Fire! Firing Inductive Rules from Economic Geography for Fire Risk Detection

David Vaz[1], Vítor Santos Costa[2], and Michel Ferreira[3]

[1] LIACC and DCC/FCUP, University of Porto, Portugal
[2] CRACS-INESC Porto LA and DCC/FCUP, University of Porto, Portugal
[3] IT and DCC/FCUP, University of Porto, Portugal

Abstract. Wildfires can importantly affect the ecology and economy of large regions of the world. Effective prevention techniques are fundamental to mitigate their consequences. The design of such preemptive methods requires a deep understanding of the factors that increase the risk of fire, particularly when we can intervene on these factors. This is the case for the maintenance of ecological balances in the landscape that minimize the occurrence of wildfires. We use an inductive logic programming approach over detailed spatial datasets: one describing the landscape mosaic and characterizing it in terms of its use; and another describing polygonal areas where wildfires took place over several years. Our inductive process operates over a logic term representation of vectorial geographic data and uses spatial predicates to explore the search space, leveraging the framework of Spatial-Yap, its multi-dimensional indexing and tabling extensions. We show that the coupling of a logic-based spatial database with an inductive logic programming engine provides an elegant and powerful approach to spatial data mining.

1 Introduction

Wildfires are an unavoidable event in Nature and play an important role in wildland ecosystems. Naturally caused wildfires are, however, a small percentage of all the wildland fires. Preventing and mitigating the consequences of wildfires that result from the increased pressure of human activity in wildland areas has been the goal of fire control programs for more than a century. Prevention techniques range from measures aiming to reduce human infractions, to the altering of stored fuels, through controlled burns, in wildlands to affect future fire risk and behavior. In addition to the straightforward impact of fuels and weather conditions in the occurrence of fires, the role of topography is also relevant. Here we understand topography in a broader sense, as a discipline concerned with local detail of space, including not only relief but also vegetation and human-made features. In areas where human intervention has importantly reshaped this topography, as happens in many European regions where native forests have been replaced by fast-growing trees and pasture areas, the impact of this human-designed organization of landscape can potentially affect the occurrence and behavior of wildfires. In this paper we focus on this landscape organization factor, which has been given little attention by fire control programs. While some aspects of the human-made organization of landscape can obviously affect fire behavior, such as the existence of major roads cutting through forest areas, that act as barriers to the propagation of fire, there are also potentially less obvious correlations between this local organization of the landscape and the occurrence of fires, which can profit from machine learning techniques. Understanding such correlations can then guide the human intervention in the landscape towards more efficient prevention and mitigation of the consequences of wildfires.

In this paper we propose the use of inductive logic programming (ILP) to design logic theories that correlate the local organization of the landscape with the occurrence of fires, using the framework of Spatial-Yap [1] to have term-based representation of vectorial geographic data. In addition, Spatial-Yap provides a logic-based approach to a geographic information system and is able to render the inductive engine with spatial relationship predicates that are used to formulate theories *on-the-fly* based on spatial reasoning. We use detailed spatial databases that contain vectorial representations of the landscape organization in the north of Portugal. We overlay these databases with another spatial database that keeps historical records of wildfires taking place over several years in the same region.

The remainder of this paper is organized as follows. The next section presents related work on the use of ILP with spatial datasets. Section 3 describes our fire dataset and the methodology we used. In Section 4 we report preliminary results, together with a discussion of these results. Finally, Section 5 ends the paper.

2 Related Work

Over the past years, the use of spatial data has increased in many areas of computer science. Relational Database Management Systems (RDBMS) were among the first systems to tackle this kind of data, both through extensions to support spatial data, and by providing functions to manipulate the data.

The Open Geospatial Consortium (OGC) proposed a standard to extend SQL-92 in "OpenGis Simple Features Specification for SQL" (OGC99) [2]. The purpose of this specification is to define a standard SQL schema that supports storage, retrieval, query and update of simple geospatial feature collections. Examples of RDBMS systems that conform to this standard are Oracle Spatial and PostgreSQL, through the PostGIS module.

The development of such sophisticated geographical databases has led to interest in *spatial data mining*, defined to be the branch of data-mining where the spatial neighbors of an object may have an influence on the object [3]. A typical task would be to find clusters of correlated objects [4], but a large number of different applications are possible.

Arguably, spatial learning can be considered as an instance of multi-relational learning, with a very specific type of domain knowledge, and should be an important application for ILP. Malerba [3] and his group have exploited this approach with very interesting results. In their approach, multi-relational data-mining techniques are applied by working at a higher conceptual level of the geographic information [5]. Their approach follows a two step algorithm. First, system such as INGENS [6] extract relevant concepts and features from a spatial database, by applying and expanding on standard GIS tools. Second, this relational representation of spatial data can be mined by ILP techniques: ATRE [7] implements a sequence coverage algorithm that learns a classifier, and SPADA [8] is an association-rule learner that can find strong spatial association rules.

The INGENS work raises a number of interesting questions. One important problem, discussed by Malerba [5], is the computational cost of performing feature extraction: although spatial facts are rarely updated, attribute expansion can be expensive in terms of time and space, with often time being spent computing unnecessary attributes. One would expect this problem to grow as databases grow in size and complexity.

One possible approach to this problem is to couple a database to a deductive system: MYDDAS [9] couples YapTab [10] and MySQL extended with geometry types to form Spatial-Yap [1] (unfortunately MySQL has never evolved to conform with OGC99). In this paper we take the next step and actually couple tightly Prolog inference with the geographical data itself. In order to perform inference with logic programming, we need to address well the three key components of geographical data-mining:

1. Spatial terms for representing and storing spatial objects.
2. Spatial predicates, to manipulate (e.g., intersect two spatial terms) and to query spatial terms toward finding interesting properties such as area or distance between two spatial terms.
3. Effective indexing of spatial terms, not only because of the usual mammoth size of such terms, but also because of the number of different terms in the database and the complexity of spatial predicates.

We address the first problem by simply using Prolog terms to represent the three main geometry types, as they are presented in OGC99 standard: *Point*, *Linestring* and *Polygon*; and collection types. We support the OGC defined attributes and restrictions, as can be consulted in the OGC99 document [2]. Our representation is thus similar to the Well Known Text of OGC99, with spatial terms conforming with the grammar in Figure 1.

We further define a set of spatial predicates that provide an interface to the GEOS API [11], that conforms to the OGC99 standard and is also used by PostGIS. Table 1 summarizes these predicates.

This machinery provides the foundation for a logic programming geographical information system. The next step was based on the observation that spatial data does not benefit from most of the traditional indexing techniques (namely the ones used in the logic programming), as most of them are based on single dimension indexing structures.The RDBMS community addressed this problem by proposing novel data-structures, namely R-Trees which have become standard [12].

We extended Prolog indexing through *User Defined Indexing* (UDI) [13], a new extension to Prolog indexing where the programmer is able to define the indexing mechanism based on *what* the terms in the arguments of a predicate

```
SpatialTerm = Point | LineString | Polygon
            | MultiPoint | MultiLineString | MultiPolygon | GeometryCollection ;
Point = "point" PointTerm ;
LineString = "linestring(" PointTermList ")" ;
Polygon = "polygon(" PointTermListList ")" ;
MultiPoint = "multipoint(" PointTermList ")" ;
MultiLinestring = "multilinestring(" PointTermListList ")" ;
MultiPolygon = "multipolygon(" PointTermListListList ")" ;
GeometryCollection = "geometrycollection(" SpatialTermList ")" ;
PointTerm = "(" Number "," Number ")" ;
```

Fig. 1. EBNF of Spatial Terms.

Type	**Predicate**	
Predicates for testing spatial properties	`ogc_is_empty(+Geom)`	
	`ogc_is_simple(+Geom)`	
	`ogc_equals(+Geom1,+Geom2)`	
	`ogc_disjoint(+Geom1,+Geom2)`	
	`ogc_touches(+Geom1,+Geom2)`	
	`ogc_within(+Geom1,+Geom2)`	
	`ogc_overlaps(+Geom1,+Geom2)`	
	`ogc_crosses(+Geom1,+Geom2)`	
	`ogc_intersects(+Geom1,+Geom2)`	
	`ogc_contains(+Geom1,+Geom2)`	
	`ogc_relate(+Geom1,+Geom2,?PatternMatrix)`	
Predicates that support spatial analysis	`ogc_envelope(+Geom,?GeomEnvelope)`	
	`ogc_boundary(+Geom,?GeomBoundary)`	
	`ogc_buffer(+Geom,+Distance,?GeomBuffer)`	
	`ogc_convex_hull(+Geom,?GeomConvexHull)`	
	`ogc_intersection(+Geom1,+Geom2,?GeomIntersection)`	
	`ogc_union(+Geom1,+Geom,?GeomUnion)`	
	`ogc_difference(+Geom1,+Geom,?GeomDifference)`	
	`ogc_symmetric_difference(+Geom1,+Geom2,?GeomSymDiff)`	
	`ogc_distance(+Geom1,+Geom2,?Distance)`	
Specific type predicates	Linestring Multilinestring	`ogc_length(+Geom,?Length)`
		`ogc_is_closed(+Geom)`
		`ogc_is_ring(+Geom)`
	Polygon Multipolygon	`ogc_area(+Geom,?Area)`
		`ogc_centroid(+Geom,?GeomPoint)`
		`ogc_point_on_surface(+Geom,?GeomPoint)`

Table 1. Spatial Predicates.

are meant to represent. This allows users to provide an indexing function that selects a subset of the clauses of a predicate, given a set of constrained variables or bound Prolog terms.

In this work, we use spatial terms [1]. As discussed above, these are simple geometry types based on 2D points. Notice that the simplicity of the primitives does not mean that the terms themselves are simple. For example, the polygons shown in the figures of this paper are represented in Prolog with several hundred points each.

The key idea in R-Trees is to use the Minimum Bounding Rectangle (MBR) to index data. Each leaf nodes stores an object (or at least a pointer), and is keyed by the object's MBR. Inner nodes are keyed by an MBR that is the union of all MBRs below. Notice, that in contrast to single dimension indexing, keys cannot be sorted as there is no order. Nevertheless, on most datasets the tree will maintain a form that allows the search algorithm to quickly discard irrelevant regions.

Figure 2 shows an example R-Tree designed to store the boundaries of European countries. Figure 2(a) details part of the index structure, and Figure 2(b) graphically depicts the actual boundaries and MBRs that define the R-Tree. Notice that although European countries do not overlap, their MBRs do. The tree has height 3. The root node (Level 3) contains two MBRs, R_1 and R_2, shown as the wider (blue) lines. Notice that there is some overlap, as we cannot find a disjoint balanced union of MBRs that covers the whole of Europe. The overlap is even more evident on Level 2, Also observe that whereas Iceland, Greece and Portugal belong to a single box *for each level*, the central Alps region in Europe is covered by a large number of overlaping MBRs *at all levels*.

The main query we use in this application is the `overlaps` binary constraint, `&&`, also the key operator on the Postgis spatial RDBMS [14]: `A && B` *constraint is satisfied if* `A`*'s bounding box overlaps* `B`*'s bounding box.*

A query using this operator is shown next:

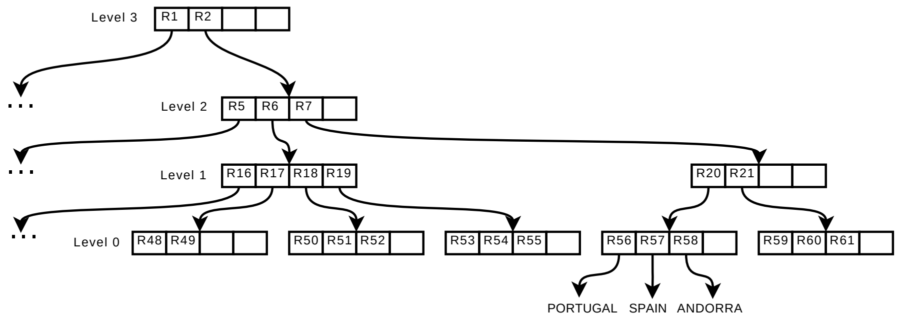
```
?- country(spain,P1), P2 && P1, country(Country,P2).
```

The first sub-goal instantiates `P1` to the polygon for `spain`. Thus, `P2 && P1` are called with `P1` bound and `P2` will be attributed a value by `overlap(_,P1)`. The benefit is that the second call to `country` will only search the database for countries that have overlapping boundaries with *spain*.

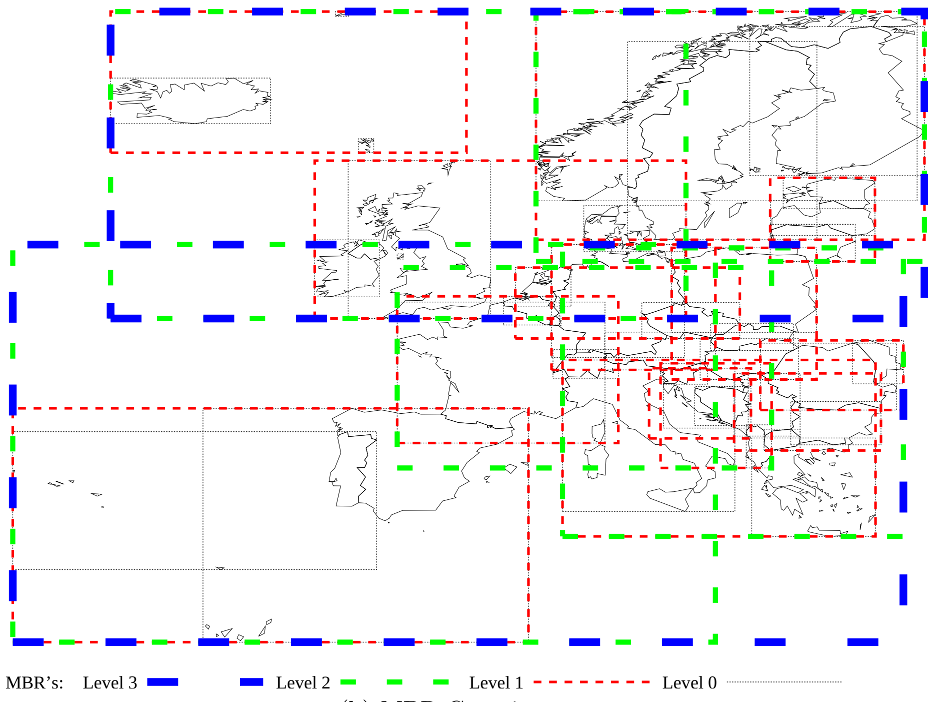
We should remark that `&&` only approximates overlapping, based on MBRs. In our implementation we perform intersection explicitely, although intersection could naturally be performed within the constraint solver. For example:

```
?- country(spain,P1), P2 && P1, country(C,P2),
   intersection(P1,P2,P3).
```

The query searches for countries that intersect with Spain. The overlapping constraint prunes the results to Portugal, France and Andorra, but only the latter will eventually succeed. Notice that the same result would be achieved without the use of UDI, but with a high penalization in time:



(a) R-Tree Structure



(b) MBR Containment

Fig. 2. R-Tree of Europe Countries.

```
?- country(spain,P1), country(C,P2), overlap(P1,P2),
   intersection(P1,P2,P3).
```

Our results show that using this form of indexing is fundamental in operating effectively with spatial data.

3 The Fire DataSet

Portugal being the smallest of the five southern Europe countries, is the most affected by fire in terms of occurrences and relative burnt area. From 1980 to 2004, 30% of the country was burnt (equivalent to 1 fire per 20ha). The closest cases (Italy and Spain) present values of fire occurrence, density and burnt area inferior by 1/3 and 1/5 respectively.

Given the increasing trend of burnt area and with the increasing changes in temperature and precipitation, it is of the outmost importance to narrow the problematic areas in order to effectively promote fire control and landscape management. Recent efforts have provided detailed information of burnt areas between 1990 and 2007.

Initial work on this area has focused on a parish-based approach, where the goal has been to study differences between different regions in the country [15], In related work, Stojanova et al use geographic and weather data to predict forest fires [16]. A number of different classifiers and regression techniques were applied, with best results obtained through bagging decision trees.

The motivation for our work was the need to consider different sources of data, given that we have both parish data and the COS'90 database, a detailed land cover map, produced by the National Center for Geographic Information. The COS'90 database was obtained by visual interpretation of aerial photographs from 1990 followed by polygon vectorization, with 3 digit nomenclature for each polygon. The nomenclature describes the principal and secondary type of use, e.g., `PE2` would express a polygon with a mixed forest based on Pine Tree and Eucalyptus covering up to 75% of the area. The order of the first letters informs that Pine Tree is dominant, the digit informs that we have between 30% to 50% of coverage.

We further remark that polygons vary widely in size. Moreover, polygons do overlap with each other, and we have cases of polygons that are contained in other polygons. We do not exploit such overlaps in this work.

Given the fine grained level of this dataset and the absence of detail in burnt areas polygons, e.g., a given burnt area polygon may represent several fires occurring with a spawn of several months within a year, we have abstracted the burnt area by tagging the COS'90 polygons with a `burnt` label for each year, making this layer our base layer. We have only used burnt information from 1991 to 1999, an acceptable spawn given the base date of COS'90.

Additional data information can be obtained by considering a second layer with socioeconomic information. In Portugal this data is available through statistics taken over parishes.

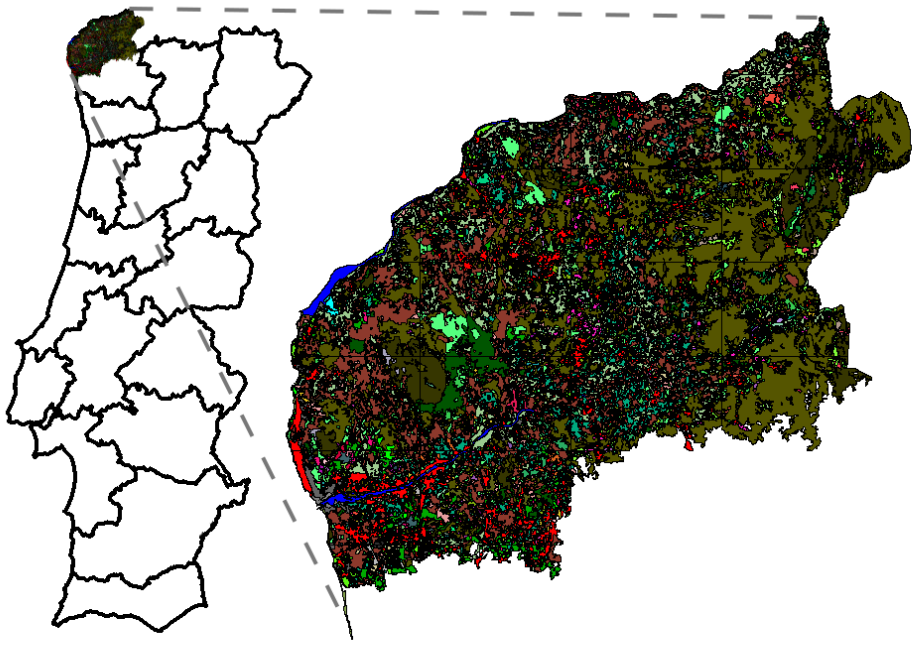


Fig. 3. COS'90 Polygons in Viana do Castelo

We focus on the Viana do Castelo district (county) - see Figure 3. This district is one of the most heavily forested in Portugal, and has suffered from a wide diversity of fires. Previous work indicates that different regions have very different patterns: Viana do Castelo is typical of the North of Portugal and is one of the most affected sub-regions of the country. The district has 290 parishes and 15091 COS'90 polygons.

Notice that the relation between parishes and polygons can be quite complex. Figure 4 shows a situation where a large polygon overlaps a number of parishes. The opposite is also possible, and a small polygon can be easily contained in a parish.

3.1 Methodology

In this work we are interested in exploring spatial predicates on-the-fly during the ILP search process. We thus rely on spatial indexing to obtain more efficient execution of queries involving spatial predicates. Notice that even with spatial indexing, geographical queries are typically very expensive. We further use *tabling* to reduce recomputation to the minimum. As an example of this type of optimization, consider the *neighbor* relation:

```
:- table neighbor/2.
neighbor(ID1,ID2) :-
```

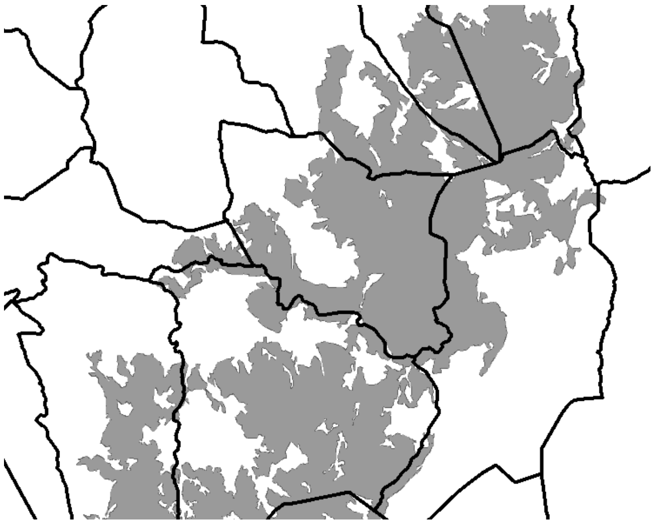


Fig. 4. Polygons and Parishes. Notice the irregular structure of the polygon and how it crosses over a number of parishes.

```
cos90(ID1,R1,P1), R2&&R1, cos90(ID2,R2,P2), ogc_touches(P1,P2).
```

Logically, it would be sufficient to express the `cos90` polygons and then use `ogc_touches`. Assuming that R_1 is bound, the `&&` constraint selects polygons such that R_2 overlaps R_1, R_1 and R_2 are the Minimum Bounding Rectangle (MBR) for the polygons P_1 and P_2 respectively. This is a necessary but not sufficient condition for the actual polygons to touch. The relation `ogc_touches` holds true if and only if the two polygons touch.

Notice that `&&` does not introduce any new logical information. On the other hand, `&&` is implemented very effectively by using UDI with `R`-Trees. In contrast, `ogc_touches` is extremely expensive: it needs to compare two complex polygons. Our approach reduces very significantly the number of calls to `ogc_touches` and makes the whole computation feasible.

However, it is clear that the `neighbor` operation will be used quite often, and may have to be recomputed every time we run a rule. We use tabling to avoid this problem. More precisely, we use tabling with *local* scheduling so that *all* solutions to the query are computed the first time the query is run.

A second interesting problem arises from the need to join the two base layers: how do parishes match COS'90? Both cover the same area, but they have different granularity and they have different information associated with it. They even overlap each other as seen in the example in Figure 4. In general, we do not expect to find an ideal solution to this problem: but in order to use this data we have used a weighted average based on the area of the intersection of both layers.

```
expbox(ID1,Class,ID2,Distance) :-
  expbox(ID1,Class,0,ID2,Distance), !.

expbox(ID1,CL,0,ID2,DISTANCE) :-
  cos90(ID1,R1,P1), R2&&R1,
  findall((ID2,D),
          (cos90(ID2,R2,P2),class(ID2,CL),ogc_distance(P1,P2,D)), L),
  mindistance(L,ID2,DISTANCE).
expbox(ID1,CL,Expand,ID2,Distance) :- Expand > 0,
  cos90(ID1,R1,P1), expand(R1,Expand,R1E), R2&&R1E,
  findall((ID2,D),
          (cos90(ID2,R2,P2),class(ID2,CL),ogc_distance(P1,P2,D)), L),
  mindistance(L,ID2,Distance).
expbox(ID1,CL,Expand_,ID2,Distance) :-
  Expand is Expand_ + 10000, expbox(ID1,CL,Expand,ID2,Distance).
```

Fig. 5. Neighbor Search in the Background Knowledge.

Spatial distance between two spatial objects corresponds to the minimum distance between any two points of each spatial object. Hence to find the minimum distance to a water class polygon, for example, we would need to calculate the distance to each water class polygon. In this case, indexing is not straightforward, but is still worthwhile. The R-Tree indexing structure abstracts spatial objects to its Minimum Bounding Rectangle, and is stored in a form that allows us to discard spatial objects far from the search rectangle. Using the indexing structure we can speedup minimum distance calculations by expanding gradually the search rectangle, starting from the MBR of the base polygon, until a matching polygon is found. A version of the algorithm is presented in Figure 5.

3.2 The Background Knowledge

We can now present the background knowledge used in this experiment. We combine a number of different information sources.

The `class/2` relation identifies all the activities pertaining to the polygon. It is defined in Prolog as:

```
class(ID,C) :-
      pol(ID, CL),
      sub_atom(CL,_,_,_,C), C \= ''.
```

The area relation corresponds to the `ogc_area/2` relation discussed above, and gives the polygon's area. The `neighbor/2` relation corresponds to `ogc_touches` and gives the connection between different polygon.

As an example of temporal correlated information, we also have relations saying whether a polygon was burnt in the previous year or whether it was burnt ever.

Besides `class` and neighbor information, we use information from parishes, currently the amount of cattle on a certain parish. As a polygon may intersect several different parishes, we estimate the cattle in a polygon using a weighted average of cattle based on the intersection area.

Last, we use the `geq/2` and `leq/2` relations to handle numeric data.

4 Results and Discussion

Our task is to predict whether a polygon will catch fire. We recall that forest fires are complex events with a large variety of causes. We would not expect to be able to predict exactly which polygons will take fire. On the other hand, it is worthwhile to find rules that are highly indicative of vulnerability to fire.

We use the ILP system Aleph [17] running under the Prolog system YAP-6 [18] to search for fire risk areas. As discussed above we performed this study on the years from 1991 to 1999. In each year, positive examples (COS'90 polygons with fire occurrence) range from 180 to 1834 occurrences. We used as negative examples the remainder of 15091 polygons in the dataset. The dataset is therefore highly skewed.

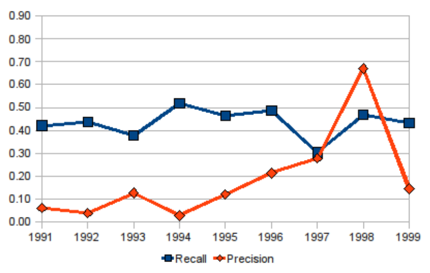
We follow two different types of approaches: first, we use cross-validation over the different years; second, we try to predict the *next* year. In the latter case, we can learn with multiple years: we use up to three consecutive years. To evaluate runs on the same year we used stratified 10-fold cross validation. Results can be seen in Figure 6.

Figure 6(a) shows system performance at every year. Given that the dataset is very skewed, we use precision and recall as a measure of performance. Recall performance on the test set tends to range around 50%, and precision around 10%. We find these values quite acceptable, given the nature of the problem and the skew of the dataset (only about 2% of the examples are positives). Notice that the results vary significantly according to each year. In general, precision tends to be best for the years with most fires. In contrast, recall tends to be worse for these years: this is because we learn less rules in these years. The results for 1998 are quite interesting. This year about 1800 polygons burned (10% of COS'90), and the following single rule is highly predictive:

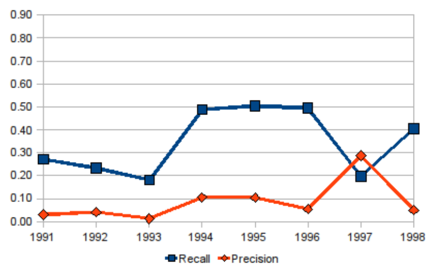
```
burnt(A,E) :-
   burnt_before(A,E).
```

Figure 6(b), 6(c), and 6(d) show next year validation performance with one year, two year, and three year training. Because years are widely different, testing the rules on the next year tends to have poorer performance than using the same year. On the other hand, as we use more years to train the system, recall and precision improve and approach same year training. Moreover, performance becomes more stable and less sensitive to variations in a year (on the other hand, we cannot take advantage of special years such as 1998). In general, with 3 year training we get a recall over 60%, with a precision of about 10%.

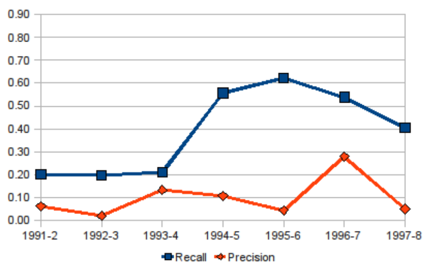
Two examples that give a flavor of the rules learned by our system:



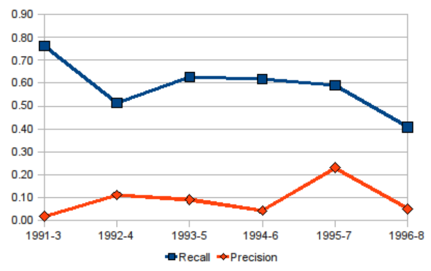
(a) Single Year, cross validation.



(b) Single year, next year validation.



(c) Two years, next year validation.



(d) Three years, next year validation.

Fig. 6. Results

```
burnt(A,_) :-
   class(A,'II'), water_pol(A,_,B), B >= 6736.85994035496.
burnt(A,E) :-
   parish(A,B), sheep(B,_,_,C), C >= 34,
   neighbor(A,D), burnt_last_year(D,E), class(D,'II').
```

The first rule refers to a polygon classified as "improductive" i.e. fallow land. The rule states that such land is quite likely to burn if more than 6Km away from a water source. The second rule applies to a polygon that is in a rural area with a high increase in sheep population, and close to fallow land that often burns. Rules also have a geographic interpretation. Figure 7 graphically shows coverage for the second rule. Notice that most polygons covered by the rule are in the interior, more precisely, on the mountain regions of Viana. Notice also that the rule mostly refers to contiguous regions. In general the found rules most often refer to previous fire activity, to types of vegetative cover such as fallow lands, brush, pine and oak, to herding and to distance to water. Also, a high percentage of rules refer to neighboring polygons.

Exporting the Rules As a way of evaluating the usefulness of our rules, we experimented with applying rules learned in Viana do Castelo on a different district (county). We experimented with Braga district, the southern neighboring district to Viana. Braga shares many of the traits in Viana, but is a larger and

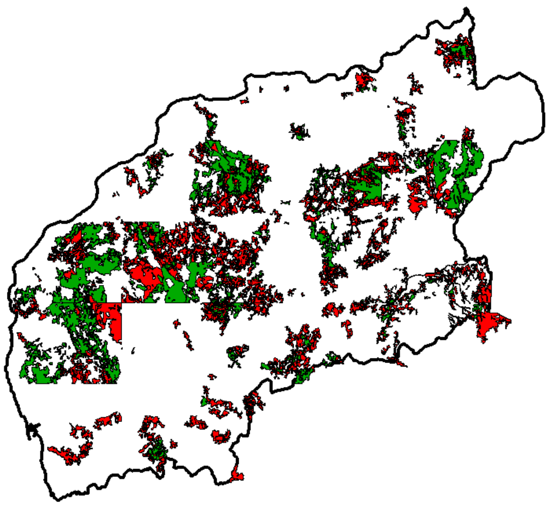
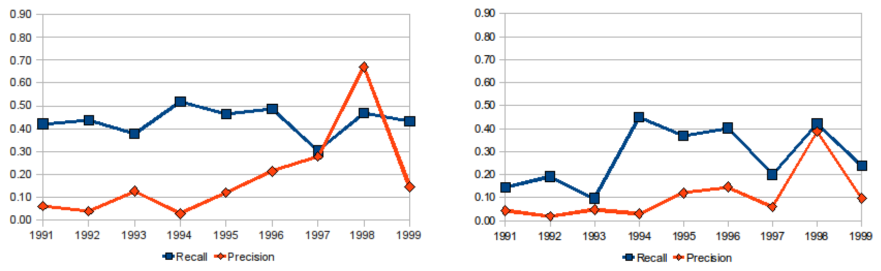


Fig. 7. Positive and Negative Coverage of Rule2 in Viana do Castelo

more complex region. It is also more heavily populated, with a smaller portion of forest area. Otherwise, the types of occupation are similar in both regions.



(a) Single Year, next year validation in Viana do Castelo.

(b) Single Year, next year validation in Braga.

Fig. 8. Comparing Results in Braga and Viana do Castelo

Figure 8 compares results for rules learned in one year and applied to the next year in Viana and Braga. Notice that there is a strong correlation between the two curses. On the other hand, the results are somewhat worst for Braga than for Viana, as expected, but still better than default accuracy.

5 Conclusions

In this paper we have presented an ILP approach to spatial data mining, addressing the pressing problem of wildfire prevention through the understanding of the impact of landscape organization. Our work leverages the machinery we developed in previous research, namely in the construction of an OGC-compliant logic-based geographic information system. A fundamental contribution of this work results from the coupling of an ILP system with a logic-based geographic information system. This coupling avoids the off-line materialization step of spatial features using external geographic information systems, allowing the search process to dynamically explore spatial relationship predicates in the formulation of clauses. The use of multi-dimensional indexing and tabling prove to be also crucial for the computational feasibility of our approach, providing an additional contribution for the use of ILP in the context of spatial data mining with real-world datasets.

Acknowledgments

This work has been partially supported by projects JEDI (PTDC/EIA/66924/2006), STAMPA (PTDC/EIA/67738/2006), and HORUS (PTDC/EIA/100897/2008), and by funds granted to LIACC, CRACS and IT through the Programa de Financiamento Plurianual, Fundação para a Ciência e Tecnologia and Programa POSC. David Vaz is funded by FCT PhD grant SFRH/BD/29648/2006.

We also thank João Honrado and João Torres for providing the fire dataset and for the preliminary discussions in understanding the role of landscape topography in the occurrence of fires.

References

1. Vaz, D., Ferreira, M., Lopes, R.: Spatial-yap: A logic-based geographic information system. In: ICLP '07: Proceedings of the 23rd International Conference on Logic Programming, Berlin, Heidelberg, Springer-Verlag (2007) 195–208
2. Open GIS Consortium, I.: OpenGIS Simple Features Specifications For SQL (1999) Available from `http://www.opengis.org/docs/99-049.pdf`.
3. Ceci, M., Appice, A., Loglisci, C., Caruso, C., Fumarola, F., Malerba, D.: Novelty detection from evolving complex data streams with time windows. In: ISMIS '09: Proceedings of the 18th International Symposium on Foundations of Intelligent Systems, Berlin, Heidelberg, Springer-Verlag (2009) 563–572
4. Ng, R.T., Han, J.: Efficient and effective clustering methods for spatial data mining. In Bocca, J.B., Jarke, M., Zaniolo, C., eds.: VLDB'94, Proceedings of 20th International Conference on Very Large Data Bases, September 12-15, 1994, Santiago de Chile, Chile, Morgan Kaufmann (1994) 144–155
5. Malerba, D., Lanza, A., Appice, A.: 10. In: Geographic Knowledge Discovery and Data Mining. 2nd Edition. CRC Press - Taylor and Francis (2009) 258–291
6. Malerba, D., Esposito, F., Lanza, A., Lisi, F.A., Appice, A.: Empowering a gis with inductive learning capabilities: the case of ingens. Computers, Environment and Urban Systems **27**(3) (2003) 265–281

7. Malerba, D.: Learning recursive theories in the normal ilp setting. Fundam. Inf. **57**(1) (2003) 39–77
8. Lisi, F.A., Malerba, D.: Inducing multi-level association rules from multiple relations. Mach. Learn. **55**(2) (2004) 175–210
9. Soares, T., Ferreira, M., Rocha, R.: The MYDDAS Programmer's Manual. Technical Report DCC-2005-10, Department of Computer Science, University of Porto (2005)
10. Rocha, R., Silva, F., Santos Costa, V.: YapTab: A Tabling Engine Designed to Support Parallelism. In: Conference on Tabulation in Parsing and Deduction. (2000) 77–87
11. The GEOS Development Team: GEOS: Geometry Engine Open Source Available from `http://geos.refractions.net/`.
12. Guttman, A.: R-trees: A dynamic index structure for spatial searching. In Yormark, B., ed.: SIGMOD'84, Proceedings of Annual Meeting, Boston, Massachusetts, June 18-21, 1984, ACM Press (1984) 47–57
13. Vaz, D., Santos Costa, V., Ferreira, M.: User defined indexing. In: ICLP '09: Proceedings of the 25th International Conference on Logic Programming, Berlin, Heidelberg, Springer-Verlag (2009) 372–386
14. The Postgis Development Team: Postgis adds support for geographic objects to the postgresql object-relational database. Available from `http://postgis.refractions.net/`.
15. Torres, J., GonÃ§alves, J., Torgo, L., Honrado, J.: Fire and landscape: A multi-scale assessment of a complex realation. In: Landscape Ecology International Conference. (2010)
16. Stojanova, D., Panov, P., Kobler, A., Džeroski, S., Taškova, K.: Learning to predict forest fires with different data mining techniques. In: Proceedings of the 9th International Multiconference Information Society 2006(IS 2006), Jožef Stefan Institute (2006) 255–258
17. Srinivasan, A.: The Aleph Manual. (2001)
18. Costa, V.S.: The life of a logic programming system. In de la Banda, M.G., Pontelli, E., eds.: ICLP. Volume 5366 of Lecture Notes in Computer Science., Springer (2008) 1–6