A Pictorial Query Language for Querying Geographic Databases using Positional and OLAP Operators

Elaheh Pourabbas[*], Maurizio Rafanelli

Istituto di Analisi dei Sistemi ed Informatica - CNR, Viale Manzoni 30, 00185 Roma, Italy
e-mail: {pourabbas, rafanelli}@iasi.rm.cnr.it

Abstract

The authors propose a declarative Pictorial Query Language (called PQL*) that is able to express queries on an Object-Oriented geographic database drawing the features which form the query. These features refer to the classic ones of a geographic environment (geo-null, geo-points, geo-polyline, and geo-region) and define the alphabet of the above mentioned language. This language, extended with respect to a previous one, considers twelve positional operators and a set of their specifications. Moreover, the possibility to use the mentioned language to query multidimensional databases is discussed. Finally, the characteristic of the mentioned language by a query example is shown.

1 Introduction

Depending on the complexity of the conventional queries to carry out in a GIS and considering their limitations, the authors propose a declarative Pictorial Query Language (called PQL*) that is able to express queries on an Object-Oriented geographic database drawing the features which form the query. The data model of the mentioned geographic database which forms the PQL* underlying data model is described in the next section. The features are interpreted by means of ad hoc procedures. These features refer to the classic ones of a geographic environment (geo-points, geo-polyline, and geo-region) and define the alphabet of the above mentioned language. This language that is extended with respect to a previous one, is characterized by twelve positional operators added to the basic operators (topologic and metric), as well as a set of their specifications. Moreover, the possibility to use PQL* for querying multidimensional databases by introducing specific attributes into the geographic data structure, is discussed. The powerful of the proposed PQL* to answer queries in a very simple and immediate way is illustrated.

2 The Geographic Data Model

The geographical data model consists of a set of abstract data types (ADTs), called *alphabet*, which are: null, point, polyline (oriented or not oriented), region. Generally, all the model which use the object-oriented methodology are based on the existence of the following sets:

- a finite set of atomic domains $D = \{D_1, \dots D_m\}$
- an infinite numerical set $A = \{A_1, A_2, \dots, A_m\}$ of symbols called *instance variables*
- an infinite numerical set *OID* of *identifiers*
- the set C of all the classes defined in the database
- the set O of all the objects defined in the database
- the set M of all the methods defined in the database.

A geographical class is a set of elements called geographical objects $go_1, go_2, \dots, go_n$, which have the same properties $A_1, A_2, \dots, A_m$ defined over the set of not necessarily distinct values $D_1, D_2, \dots, D_m$ and a set of methods which describe the behavior of the objects. For obvious reasons, now we give only the formal definition of a geographical class and of a symbolic feature:

Definition: a *geographic class* (gc) is a quadruple:
<n, sc, P, M> where:
- n is the name of gc;
- sc is the class-parent of gc;
- P is the set of attributes which define the geographic entity properties (enclosing the set of attributes inherited from the relative superclass sc); it consists of:
- a *geometric attribute*, is the type of geometric structure which refers to the geographic class. It can assume only one value chosen in its domain definition: {*null*, *point, polyline, region*};
- a set of *alphanumeric attributes* which defines the non geometric characteristics of the gc instances; each of them consists of a couple (attribute name, data type);
- M is the non empty set of methods such that:
M ={*inherited methods from* sc} ∪ {*methods defined on* gc} ∪ {*overridden methods*}

Definition: A *symbolic feature* (ψ) is defined by a 5-tuple:
$\psi = < id, objclass, objalias, S, L >$ where:
- id is the identifier of the symbolic feature,
- *objclass* is the class set iconized by ψ;
- *objalias* allows to create references to this ψ from inside other ψ;
- S (set of properties) represents the attributes to which user can assign a value range; this assigning allows to carry out a selection among the objects or the classes iconized by ψ. In particular, the geometric attribute is the only

[*] The author is currently visiting researcher at the University of California - Lawrence Berkeley Laboratory. MS: 50B-3238, Berkeley, CA 94720

attribute to which only one value (and not a range) has to be assigned;

- L is the ordered set of couples (h, v) which specifies the object position respect to the top-left point of the query work space. By these ordered sets of couples it is possible to determine the topologic relations among the symbolic features drawn on the work space.

The user formulates his query by a finite set of spatially related symbolic features, assigning to them the wishing meaning. The symbolic features to which it is possible to assign a semantics are: 1) geo-region; 2) geo-polyline; 3) geo-point. These symbolic feature types form, together the element "geo-null", the elements of an alphabet A, that is:

A= {geo-region, geo-polyline, geo-point, geo-null}

The relative proposed operators are:

Op = {Geo-Union, Geo-Difference, Geo-Disjunction, Geo-Touching, Geo-Inclusion, Geo-Crossing, Geo-Pass-Through, Geo-Overlapping, Geo-Equality, Geo-Distance, at_Nord_of, at_East_of, at_South_of, at_West_of, at_Nord-East_of, at_South-East_of, at_South-West_of, at_Nord-West_of, Above, Below, at_Left_of, at_Right_of}

In order to avoid confusion with respect to the set theory operators, from which some differences exist, and because some of these operators are not applicable to any symbolic feature combination because such combinations can have no sense in a generic geographic context, all the above mentioned operators have as prefix *geo*. For example, by a Geo-Union it is not possible to obtain an object formed by a geo-region with attached a geo-polyline as unique element.

Four different sub-sets of operators are defined: geometric, topologic, metric and positional. The first one consists of a significant part of the set theory operators redefined using a geometric point of view (Geo-Union and Geo-Difference), the second one refers to the topologic operators (Geo-Disjunction, Geo-Touching, Geo-Inclusion, Geo-Equality and Geo-Overlapping), the third one consists only of the Geo-Distance metric operator (all these operators are already formally defined in [FMR99]) and the last one refers to the position of one object respect to another object. For each of the positional operators it is often possible to specialize their meaning. These specializations are: *Completely, Only, Partially, Also.*

All the operators considered in the following are dyadic, because the monadic operators (for example, length, area, boundary-box, etc.) are considered as properties of the single classes, which, obviously, depend on the type. It is important to note that the same operator, applied to different topological combination of the same operands, can give different results [CDF94]. The semantics of the geometric operators is restricted with respect to the classic set theory definitions, in order to obtain, as a result, symbolic features belonging to the base alphabet . The intersection is obtained combining among them topologic operators. For example, for the *Geo-Union* operator the above mentioned restricted semantics (respect to the classic "union") consists to allow, as operands, only couples of the same type. In fact, for example, a Geo-Union between a geo-region and a geo-polyline gives, as result, a geometric object which is not an element of A.

3 The Proposed Positional Operators

We distinguish twelve possible positional relationships between two objects: (*at_North_of, at_South_of, at_East_of, at_West_of, at_North-Est_of, at_North-West_of, at_South-East_of, at_South-West_of, Above, Below, at_Left_of* and *at_Right_of*).

For the first eight ones it is possible to specialize their meaning, that is, each of them can have the following prefix: *Completely, Only, Partially, Also*.

In the following, for sake of brevity, we will formally define only the operator At_North_of and its specializations. The definition of the other operators is the same, a part the obvious modifications.

Definition: A symbolic feature ψ_i is *at_North_of* another symbolic feature ψ_j (and denoted by $\psi_i \upuparrows \psi_j$) if it exists at least one point $p_{\psi_i} \in \psi_i$ such that its y-coordinate $p_{\psi_i y}$ is greater than the y-coordinate ($p_{\psi_j y}$) of all the points $p_{\psi_j} \in \psi_j$ The result of the operation is a symbolic feature ψ_h which coincides with the first operand ψ.

Formally speaking we have:

Let $\psi_i, \psi_j \in A$ be two symbolic features of the alphabet A. Let $\psi_i{}^\circ \cap \psi_j{}^\circ = \varnothing$. Let $p_{\psi_i} \in \psi_i$ a generic point of ψ_i and let $p_{\psi_j} \in \psi_j$ be a generic point of ψ_j. Let $p_{\psi_i y}$ and $p_{\psi_j y}$ be respectively the y-coordinate of the generic points p_{ψ_i} and p_{ψ_j}. Then:

$\psi_i \upuparrows \psi_j$ *iff* $\exists\, p_{\psi_i} \in \psi_i | \; p_{\psi_i y} \geq p_{\psi_j y} \; \forall p_{\psi_j} \in \psi_j$ ◊

In the following, we will consider different cases which can and for each one, we examine all the possible feature combinations. For describing these cases, it is necessary to define the possible *specialization* of the first eight of the positional operators. They are mainly based on the well known concepts of bottom point, top-point, left-point and right-point of a symbolic feature (geo-region or geo-polyline), that for sake of brevity we will consider only the definition of the first one.

Definition: A generic point $p_{\psi_i} \in \psi_i$ of a symbolic feature (geo-region or geo-polyline) is a bottom-point of ψ_i (and denoted by $p_{b_{\psi_i}}$) if its y-coordinate $p_{b_{\psi_i y}}$ is less than (or equal to) the y-coordinate $p_{\psi_i y}$ of all the other points p_{ψ_i} of ψ_i. ◊

The above mentioned specializations are the following:

Definition: A symbolic feature ψ_i is *Completely_at_North_of* another symbolic feature ψ_j (and we write $\psi_i \, C \upuparrows \psi_j$) if the y-coordinate of the bottom point of ψ_i is greater than (or equal to) the y-coordinate of the top point of ψ_j (see Figure 1). We formalize this concept.

Let ψ_i and ψ_j be two symbolic features of the alphabet A. Let $p_{\psi_i} \in \psi_i$ be a generic point of ψ_i and let $p_{\psi_j} \in \psi_j$ be a generic point of ψ_j. Let $p_{\psi_i,y}$ and $p_{\psi_j,y}$ be respectively the y-coordinate of the generic points p_{ψ_i} and p_{ψ_j}. Let $p_{b_{\psi_i}}$ be a *bottom-point* of ψ_i and let $p_{t_{\psi_j}}$ be a *top point* of ψ_j. Then:

$\psi_i\, C \Uparrow \psi_j$ *iff* $p_{b_{\psi_i}} \ge p_{t_{\psi_j}}$ ◊

Analogously we can define the other seven positional operators and the relative specializations, with the obvious modifications.

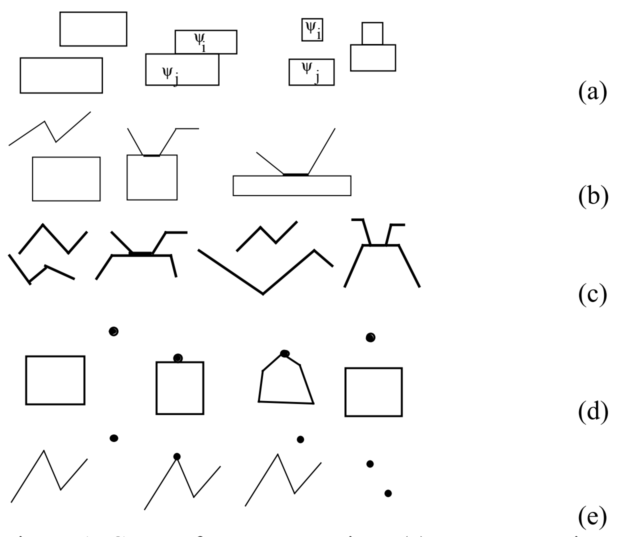


Figure 1. Cases of: two geo-regions (a), one geo-region and one geo-polyline (b), two geo-polylines (c), one geo-region and one geo-point (d), one geo-polyline and one geo-point, and two geo-points (e)

Informally speaking, we have:

- ψ_i is *Only_at_North_of* ψ_j (denoted by $\psi_i\, O \Uparrow \psi_j$) if all the points of ψ_i are included between the Left-point and the Right-point of ψ_j (see Figure 2).

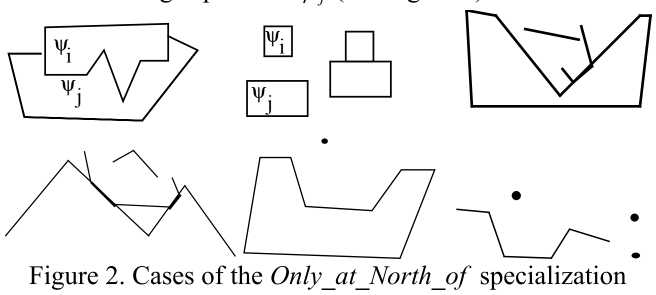


Figure 2. Cases of the *Only_at_North_of* specialization

- ψ_i is *Partially_at_North_of* ψ_j (denoted by $\psi_i\, P \Uparrow \psi_j$) if at least one point of ψ_i is at_North_of all the points of ψ_j and no point of ψ_i is at_South_of all the points of ψ_j (see Figure 3).

- ψ_i is *Also_at_North_of* ψ_j (denoted by $\psi_i\, A \Uparrow \psi_j$ if at least one point of ψ_i is at_North_of all the points of ψ_j, and at least one point of ψ_i is at_South_of all the points of ψ_j (see Figure 4).

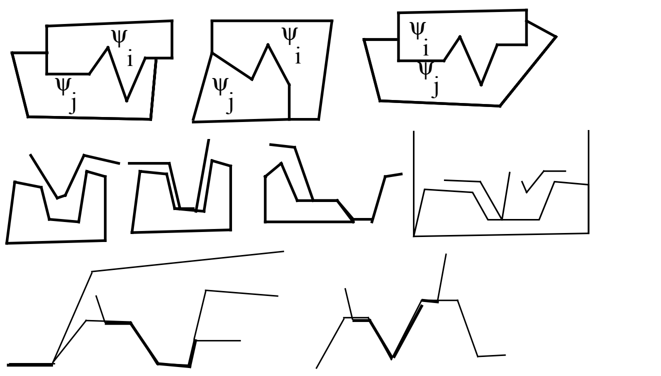


Figure 3. Cases of the *Partially_at_North_of* specialization

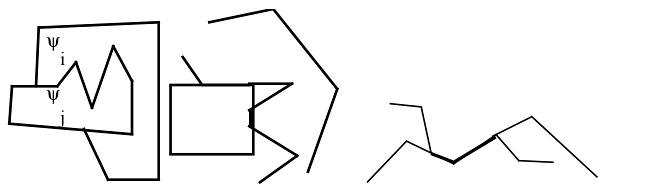


Figure 4. Cases of the *Also_at_North_of* specialization

The other positional operators considered in this paper are: *Above* (∇), *Below* (Δ), *at_Left_of* ($\Leftarrow$), and *at_Right_of* ($\Rightarrow$). For the Above operrator (see Figure 5) we define the following defnition:

Definition: Let ψ_i and ψ_j be two symbolic features of the alphabet A, a feature ψ_i is *Above* another feature ψ_j (denoted by $\psi_i \nabla \psi_j$) if all the points of ψ_i are in the area included between the Left-point $p_{l_{\psi_j}}$ and the Right-point $p_{r_{\psi_j}}$ of ψ_j and if at least one point of ψ_i which has the y-coordinate greater than (or equal to) the y-coordinate of all the points of ψ_i exists.

Formally we have: Let $\psi_i, \psi_j \in$ A be two symbolic features.

$\psi_i \nabla \psi_j$ *iff*

$\forall p_{\psi_i} \in \psi_i |\ p_{l_{\psi_j}x} \le p_{\psi_i,x} \le p_{r_{\psi_j}x}$ *and* $\forall p_{\psi_i} \exists p_{\psi_j} | p_{\psi_j,y} \ge p_{\psi_i,y}$

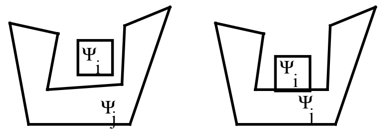


Figure 5. Cases of Above operator

Note that the difference between at_North_of and Above is that in the former it exists at least one point of ψ_i whose y-coordinate is greater than or equal to the y-coordinate of all the points of ψ_j, while in the latter it exists at least one point of ψ_j whose y-coordinate is greater than or equal to the y-coordinate of all the points of ψ_i . Analogously we can define the other specializations, with the obvious changings.

Finally, with regard to the operator *at_Left_of* (and, analogously, to the operator *at_Right_of*), in case in which the reference feature is an *oriented* polyline, it is always transformed in a *double oriented* polyline (suppose to have a river, with the left and right bank, see Figure 6). For example, if a city is only on the left bank of a river, its graphical representation is that of Figure 6 (b). Note that if a river crosses a city, the *alias* modality is used, so that we have the graphical representation of Figure 6 (d).

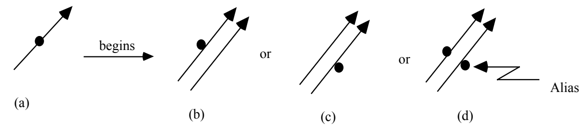


Figure 6. Example of *at_Left_of* and *at_Right_of* operators in case in which the reference feature is an *oriented* polyline

4 Extension of the Model to Support OLAP Operators

Concepts and terminology of geographic data and multidimensional data have been discussed in the literature for a number of years. In both fields, there are several different papers available that deal with conceptual and logical models, operators and relative algebra, query language, etc. While they have become very popular, they lack the ability to capture many important features. In [FPR00] an approach to support an extension of geographic data with multidimensional data in order to query multidimensional databases, is presented. Now we illustrate in which way the correspondence between the same geographic hierarchy present both in Geographic Database (GDB) and in Multidimensional Database (MDDB) can be used to formulate queries to the Geographic Database referring to the typical properties or using the typical operators of multidimensional data.

The concepts of *cube* (e.g., [GBL96], [GL97]), *dimension*, *variable* of a dimension, *measure* [CCS93], *hierarchy* [PR00], as well as the *roll-up*, *slice* and *dice* OLAP operators [Olap97] are widely discussed in literature.

The proposal of using in GDB information which is stored in MDDB depends on the following assumptions.

- If a geographic class name in the GDB corresponds to a level of hierarchy (for simplicity we will call *geographic variable*) with the same name in a MDDB, then to each an instance of this geographic class corresponds only one instance of the geographic variable with the same name in the MDDB; and vice versa.
- Each multidimensional cube defined in the MDDB is simple, that is, each cell of cube contains only one value of measure.
- Each cube's cell contains only a measure with a not null single value, that is, all cubes are complete and without missing or unavailable value.
- Each cube's geographic variable corresponds a geographic class in the GDB.
- All cubes in MDDB are independent and different from each other, that is, there is no cube derived from another one by means of well known OLAP operators.

In a MDDB for using *roll-up* and *drill-down* operators we defined a total and surjective function between each pair of variables (geographic and non geographic). This function provides the full mapping between instances of variables. Then, the *Contains* relationship correspond to a full mapping function, called in this case *Full-Contains* [FPR00].

Suppose a user is working with an object-oriented GDB. If s/he desires to interoperate with MDDB, it is necessary to extend the data structure of GDB, by adding some particular attributes called functional attributes to the already existing ones of a geographic class. Then, with this assumption we classify all attributes that can be included in a geographic class as the following:

- identifier (name), for univocal identification of geographic class.
- geometric, for describing the spatial configuration of the class (geo-region). It is important because the topological, metric, and directional relationships are defined on this attribute.
- measurable, for defining the numeric measure associated to the geographic class, e. g., surface, length, width, depth (profundity), height, etc.
- descriptive, for defining particular information (such as, capital, flag, etc.) associated to the geographic class.
- functional, for describing all the phenomena represented by cubes in the MDDB. Every such attribute consists in a pair <cube name, {cube variable name}>. The former corresponds to the considered phenomenon (e.g., Car_Sales by model, city, month) and represents the cube measures. The later is composed of all variable names except, the geographic one, because it is implicit being the same class name (e.g. CITY). These variables are called *local variables*. Note that every functional attribute represents one and only one cube.

To each functional attribute of a given geographic class (for instance, PROVINCE) corresponds a multidimensional cube of a MDDB in which the geographic variable is the class name (in this case, Province). Obviously, all and only the cubes which have *province* as geographic dimension appear in the functional attributes of the corresponding PROVINCE geographic class. In order to instance each functional attribute of a given geographic object, it is necessary to point out in the geographic variable of the cube identified by the same functional attribute, the instance corresponding to the name of the geographic object. This instance points out a subcube. The instances of the other variable of this subcube and the reactive measures define the instances of the functional attribute of the above geographic object. Each geographic variable of a cube belongs to a level of the geographic hierarchy. Then, along this hierarchy, we can obtain a new cube aggregating all values along the geographic classification hierarchy by roll-up operator. To

this so called aggregate geographic variable corresponds a geographic class with the same name. The functional attribute of this class is defined by the name of such new cube and local variables, naturally making implicit only the geographic variable.
Moreover, in [FPR00] are extensively discussed the cases relative to the presence of two or more different geographic variables in a cube and the relative influence on the organization of the functional attributes of the corresponding geographic classes and the semantic of queries.

5 The Pictorial Query Language: PQL*

By the pictorial query language (PQL*) complex queries can be expressed by simple and intuitive pictorial operations. In fact, the language allows the user to draw all the features involved in the query, to instance each feature giving the selection criteria and, finally, to point the feature which represents the query target. In this way the user expresses in a declarative manner, what he wishes to know. Moreover, the user does not need to learn a query language syntax and semantics to carry out a correct query to the system.

5.1 The queries

Using symbolic features as a representation of geographic classes and objects [DGJ 95], the queries are "composed" putting graphically in evidence the spatial objects (symbolic features) which are involved in it and the topological, positional, etc. relationships which link them. Then, the semantics of the query are expressed drawing geometric figures. Two distinct levels of semantic interpretation exist. The former is the semantics assigned to the algebra operators, the latter is due to the implementation of the class methods which customize the basic behavior of methods instanced to solve the query. The queries which can be formulate on a geographic database express two types of geographical object properties:
- *spatial properties*, as, for example, the positional relationships or the distance; these properties are verified by the geometric attributes of the different objects which form the database.
- *descriptive properties*, as, for example, the population of a given region, or the length of a given river. These properties are verified by the alphanumeric attributes of the above mentioned objects.
Formally, we define a query as a set of seven subsets:

$$Q=\{D, F, H, E, R, S, T\} \quad \text{where:}$$

D is a non-empty set of databases which form the active domain of the current query. All the geographic object instances and classes involved in the query by the same symbolic features are searched in these databases.
F is a non-empty set of symbolic features, created directly by the user, which are involved in the query.
H is a non-empty set of type declaration which links each symbolic feature (enclosed in $S \cup F$) to a class of the class hierarchy rooted to the symbolic feature data type.
E is a set of selection expressions associated to symbolic feature fields which form the selection criterion to choose the object instances which satisfy the query target from the databases which are enclosed in D.
R is a set of relationships (set-type, topological-type, metric or positional-type), eventually empty, among symbolic features belonging to the sets F or S.
S is a set of symbolic features generated as a result of the relationship in R.
T is a non-empty set of symbolic features belonging $S \approx F$ which form the current query target.
A query carries out by the pictorial operations defined in the PQL* (dragging of a symbolic feature, selection of a symbolic feature as the target of the following pictorial operation, grouping and ungrouping of symbolic features, assignment of selection criteria to a symbolic feature, query target definition, etc.) is a combination of a sequence of these pictorial operations. It is generally terminated by the pointing out of the resulting feature, which constitutes the query target.
The creation of a symbolic feature involves not only a new object class, such as instances of the query, but also generates symbolic features to represent relationships (topological, positional, etc.) among the existing objects.
At the end of the pictorial operations the system produces a string of syntactically correct instructions, which corresponds to the graphical query.

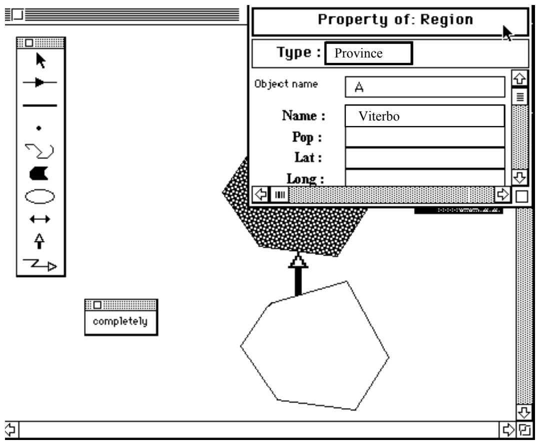
5.2 Query example

In order to describe a query examples, we will give both the pictorial language and a key-words language. This last one is an SQL-like language [Ege92] with the following four clauses:

```
SELECT     Target of query
FROM       Databases
WITH       Properties of features
WHERE      Built-in predicates
           on spatial relationships
```

Let us consider this other following query:
"Find the Italian provinces which are situated Completely_at_North_of Viterbo province"
First of all we generate the symbolic features standing for the objects involved in the query. If we suppose to have a database in which only data regarding Italy are stored, we need two geo-provinces A, and B. The first one, A, represents the Viterbo province. B, instead, represents the generic instance of the *Province* class. We define the object B (the province) as the target of the query. We don't specify any selection criteria regarding the provinces, because we are looking for all the instances which satisfy the query. We mark only the attribute "name of the province" to specify that we need only the province name. Besides, to specify the selection criteria, we need to place properly the symbolic features involved in order to specify the positional relationships existing among these objects. We place A, and B in order to obtain all provinces, represented by B, that are completely at north of Viterbo province (represented by A). This is indicated by an arrow as is shown in Figure 7.
The query expressed in a language SQL-like generated by the system is the following:

```
SELECT      ALL B.name
FROM        database := Italy
WITH        B.type = province
            A.type = province
            A. property.name="Viterbo"
WHERE       B Completely_at_North_of A
```


Property of: Region

Type : Province

Object name: A

Name : Viterbo

Pop :

Lat :

Long :

Figure 7. A pictorial query composition

The result of query is shown in Figure 8.

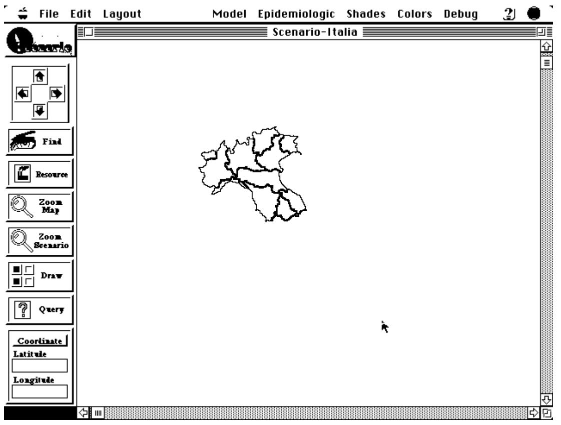


Figure 8. Result of the query

Now, let us consider that the user desires to know cars sold in these provinces and let <Car_Sales, model, month, city> with hierarchy along location dimension) be a cube. We can obtain, as discussed in section 4, the functional attribute <Car_Sales, {model, month}> for each instances of the query shown in Figure 8, by the following formula:

$$slice_{month,model}(roll-up^{sum(Car\_Sales)}_{City \rightarrow Province}(Car\_Sales))$$

6 Conclusions

The proposed PQL* uses strongly the object-oriented paradigm and its peculiar characteristics. This fact allows to the authors to implement a tool characterised by a strong flexibility which helps the user to formulate queries without using a complex syntax and semantics of a procedural query language.

One of the main peculiarities which characterizes PQL* with respect to other proposals (e.g., [LCh95], [AP95], [Ege94], [KwM93], [Mai94], and [Ege97]) is the possibility to perform query at intensional (database schema) and extensional levels. The non procedurality of the query composition, that supports the user attention to what he wants and not how to obtain the result, is another facility of the query language. It allows the user to specify all the properties of and the spatial relationships among the objects through only drawing some symbolic features belonging to a minimal data set in the workplace. The meaning of them are then automatically defined by the system. This type of functionality, that is only adopted in [LCh95], characterizes the high expressive power of the PQL*. Since, the above mentioned language is defined in the context of an object-oriented data model, the exact semantic and calculus method for each operator are established at runtime and they depend on geographic classes on which the relative operator is applied. Finally, it can be interfaced with several commercial OO-DBMS by introducing appropriate algorithms in order to convert it to an SQL-like query language.

References

[AP95] M. A. Aufaure-Portier. A High Level Interface Language for GIS. Journal of Visual Languages and Computing, Vol. 6, 1995, pp. 167-182.

[CCS93] E.F. Codd, S.B. Codd, C.T. Salley, Providing OLAP (on-line analytical processing) to user-analysts: An IT mandate. Technical report, 1993.

[CDF94] E. Clementini, P. Di Felice. Topology in Object-Oriented GIS. Second national congress Sistemi Evoluti per Basi di Dati, Rimini, Italy, 6-8 June 1994.

[DGJ95] S. Dar, H. Gehani, V. Jagadish, J. Srinivasan. Queries in an object-oriented graphical interface, Journal of Visual Languages and Computing, Vol. 6, n°. 1, pp. 27-52, 1995.

[Ege94] M. Egenhofer. Spatial SQL: A Query and Presentation Language, IEEE Transactions on Knowledge and Data Engineering 6 (1): 86-95.

[Ege97] M. J. Egenhofer Query processing in spatial-query-by-sketch. *Journal of Visual Languages and Computing*, Vol.8, pp. 403-424, 1997.

[FMR99] F Ferri., F. Massari, M. Rafanelli, PQL: A Pictorial Query Language for Geographic Features in an Object-Oriented Environment, Journal of Visual Languages and Computing, in press, 1999.

[FPR00] F. Ferri, E. Pourabbas, M. Rafanelli, F.L. Ricci. Extending geographic data bases for a Query Language to Support Queries Involving Statistical Data. In Proceedings of 12th IEEE International Conference on Scientific and Stitistical Database Mangement, SSDBM, July 26-29, Berlin, Germany, pp. 220-230, 2000.

[GBL96] J. Gray, A. Bosworth, A. Layman, H. Pirahesh, Data cube: a relational aggregation operator generalizing group-by, cross-tabs and subtotals. in: 12th IEEE International Conference on Data Engineering, New Orleans, Louisiana, Feb. 26-Mar 1, pp. 152-159, 1996.

[KwM93] J. C. Kwak, S. Moon. Object Query diagram: An extended query graph for object-oriented databases. In. Proc. IEEE Workshop on Visual Languages, pp. 44-48, 1993.

[LCh95] Y. C. Lee, F. L. Chin. An Iconic Query Language for Topological Relationship in GIS. IJGIS vol. 9, n°. 1, pp. 25-46, 1995.

[Mai94] M. Mainguenaud, CIGALES: A Visual Query Language for Geographical Information System: The User Interface, International Journal of Visual Languages and Computing, Acad. Press, 5, 113-126, 1994.

[PR00] E. Pourabbas, M. Rafanelli. Hierarchies and Relative Operators in the OLAP environment. Journal of ACM Sigmod Record, 29(1):32-37, March 2000.

[Olap97] OLAP Council, "The OLAP glossary". http://www.olapcouncil.org. The OLAP Council 1997.