, X2) \in \{true\}$.
 $cluster(X_1) = c_1 \leftarrow townsend(X_1) \in [-5.4.. - 3.2]$,
 $doe(X_1) \in [-8.8.. - 2.1]$, $carstairs(X_1) \in [-4.4.. - 2.5]$,
 $jarman(X_1) \in [-22.8.. - 2.4]$, $overlapped_by_wood(X1, X2) \in \{true\}$.

C_2 : $cluster(X_1) = c_1 \leftarrow townsend(X_1) \in [-2.0..0.6]$,
 $doe(X_1) \in [-4.2..1.6]$, $carstairs(X_1) \in [-2.6..2.1]$,
 $jarman(X_1) \in [-9.7..8.8]$, $overlapped_by_largeUrbArea(X1, X2) \in \{true\}$.
 $cluster(X_1) = c_1 \leftarrow townsend(X_1) \in [-2.7..2.8]$,
 $doe(X_1) \in [-4.2..4.0]$, $carstairs(X_1) \in [-2.2..2.7]$,
 $jarman(X_1) \in [-8.8..21.3]$, $overlapped_by_largeUrbArea(X1, X2) \in \{true\}$

C_3 : $cluster(X_1) = c_1 \leftarrow townsend(X_1) \in [-3.4..0.4]$,
 $doe(X_1) \in [-8.2.. - 0.2]$, $carstairs(X_1) \in [-3.7..0.6]$,
 $jarman(X_1) \in [-27.7.. - 1.5]$,
 $overlapped_by_smallUrbArea(X1, X2) \in \{true\}$.

C_1 , C_2 and C_3 cover adjacent areas with quite similar range value for deprivation indexes but C_1 models the presence of woods while C_2 and C_3 model the presence of small urban areas and large urban areas, respectively. Discontinuity of geographical environments modeled by these clusters is confirmed by visualizing map data about the area (see Figure 2).

4 Conclusions

This paper presents a novel approach to discover clusters from structured spatial data taking into account relational constraints (e.g. spatial correlation) forming the discrete spatial structure representable as a graph. In this way, the concept of graph neighborhood is exploited to capture relational constraints embedded in the graph edges. Moreover, we resort to a relational approach to mine data scattered in multiple relations describing the structure that is naturally embedded in spatial data. As a consequence, only spatial units associated with (transitively) graph connected nodes can be clustered together according to judgment of similarity on relational descriptions representing their internal (spatial) structure. As future work, we intend to investigate the possibility of assigning the same unit to multiple clusters. In addition, we plan to employ CORSO in analyzing geo-referenced data of air pollution in order to support a good policy of environmental planning.

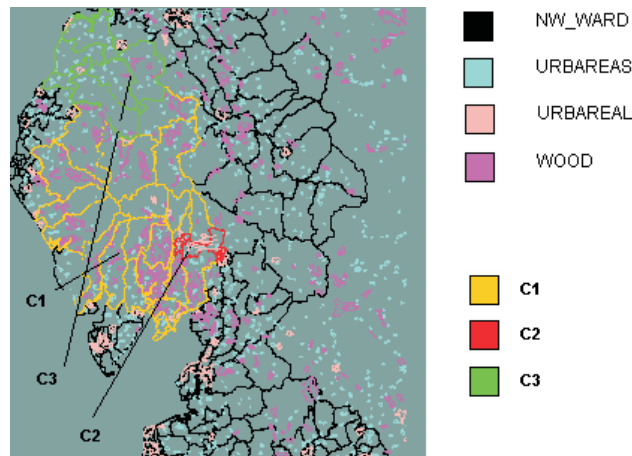


Fig. 2. Spatial clusters detected on NWE with h - threshold = 0.95.

5 Acknowledgment

The work presented in this paper is partial fulfillment of the research objective set by the ATENEO-2004 project on “Metodi di Data Mining Multi-relazionale per la scoperta di conoscenza in basi di dati”.

References

1. H. Blockeel. *Top-down induction of first order logical decision trees*. PhD thesis, Department of Computer Science, Katholieke Universiteit, Leuven, Belgium, 1998.
2. S. Džeroski and N. Lavrač. *Relational Data Mining*. Springer-Verlag, 2001.
3. M. Egenhofer. Reasoning about binary topological relations. In *Symposium on Large Spatial Databases*, pages 143–160, 1991.
4. F. Esposito, D. Malerba, and G. Semeraro. Flexible matching for noisy structural descriptions. In *International Joint Conference on Artificial Intelligence*, pages 658–664, 1991.
5. M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Knowledge Discovery in Databases*, pages 226–231, 1996.
6. L. Kaufmann and P. J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley, 1990.
7. M. Kirsten and S. Wrobel. Relational distance-based clustering. In *Inductive Logic Programming, 8th International Conference*, volume 1446, pages 261–270. Springer-Verlag, 1998.
8. D. Malerba. Learning recursive theories in the normal ilp setting. *Fundamenta Informaticae*, 57(1):39–77, 2003.
9. D. Malerba, F. Esposito, A. Lanza, F. A. Lisi, and A. Appice. Empowering a gis with inductive learning capabilities: The case of ingens. *Journal of Computers, Environment and Urban Systems, Elsevier Science*, 27:265–281, 2003.

10. D. Mavroeidis and P. Flach. Improved distances for structured data. In T. Horváth and A. Yamamoto, editors, *Inductive Logic Programming, 13th International Conference*, volume 2835, pages 251–268. Springer-Verlag, 2003.
11. R. T. Ng and J. Han. Efficient and effective clustering methods for spatial data mining. In J. Bocca, M. Jarke, and C. Zaniolo, editors, *Very Large Data Bases, 20th International Conference*, pages 144–155. Morgan Kaufmann Publishers, 1994.
12. D. Patterson. *Introduction to Artificial Intelligence and expert systems*. Prentice-Hall, 1991.
13. L. D. Raedt and S. Dzeroski. First-order jk-clausal theories are pac-learnable. *Artificial Intelligence*, 70(1-2):375–392, 1994.
14. J. Sander, E. Martin, H.-P. Kriegel, and X. Xu. Density-based clustering in spatial databases: The algorithm gbscan and its applications. *Data Mining and Knowledge Discovery*, 2(2):169–194, 1998.
15. G. Toussaint. Some unsolved problems on proximity graphs. In D. Dearholt and F. Harary, editors, *First Workshop on Proximity Graphs*, 1991.
16. X. Wang and H. J. Hamilton. Dbrs: A density-based spatial clustering method with random sampling. In *PAKDD*, pages 563–575, 2003.

A Appendix

Let us recall definitions (4) and (5) and apply them to numerical case. We have:

$$fm(c, [a, b]) = \max_{v \in [a, b]} P(\text{equal}(c, v)) = \max_{v \in [a, b]} P(\delta(X, v) \geq \delta(c, v))$$

By assuming that X has a uniform distribution on domain $D = [\alpha, \beta]$ with density function $f_D(x) = 1/(\beta - \alpha), \forall x \in D$ and fixing $\delta(x, y) = |x - y|$, $P(\delta(X, v) \geq \delta(c, v))$ can be rewritten as $P(|X - v| \geq |c - v|)$ that is maximized when minimizing $|c - v|$.

If $a \leq c \leq b$ then $\max_{v \in [a, b]} P(|X - v| \geq |c - v|) = P(|X - v| \geq |c - c|) = 1$.

If $c < a$ then $\max_{v \in [a, b]} P(|X - v| \geq |c - v|)$ is written as $\max_{v \in [a, b]} P(|X - v| \geq v - c)$.

Since the maximum of $P(|X - v| \geq v - c)$ is obtained for $v = a$, we have that $\max_{v \in [a, b]} P(|X - v| \geq v - c) = P(|X - a| \geq a - c) = P(X - a \geq a - c) + P(X - a \leq c - a) = P(X \geq 2a - c) + P(X \leq c)$ where:

1. $P(X \geq 2a - c) = \int_{\beta}^{2a-c} 1/(\beta - \alpha) dx = (\beta - 2a + c)/(\beta - \alpha)$ if $2a - c \leq \beta$, 0 otherwise;
2. $P(X \leq c) = (c - \alpha)/(\beta - \alpha)$.

Hence, we obtain that:

$$\max_{v \in [a, b]} P(|X - v| \geq v - c) = \begin{cases} 1 - 2(a - c)/(\beta - \alpha) & \text{if } c < a \wedge 2a - c \leq \beta \\ (c - \alpha)/(\beta - \alpha) & \text{if } c < a \wedge 2a - c > \beta \end{cases}$$

If $c > b$ then $\max_{v \in [a, b]} P(|X - v| \geq |c - v|)$ can be equivalently written as $\max_{v \in [a, b]} P(|X - v| \geq c - v)$ that is obtained for $v = b$. Therefore, $\max_{v \in [a, b]} P(|X - v| \geq c - v) = P(|X - b| \geq c - b) = P(X - b \geq c - b) + P(X - b \leq b - c) = P(X \geq c) + P(X \leq 2b - c)$. We have that:

$$\max_{v \in [a, b]} P(|X - v| \geq c - v) = \begin{cases} (\beta - c)/(\beta - \alpha) & \text{if } c > b \wedge 2b - c < \alpha \\ 1 - 2(c - b)/(\beta - \alpha) & \text{if } c > b \wedge 2b - c \geq \alpha \end{cases}$$