

# Experimenting Advanced Spatio-Temporal Formalisms: an Application to Behavioural Ecology

T. Ceccarelli<sup>1</sup>, D. Centeno<sup>2</sup>, F. Giannotti<sup>3</sup>, A. Massolo<sup>1</sup>, C. Parent<sup>4</sup>, A. Raffaetà<sup>5</sup>, C. Renso<sup>3</sup>, S. Spaccapietra<sup>2</sup> and F. Turini<sup>6</sup>

<sup>1</sup> ISTRICE, Siena

<sup>2</sup> EPFL, Lausanne

<sup>3</sup> ISTI CNR - Pisa

<sup>4</sup> Université de Lausanne

<sup>5</sup> Dipartimento di Informatica - Università Ca' Foscari Venezia

<sup>6</sup> Dipartimento di Informatica - Università di Pisa

## 1 Introduction

This is a story of women, men and porcupines. It deals with the attempt of exploiting advanced database technologies for the representation and the analysis of animal behavior, better of what can be understood from reasoning about the movements of animals. Abstracting away from the specificity of the domain of inquiry, we can say that the paper describes an experiment in utilizing advanced database technology for tackling problems about moving objects in general. Data and, consequently, opportunities for exploiting them for extracting new knowledge and taking informed decisions are increasing day by day. In fact, digital geographic datasets are growing rapidly, due to

- the collection of telecommunication data from mobile phones and other location-aware devices,
- the collection of satellite imagery with higher coverage and resolution,
- the collection of point-of-sale data through transactional databases, recorded into data warehouses.

Such a wealth of data has the particularity of being referenced in space and time and it is expected to enable novel classes of applications of high societal and economic impact, provided that the discovery of consumable, concise, and applicable knowledge out of these raw data is made possible. From this point of view, the ideas and technologies discussed here may have, in our opinion, a stronger impact than the current application in the field of animal behavior analysis.

In this context, formalisms such as MADS[16, 17] and MuTACL P [15] have been proposed as approaches representing and handling spatio-temporal information.

MADS is an object and relationship data model that provides many interesting features such as orthogonality of the structural, spatial, and temporal modeling dimensions, explicit description of spatial and temporal relationships, explicit definition of aggregates, generalization/specialization hierarchies, support of both the discrete and the continuous view in space and time, and generic spatial and temporal abstract data types.

MuTACL P is a formalism, based on Constraint Logic Programming, designed to favor the construction of a software layer, that supplies the user with a declarative spatio-temporal interaction with complex and, almost always, non declarative systems, like heterogeneous databases, the web, and systems managing unstructured or semi-structured data. It allows for the representation and the handling of spatio-temporal information, and, at the same time, knowledge can be separated and combined by means of meta-level composition operations.

These formalisms are experimented here in a case study, provided by biologists, concerning the study of animal behaviour. This field is a typical spatio-temporal reasoning process, since the biologists, starting from a number of spatio-temporal localizations of the animals, called fixes, look for correlations between the time of the fix and the spatial position of the animals. Typical spatio-temporal queries refer to the search of which animals form a couple, or how the home range

of animals changes, or studying possible correlations between meteorological events (e.g., rain, snow, fire) and the movement of animals.

This paper copes with the case study from three different points of view. First of all the conceptual model of the problem is provided, by using MADS. Such a model allows biologists to have a clear structure of the involved entities and it establishes the relationships among such entities. Then, a subset of the problem is represented in MuTACL, with the aim to offer a logical model able to formalise and solve the queries the biologists are interested in. Finally, we consider what the commercial GIS technology provides as instruments to support the analysis of spatio-temporal data by highlighting advantages and lacks of these systems.

Following this order, first Section 2 gives some background on the case study. Section 3 presents the data model MADS and the conceptual model of our application case realized with MADS. Section 4 describes the language MuTACL and the implementation of some spatio-temporal queries asked for by biologists, whereas Section 5 is devoted to the description of a customization of a commercial GIS to provide useful support for our case study. Finally, Section 6 draws some conclusions.

## 2 Hystrix Case Study

Behavioural ecology is the science which studies the animal behaviour in relation to the environment where the animal lives. More specifically our attention is geared towards the study of the behavioural ecology of the crested porcupine (*Hystrix cristata*).

### 2.1 Biological Background

The crested porcupine is a semi-fossorial rodent [22, 7, 13, 24]. The scarce information available on the behaviour of the species belonging to the genus *Hystrix* seems to indicate that the main activity of such animals during the time spent outside the den, is feeding [7, 13, 24]. Clearly, most of the social interactions take place inside the den, which plays a critical role in terms of protection from predators, thermoregulation of the cubs, social behaviour and reproduction. The social unit is represented by extended family groups, composed by 2-4 adults (or sub-adults) as well as the cubs, that use the same dens [22, 8]. The choice of suitable sites for establishing a den, seems to be conditioned by the availability of pre-existing cavities, by the pedological characteristics of the area, as well as by its climatic conditions [18]. Thus, dens may represent an important resource for the species and, as such, may influence the modalities of social distribution and aggregation.

The distribution and the abundance of resources, which are critical for a given species, influence its use of space and the modalities of social and spatial aggregation. The home range of an individual is thus a dynamic expression of the use of space; this may show temporal variations with the age, the reproductive and social state of an animal, as well as with the distribution and abundance of resources. In *Hystrix cristata* the dimensions of the individual, annual home range reported by studies conducted in the inner parts of Tuscany, showed variations ranging from 30 to 255 hectares [13, 24].

The biological research in this context aims at investigating the social organization and the variations in the activities and home ranges depending on the kind of food used. More specifically, the goal is to evaluate the seasonal variations of home ranges in *Hystrix cristata* in a Mediterranean coastal area and to infer those factors which determine the dimensions of the home ranges of this species.

The study has also tried to: investigate the existence of possible forms of spatial segregation or territoriality, induced by differences in patterns of abundance and distribution of resources; identify characteristics of the sites used as dens by the animals; assess distribution and use of what is within the study area; and verify the aggregation of animals in relation to availability of dens and to variations in food resources.

## 2.2 Methods and Techniques

The study site is located in a Mediterranean coastal area, more precisely in the Maremma Regional Park (Tuscany, Italy). Vegetation is characterised by dense scrubwood, pinewood, sparse pastures and cultivated lands (olive, maize, and sunflower). The data were collected between May 1998 and July 1999, as part of a long-term research on the behavioural ecology of crested porcupines. A number of animals were trapped by means of 14 double entrance box traps, positioned along the main trails used by crested porcupines. Traps were activated for at least 7 nights/month and checked at dawn.

In order to remotely localize the animals, a technique named radio-tracking, commonly used in the field of Ethology and Animal Ecology since the seventies, was employed. This technique consists of the application of radio-collars with individual frequencies to the animals and of the subsequent spatio-temporal location of the source of the signal and thus of radio-tagged individuals [12, 26]. As a consequence, adults were equipped with a radio-collar and re-located the first day after capture, in order to detect possible deaths. The location of the animals has been calculated by triangulating the directions of maximum emission of the recorded signals, by means of radio-receivers and antennas, from a number of stations which can range from 2 to many (Figure 1). The directional measurements have been carried out from 78 stations selected within the study

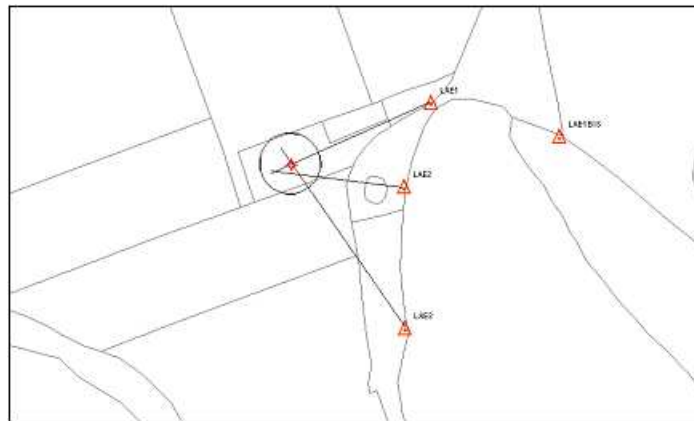


Fig. 1. Bearing

area, which have been subsequently geo-referenced.

The average number of locations for each month and for each individual varied from 20 to 50. In a global time period of two years about 5000 fixes total were collected. The location error was assessed by positioning a number of radio-collars in known positions and by calculating the gap (in meters) between real and estimated position and the difference between estimated and real bearings. The estimations resulted to be  $\pm 5^\circ$  for the bearings, and 62 meters for the position. When estimating the location of radio-tagged animals, telemetry measurements are subject to the influence of many error sources: errors given by the measurement devices, by the operators or errors related to the environmental conditions. Despite the fact that new GPS technologies are nowadays rapidly progressing, telemetry still holds an important role in the Ethology and Animal Ecology. The small dimensions of radio-transmitters, their limited weight and low-cost all make it unlikely that they will be replaced, at least for the next decade. Given the widespread use of this technique in the study of several animal species, the need of obtaining increasingly precise measurements and consistent error estimates has emerged.

For each animal monitored the annual and seasonal home range has been calculated applying the method of Minimum Convex Polygon (MCP)[26]; also the Fixed Kernel method was applied, for the calculation of the usage distribution [27]. In particular, MCP Home Range is one of the

oldest and simplest methods and it determines the minimum convex polygon that includes all the localizations of the animal. The weakest point of this method is the presence of *outliers*, that is localizations that are far from all the others and that influence the extent of the polygon including areas which are never used by the animal. Kernel Home Range is one of the most powerful probabilistic techniques that return the home range area with different probabilities.

Spatial analyses have been performed using ArcView 3.2 by ESRI [11] and a number of extensions (Spatial Analyst, 3D Analyst, Patch Analyst and Movement version 2.0, by USGS Alaska). This choice was made on the basis of several factors, such as in-house availability of the software, a user-friendly interface, the existence of the extension *Movement* which allows home range calculations and other spatial analyses applied in animal ecology.

### 2.3 Spatio-Temporal Questions From The Application Domain

Given the above mentioned general biological goals, we focus here on a number of problems and queries specifically related to spatio-temporal aspects. Each problem is expressed from two points of view: the biological interest in solving that question and the expected behavior of the software tool (GIS and/or DBMS) to solve the query. In details:

#### Problem (1) Den Localization

Den localization is performed by biologists on a bi-monthly basis by a procedure called *homing in*. The biologist follows the radio-signal of individuals during the day to *physically* locate the dens of animals. Each time a den is found, it is recorded in a database.

On the one hand, this method is very expensive and cannot be always applicable. On the other hand, animals usually change their dens, therefore den localization of some animals is not known in periods not covered by homing in. Thus a strong concern consists of finding the den location when no information is available.

**Query.** Given a number of known den positions (collected only on a bi-monthly basis) and the animal fixes, infer the position of dens in periods when no information is available.

#### Functionalities Expected From GIS/DBMS.

The problem of the den localisation requires a system that, starting from the selected time interval of interest for the analysis, is capable of automatically computing locations that are likely a den. The user should have the option of selecting the individual, the period of interest and the probability threshold for a den.

According to the expert knowledge, a crested porcupine stays inside or close to its den between dawn to sunset, hence the function determining a den should select only fixes collected in this specified time interval (i.e., from dawn to sunset) and analyse such data grouping them keeping into account both the spatial and the temporal dimension. As a result, the system returns a set of possible dens, i.e. coordinates that can represent a den with a certain probability.

#### Problem (2) Relations Among Animal Localizations

For understanding the habits and the social behaviour of animals it is extremely relevant to discover the relationships existing among individuals.

To assess the degree of association among individuals, a possibility consists of comparing the overlap of their home ranges, estimated at given time intervals (typically one month or, more generally,  $k$  months). However, such method is quite raw because it can happen that animals stay in common areas but in different periods of time or the overlap might be so large that the simultaneous presence of the animals in the area does not ensure that they are really close to each other.

A more precise procedure requires the calculation of the inter-individual distance between animals localised at the same time. We say that two fixes are *contemporary* if they refer to localisations of animals in the same place and at the same time, i.e., we consider a kind of spatio-temporal closeness among individuals. Since the tracking technique usually presents several sources of error, in the analysis two fixes are assumed to be *contemporary* if they fall within a given time interval and the corresponding positions are within a certain distance. The effective values for the temporal and spatial thresholds are established by biologists.

Analysing this kind of inter-individual distance between animals, it is possible to make hypotheses about which animals can be considered a couple, form a herd, or avoid other individuals.

**Query** Given the animal fixes, determine whether two animals are likely to be a couple by using the above inter-individual distance.

### **Functionalities Expected From GIS/DBMS**

This problem seems to be very “heavy” from the computational point of view, when standardized and automated procedures are not applied. The use of DBMS/GIS should help solving the problem, provided that accurate procedures are developed for identifying contemporary intervals, for estimating the required metrics (distance among points), as well as for calculating outputs for subsequent statistical analysis. The contemporary condition should be definable by the user. Furthermore, the user should have the option of performing calculations on a subset of the original data selection time and/or animal identifier.

### **Problem (3) Time Step For Calculating The Home Range**

A further goal consists of evaluating the seasonal variations of the home ranges in *Hystrix cristata* and also, of inferring the factors determining the dimensions of home ranges for these species.

For this reason home ranges needed to be calculated in different time periods to understand whether home range showed significant variations.

The definition of time intervals to be used in the estimation of the home ranges is often subjective and conditioned by our perception of the temporal units (weeks, months, seasons, years). Very frequently months or seasons (or aggregations of the above) from the Gregorian calendar are used as temporal sub-divisions. Using this criterion for defining the temporal intervals, there is a risk to identify intervals which are shifted with respect to the observed phenomena. In our case this applies to variations in the home range of individuals. Thus, there is a clear need for a technique which provides an estimate of home ranges, regardless of the a-priori definition of time intervals. For this reason (once relevant parameters such as duration and time lag between two consecutive steps, are defined), the estimate of home ranges referring to all possible intervals within the period of study, seems to be very relevant. This is in order to define the most appropriate time step as the basis for the calculation of the home range and to increase the accuracy in defining the moment when the home ranges are varying in a consistent manner.

**Query.** Given the data set described above, the extent of the home range area should be estimated for each individual, starting from a point, say  $t_0$ . The idea is to calculate home range in a “continuous” way within a given time interval (e.g., a season). The “continuity” can be obtained by computing home range on partially overlapping time intervals, each subinterval shifted by an interval ( $\delta t$ ) defined by the user. For example, the query should enable the calculation of the home range for the interval from January 1 to January 31, from January 2 to February 1, from January 3 to February 3, and in this case the shift  $\delta t$  is one day.

**Functionalities Expected From GIS/DBMS.** DBMS and GIS (equipped with dedicated components for home range calculation) are requested to compute home ranges for all possible intervals within the study period. The resulting pattern of home ranges is then analysed by the

user (e.g. by means of time series analysis); duration and time lag between two consecutive steps can be specified by the user him/herself and applied in subsequent computations of home ranges. As for problem 2, automated procedures are needed due to the “heaviness” of the computation. The user should be also given the option of selecting, in addition to the duration variables, among a number of other criteria (i.e. the animal identifier).

#### **Problem (4) Spatio-Temporal Distance Between Localizations And Events**

The spatio-temporal distance between an animal and a particular event, defined in time and space, is one of the problems which are often addressed by animal ecology researchers. Indeed very frequently, there is a need for assessing whether and to what extent an event (e.g. a change in a crop cover, a hunting chase, a meteorological or geo-morphological occurrence, etc.) defined in time and space, has caused variations in the movement of the monitored animals. This calculation has difficulties which are similar to those mentioned for problem 1; an additional issue is that spatial and temporal limitations of the effect of the event on animals, must be defined. For instance, we could exclude all events taking place within a spatial distance from individual greater than a fixed threshold or those events occurring later than the occurrence of the localization itself. Alternatively one could exclude the events occurring before a given period of time. We should expect that if an event concretely influences the animals, it would then “attract” or “reject” them, to an extent which is proportional to their spatio-temporal distance. Moreover this technique should allow us to estimate the time interval between the occurrence of the event and its effects (as well as their end) on the animal.

**Query.** Given a dataset with spatio-temporal localizations of all monitored animals during the study period and a dataset referring to a series of events defined in space and time, the spatio-temporal distance between objects included in the two sets should be calculated.

**Functionalities Expected From GIS/DBMS.** DBMS and GIS are expected to allow the user to discover complex spatial and temporal relationships between the event set and the animal localization set. At first, some parameters have to be selected, such as which animals and event to study. Then, it is particularly relevant to define spatio-temporal constraints on the events with respect to the animal localization set. Indeed, setting the spatial and temporal distances from animals and events allows the user to define the interval of influence of the event on individual. Clearly, depending on the specific case, localizations too distant (in time and/or in space) from the occurring event may not be affected from the event itself and should be automatically excluded from the analysis. To this end a number of parameters are to be set, i.e.:

- the maximum *time* distance between the animal localization and the selected event
- the *spatial* distance threshold between the event and the animal localization.

#### **2.4 Data Organization and Conceptual Modelling**

In wildlife research, a large amount of field data is collected: many environmental variables are monitored in order to infer the influences of their change on behavioural patterns. Furthermore many factors which could influence animal choices may have not been taken fully into consideration from the beginning of the study; as a consequence, a large set of data is usually collected in order to take into account possible variations in the research design protocol.

For this reason the extent and structure of the data collected is very often not clear from the onset. This is even more evident when trying to formalize further the research questions in terms of a conceptual data model. Nevertheless, an attempt was made to develop such data models, even if only as an ex-post exercise (i.e. after most of the data collection had already been undertaken). In this case study we successfully experimented the MADS spatio-temporal data model [17, 16], although limited to only some of the previously mentioned research questions.

### 3 MADS

MADS is an object and relationship data model. Its major features are: orthogonality of the structural, spatial, and temporal modeling dimensions, provision for explicit description of spatial and temporal relationships, explicit definition of aggregates, generalization/specialization hierarchies equipped with appropriate inheritance and late binding mechanisms, support of both the discrete and the continuous view in space and time, and generic spatial and temporal abstract data types. In terms of software tools, MADS is available in a schema editing CASE tool which provides its users with an intuitive visual interface for the definition of a MADS schema. The schema can be easily stored in an existing DBMS or GIS thanks to the automatic translation provided by another complementary tool.

In the following we will present MADS and we will exemplify the introduced concepts by using the crested porcupine case study.

#### 3.1 Structural Dimension

Data structures in MADS basically consist of object types and relationship types, both described by attributes. An object is the database representation of a real world entity. An object type is the descriptive pattern that is defined to characterize a set of real world entities that are classified as being similar. For instance, an object type *Habitat* is defined to represent all kinds of habitats where crested porcupines may live within the region of interest. Each individual habitat will be represented in the database as an object of the type *Habitat*. The definition of the type enumerates the specific attributes and methods (in the object-oriented sense) of interest to the application. Hence all instances of a given type share the same structure and behavior. Each instance of the type holds a (possibly empty) value for each of the defined attributes. For instance, in Figure 9 the *Habitat* object type is defined as bearing five attributes: *habitat-id*, *tree*, *shrub*, *gramineous* and *other-herb*. The *habitat-id* attribute allows identifying each individual habitat. The other attributes store information on its vegetation, in terms of which trees, shrubs, gramineous herbs and other herbs are covering the habitat. For trees and shrubs, the stored information consists of the tree species, height, and quantity (expressed as a percentage), plus a remark attribute where any comment may be stored. The same (except for species information) holds for the herbs and other herbs attributes. The [0:N] and [0:1] specifications attached to the vegetation attributes express that a given habitat may include any number (possibly none) of trees and shrubs species, while only 0 or 1 type of herbs and other herbs may characterize the habitat.

As objects do not live in isolation, but interact with other objects in many possible ways, the MADS data model also provides relationships as a first class concept. A relationship in a database is the representation of a link of interest between real world entities. Hence a relationship is a link among objects, where each object plays a given role in the relationship. As for object types, a relationship type is the descriptive pattern that is defined to characterize a set of similar relationships, i.e. relationships linking objects of the same types with the same roles, and described by the same properties. For instance, the binary relationship type *Frequent* in Figure 9 links the object types *Individual* (representing the crested porcupines) and *Habitat*. Its instances link a given habitat to a given individual to convey that this porcupine frequents this habitat. *Frequent* has one attribute, *error*, to hold an estimation of the error probability in asserting that the individual frequents the habitat. MADS allows n-ary relationship types to be defined (i.e., any number of roles may be attached to a relationship type), and allows two or more roles to link the same object type in a given relationship type (in this case the relationship type is said to be cyclic). As for attributes, numbers shown on the roles define the cardinality of the role. They come in pairs representing the minimum and maximum number of relationships that may link an object of the corresponding type. For instance, the pair 0:N on the role between *Frequent* and *Habitat* tells that the database may store habitats for which no link to an individual exists (minimum: 0), and it may store any number of links to individuals for a given habitat (maximum: N, where N stands for unlimited). Cardinality 1:N on the role between *Individual* and *Frequent* states that every individual stored in the database has to have at least one link of type *Frequent* to a habitat.

Simply stated, this means that every porcupine in the database has at least one known habitat, while the database also contains habitats for which no porcupine is known to frequent it.

Storing data in a database calls for classification of objects and relationships into classes described by a type. Because the same real world information serves many purposes, multiple coexisting classifications are sometimes needed. For instance, the application on hand includes some processes to manage pairs of porcupines and some other, more generic processes, that relate to groups of porcupines. Clearly, the latter also apply to pairs, hence there is a need to keep the information that a pair is a specific kind of group. In advanced data models, such as MADS, this information is kept as a binary link, named an *is-a link* (an arrowhead  $\uparrow$  in the diagrams, see Figure 7, that connects the more specific type (the subtype) to the more generic type (the supertype). By definition, the subtype has all the properties defined for the supertype.

A group of crested porcupines is composed of several individuals. To link a component object to the object it is part of, MADS uses a so-called *aggregation link*, represented in the diagrams by a diamond-shaped icon associated with the relationship type. For example, in Figures 7 and 8 the *Belongs* relationship type between the *Individual* and *Group* object types is an example of this link.

All types are described by properties, which are either attributes or methods. An attribute conveys an information that is necessary about a real world object or relationship. Attributes are either simple (they bear an atomic value), or complex (they are composed of other simple or complex attributes). They are said *monovalued* if they bear a single value, or *multivalued* if they bear a set of values. They may be mandatory (at least one value), or optional. Attributes may be derived, i.e. their value is computed from values of other attributes. Methods are operations on object instances that may be applied to any instance of a type.

## 3.2 Spatial Dimension

Intelligent management of spatial and temporal data calls for the definition of new value domains that support the specific semantics of these kinds of data. Hence MADS includes predefined spatial and temporal abstract data types (SADTs and TADTs), that can be associated with both object types and attributes, thus defining spatial and temporal object types and attributes.

Spatial object types are object types that have a special attribute, called geometry, that defines the spatial extent of the object. Each geometry attribute holds values from one of the known SADTs. Figure 2 lists the SADTs supported in MADS, showing the associated icon (pictogram), their dimensionality in space and their definition. SADTs are organized as an is-a hierarchy, shown in Figure 3, which allows each subtype to inherit properties from its supertypes. The use of generic SADTs, e.g. geo, allows different objects of the same type (or different values of the same attribute) to acquire spatial values that belong to different but compatible types. For instance, in a database that has an object type *WaterExtent*, with subtypes *Lake* (with *SimpleArea* geometry) and *River* (with *Line* geometry) the definition of a geometry for *WaterExtent* would call for using the *SimpleGeo* SADT (both lakes and rivers may be instances of *WaterExtent* and they have different geometry types). SADTs directly support a discrete perception of space, i.e. they allow to associate a geometry with objects, relationships and attributes known in the database. When storing geographical data, it is also sometimes needed to store information that is attached to a geographical zone as a whole, rather than to an identifiable object. For instance, applications may need to store altitude or temperature in a given region. Altitude and temperature are examples of information that pertains to every point in space and, to support this feature, known by GIS users as the continuous view of space, MADS offers the concept of space-varying attribute. It is an attribute whose value is defined by a function whose source domain is some part of the space (either the whole space covered by data in the database, known as *DBSPACE*, or any known extent associated with an object, relationship or attribute) and whose target domain is a given value domain (for instance, integers for altitude). Space-varying attributes are denoted in the diagram using the icon  $f(\text{🌐})$ .

For richer representations of spatial phenomena, MADS supports a predefined set of spatial topological relationships (see Figure 4). They can be used within the definition of a schema to



attach a topological semantics to a relationship type. For instance the topological relationship *Is-Found* in Figure 7 that links the object types *Territory* and *Burrow* enforces the rule that a burrow of a crested porcupine may be linked to a territory only if the geometry of the burrow (a point) is inside the geometry of the territory (an area).

| SADT               | Pictogram | Dimension | Definition  |
|--------------------|-----------|-----------|---|
| Geo                | ⊕         | 0 - 2     | Generic spatial type<br>(can contain any spatial value)   |
| SimpleGeo          | ○         | 0 - 2     | Simple spatial type<br>(can contain a Point, a Line, an<br>OrientedLine, or a SimpleArea)               |
| Point              | •         | 0         | Single point  |
| Line               | ~         | 1         | Set of connected points:<br>a line or a polyline, straight or<br>curve, open or closed, oriented or not |
| OrientedLine       | ↖         | 1         | Line<br>with a start and an end extremities   |
| SimpleArea         | ◀         | 2         | Connected area<br>(with or without holes)   |
| ComplexGeo         | ⊕         | 0 - 2     | Set of Points, Lines, OrientedLines,<br>and/or SimpleAreas  |
| MultiPoint         | ⋈         | 0         | Set of Points   |
| MultiLine          | ⋈         | 1         | Set of Lines  |
| MultiOriented-Line | ⋈         | 1         | Set of OrientedLines  |
| ComplexArea        | ⊕         | 2         | Set of SimpleAreas  |

Fig. 2. MADS Spatial ADTs

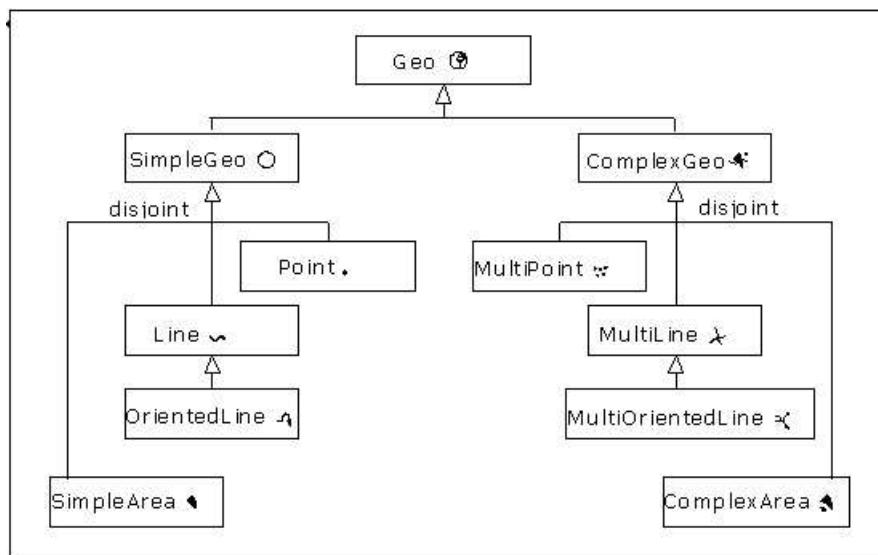


Fig. 3. MADS spatial ADTs hierarchy







| Spatial Type | Pictogram   | Definition   |
|--------------|---|--|
| Disjunction  |  | Objects have no shared geometries  |
| Adjacency    |  | Shared geometries but no common interior   |
| Crossing     |  | Share some part of the interiors such that the dimension of the shared part is less than the greater dimension of the two linked objects |
| Overlapping  |  | Share some part of the interiors such that the dimension of the shared part is equal to the dimension of the two linked objects          |
| Inclusion    |  | The whole interior of one object is some part of the interior of the other   |
| Equality     |  | The two objects share their whole interiors and boundaries   |

Fig. 4. Topological relationships

### 3.3 Temporal Dimension

Temporal abstract data types (TADTs) and temporal relationships are provided in MADS to support description of real world evolution. MADS TADTs are listed in Figure 5 and, as well as SADTs, they are organized in an is-a hierarchy shown in Figure 6. Temporal objects and relationships record their lifecycle (i.e., when the instance is created, suspended, reactivated or deleted). For instance, defining *Individual* in Figure 7 as a temporal object type allows keeping the lifecycle of each crested porcupine: when it was first identified by the application, when it became of no interest anymore for the application. If a porcupine is for some reason captured and removed from the monitored zone covered by the application, its instance in the database may be suspended and reactivated later when the crested porcupine is put back into the study area.






| TADT        | Pictogram   | Dimension | Definition   |
|-------------|---|-----------|--|
| Temporal    |  | 0 – 1     | Generic temporal type (can contain any temporal value) |
| Instant     |  | 0         | Single point in time                                   |
| Interval    |  | 1         | Set of connected points in time                        |
| InstantSet  |  | 0         | Set of disjoint Instants                               |
| IntervalSet |  | 1         | Set of disjoint Intervals                              |

Fig. 5. MADS Temporal ADTs

Evolution in time of the value of an attribute can be recorded by defining the attribute as being time-varying, described in schema diagrams by the icon  $f(\odot)$ . A time-varying attribute is an attribute whose value is defined by a function such that its source domain is some time extent and its target domain is a given value domain. For instance, the *tree* attribute of *Habitat* may be defined as time-varying if the application is interested in monitoring the evolution of the vegetation while keeping past states of the habitats.

In terms of time-related relationships, MADS supports synchronization relationships (e.g., after, before, meets). They are the equivalent in time of topological relationships in space, and

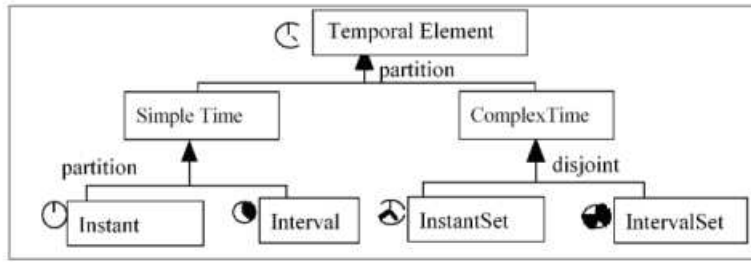


Fig. 6. MADS Temporal ADTs hierarchy

define constraints on the lifecycles of the linked objects. Moreover, MADS introduces some new relationship types with a predefined semantics.

Many applications, and in particular those that are interested in keeping an accurate record of object evolutions, may greatly benefit from this kind of relationships. For instance, MADS defines a *transition* relationship as a relationship linking two instances, from two object types, representing the same real world entity (hence, having the same object identifier within the database) to express that the entity in the real world has acquired a new role and consequently acquires a new membership in a new class. The source object may be preserved or not. *Generation* (with a generation icon) relationships may be used to model real world situations where objects are created from other objects, e.g. in a land reallocation process. A generation relationship type relates one or more source object types to one or more target object types. Related objects have different object identifiers. Some or all of the source objects may be preserved.

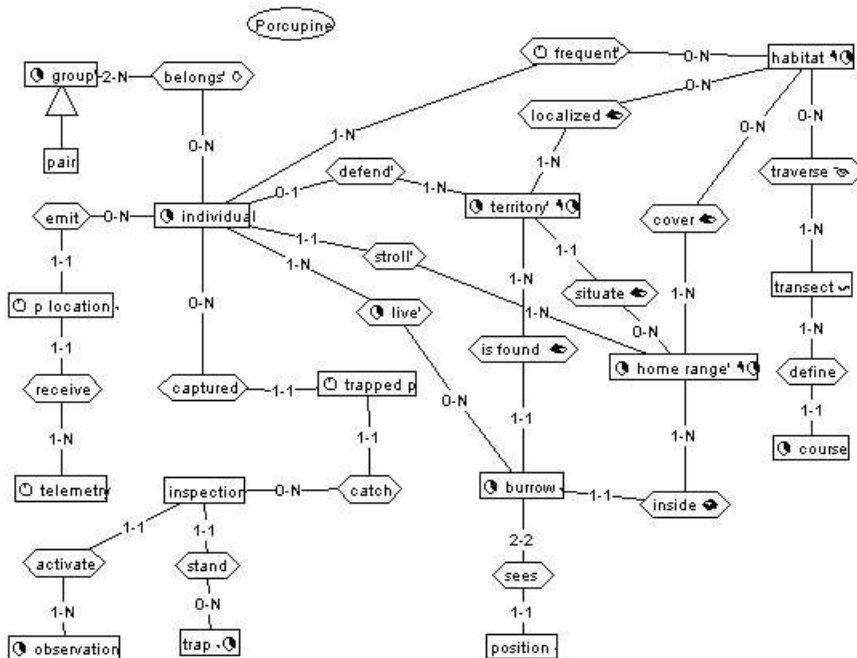


Fig. 7. Simplified diagram of the MADS porcupine database

### 3.4 A Schema for the Porcupine Database

Modeling the crested porcupine monitoring database with MADS has resulted in the schema illustrated in Figure 7 (for readability, the figure only shows types, with no details of attributes and methods). For instance, the possibility to model complex objects, with multiple levels of decomposition in the attribute structures, has allowed to represent trapped porcupines as a single object type (instead of the many tables that would have been needed to design the database using the relational data model). Temporal and spatial concepts have been used to model the lifecycle of several objects (e.g., a crested porcupine is modeled as a temporal object type *Individual*), and the location of many others (e.g., the observation point for burrows). Combined space and time has been used to model time-varying geometry for objects such as territory and home range. Topological relationships model spatial constraints. A synchronization relationship, the *Live* relationship type, is used to allow linking crested porcupines (the *Individual* object type) to burrows (*Burrow* object type) only if at least at some time both the crested porcupine and the burrow were active.

The crested porcupine database schema is quite complex, as is the real world it represents. It is possible to perceive within it three domains of interest: what relates to the social activities of the animals, what relates to their home ranges, and measurements.

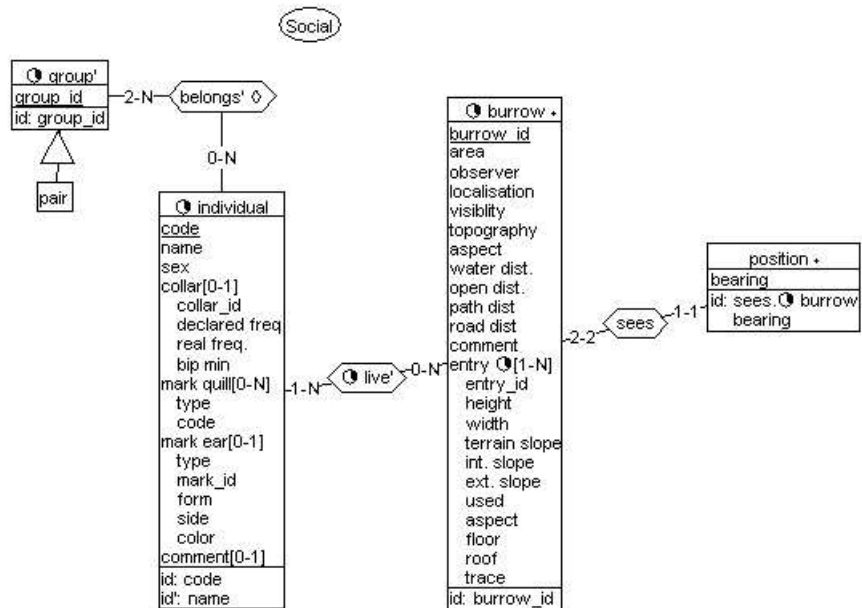


Fig. 8. The social sub-schema

Social behavior is described using the following object and relationship types (see Figure 8):

- *Individual*, a temporal object type that describes porcupines. In addition to the listed attributes, it bears a lifecycle to record the porcupine life length. It is worth noting that the life length is not always precisely known. An extension of the data model to support imprecise time specification would be useful here.
- *Group*, a temporal object type that models groups of individuals living in the same burrow. Its lifecycle defines the set of intervals for which a recording exists of porcupines living together in the same burrow. A group is a derived object: whenever a set of porcupines is observed living in the same burrow, the set is assumed to form a group.
- *Pair* is a subtype of the *Group* object type. A pair is composed by only two adult individuals of opposite sex and eventually cubs. *Pair* inherits all *Group* properties.

- *Burrow* defines the porcupine den. It is a spatial object type with a geometry of type *Point*. Its lifecycle represents the set of intervals during which the burrow is inhabited.
- *Position* models points from where the burrow is seen. It is a spatial object type with a geometry of type *Point*.
- *Sees* is a relationship linking a burrow and a position. Two observation points are necessary to determine the position of a burrow.
- *Live* defines which individual lives in a burrow. Its lifecycle is a set of intervals during which the individual lives in this burrow.
- *Belongs* is an aggregation describing that a group is composed of individuals.

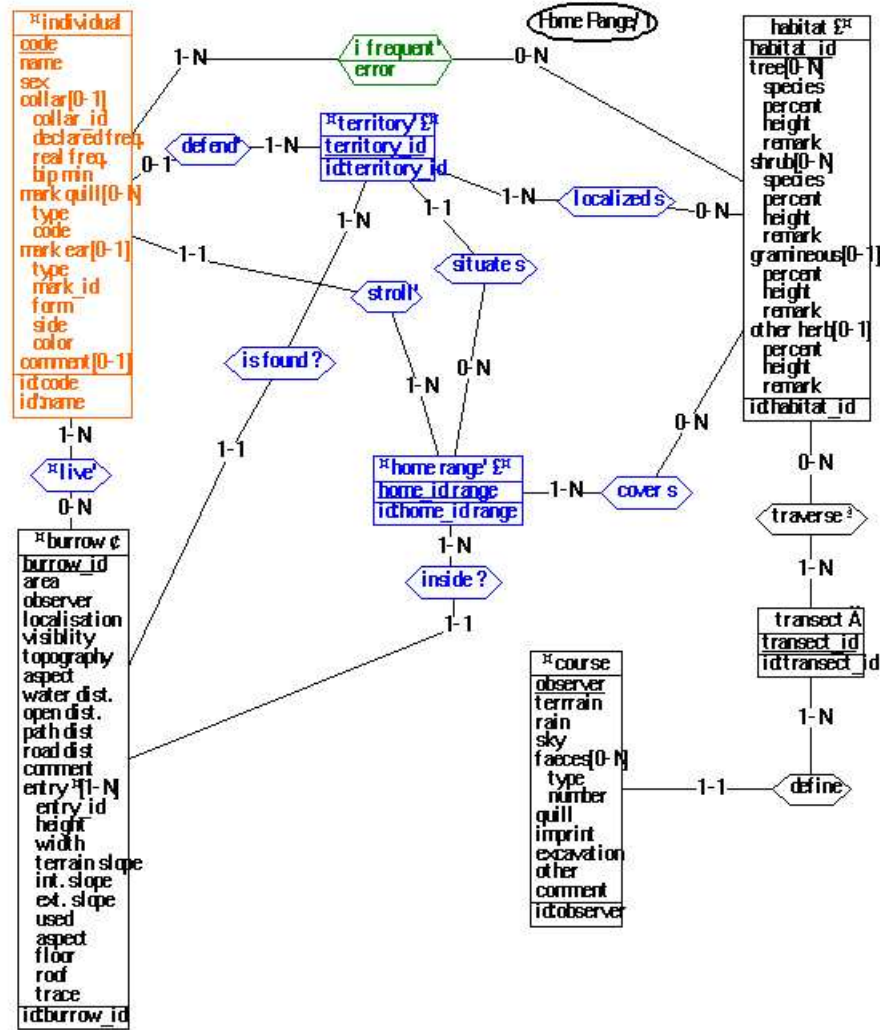


Fig. 9. The home range sub-schema

## 4 MuTACLP: a logical language to support reasoning on animal behaviour

MADS succeeds in providing a conceptual model for the crested porcupine case study. Such a model allows one to represent the relevant entities with their properties and understand the relationships among them. In this section we introduce the language MuTACLP [15, 2] which is able to offer a reasoning capability by which we can answer to some of the queries indicated by the biologists.

MuTACLP has been proposed as a language for representing and handling spatio-temporal information in a framework where pieces of knowledge can be encoded in different programs and possibly combined together by means of composition operators.

Such a language is designed to favor the construction of a software layer, that supplies the user with a declarative spatio-temporal interaction with complex and, almost always, non declarative systems, like heterogeneous databases, the web, and systems managing unstructured or semi-structured data. In [14, 20, 21] we have shown, by means of several examples, how MuTACLP can be used to improve spatio-temporal analysis of geographical data. On one hand, both temporal and spatial information can be represented in MuTACLP [14]. On the other hand, as illustrated in [20, 21], MuTACLP can also be used to express (temporal) knowledge on spatial data stored in a commercial GIS. In this case, the language allows the user to access directly the specific functionalities provided by the underlying system which are thus “combined” with the high-level reasoning capabilities of MuTACLP.

In Section 4.1 we describe the language MuTACLP and in Section 4.3 we demonstrate how MuTACLP can be useful to support spatio-temporal analysis for the crested porcupine case study. As pointed out in Section 2.3, many questions concerning the mating system and the social structure of this animal are still open. Such questions often involve both temporal and spatial aspects, which are usually strictly correlated. For this reason they are difficult to express by using current GIS technology, that does not provide high-level operations to support spatio-temporal queries. On the contrary, our language provides encouraging results, suggesting its adequacy both for the formulation of some of the queries of interest and for the reasoning activity needed to derive lacking information.

### 4.1 Multi-theory Temporal Annotated Constraint Logic Programming

In this section we present the language MuTACLP (Multi-theory Temporal Annotated Constraint Logic Programming), introduced in [15]. Furthermore, in MuTACLP spatial data can be represented by using constraints in the style of the constraint databases approaches [3, 6, 10].

**Time Domain and Annotated Atoms.** Let us start by describing the temporal domain underlying MuTACLP. Time can be discrete or dense. Time points are totally ordered by the relation  $\leq$ . The set of time points, denoted by  $\mathcal{T}$ , is equipped with the usual operations (such as  $+$ ,  $-$ ). We assume that the time-line is left-bounded by 0 and open to the future, with the symbol  $\infty$  used to denote a time point that is later than any other. A *time period* is an interval  $[r, s]$ , with  $0 \leq r \leq s \leq \infty$ ,  $r \in \mathcal{T}$ ,  $s \in \mathcal{T}$ , that represents the convex, non-empty set of time points  $\{t \in \mathcal{T} \mid r \leq t \leq s\}$ <sup>1</sup>. Thus the interval  $[0, \infty]$  denotes the whole time line.

*Annotated formulae*, the basic constituents of MuTACLP programs, are of the form  $A \alpha$  where  $A$  is an atomic formula and  $\alpha$  is an annotation. We consider three kinds of annotations based on time points and time periods. Let  $t$  be a time point and let  $J = [r, s]$  be a time period. Then

- (**at**) The annotated formula  $A \text{ at } t$  means that  $A$  holds at time point  $t$ .
- (**th**) The annotated formula  $A \text{ th } J$  means that  $A$  holds *throughout*, i.e., at *every* time point in the time period  $J$ . The definition of a **th**-annotated formula in terms of **at** is:

$$A \text{ th } J \Leftrightarrow \forall t (t \in J \rightarrow A \text{ at } t).$$

---

<sup>1</sup> The results we present naturally extend to time lines that are bounded or unbounded in other ways and to time periods that are open on one or both sides.

- (in) The annotated formula  $A \text{ in } J$  means that  $A$  holds at *some* time point(s) - but we may not know exactly which - in the time period  $J$ . The definition of an **in**-annotated formula in terms of **at** is:

$$A \text{ in } J \Leftrightarrow \exists t (t \in J \wedge A \text{ at } t).$$

The **in** temporal annotation accounts for indefinite temporal information.

**Partial Order and Constraint Theory.** The set of annotations is endowed with a partial order relation  $\sqsubseteq$ . Given two annotations  $\alpha$  and  $\beta$ , the intuition is that  $\alpha \sqsubseteq \beta$  if  $\alpha$  is “less informative” than  $\beta$  in the sense that for each formula  $A$ ,  $A\beta \Rightarrow A\alpha$ . More precisely, in addition to Modus Ponens, we consider two further inference rules, i.e., the rule ( $\sqsubseteq$ ) and the rule ( $\sqcup$ ) below.

$$\frac{A\alpha \quad \gamma \sqsubseteq \alpha}{A\gamma} \quad \text{rule } (\sqsubseteq) \qquad \frac{A\alpha \quad A\beta \quad \gamma = \alpha \sqcup \beta}{A\gamma} \quad \text{rule } (\sqcup)$$

The rule ( $\sqsubseteq$ ) states that if a formula holds with some annotation, then it also holds with all smaller annotations. The rule ( $\sqcup$ ) says that if a formula holds with two annotations  $\alpha$  and  $\beta$ , then it holds with the least upper bound  $\alpha \sqcup \beta$  of such annotations.

Now, we define the *constraint theory for temporal annotations*. Recall that a constraint theory is a non-empty, consistent, first order theory that axiomatizes the meaning of the constraints. First of all, our constraint theory includes an axiomatization of the total order relation  $\leq$  on time points  $\mathcal{T}$ . Then it contains the following axioms defining the partial order on temporal annotations.

$$\begin{array}{ll} (\text{at th}) & \text{at } t = \text{th}[t, t] \\ (\text{at in}) & \text{at } t = \text{in}[t, t] \\ (\text{th } \sqsubseteq) & \text{th}[s_1, s_2] \sqsubseteq \text{th}[r_1, r_2] \Leftrightarrow r_1 \leq s_1, s_1 \leq s_2, s_2 \leq r_2 \\ (\text{in } \sqsubseteq) & \text{in}[r_1, r_2] \sqsubseteq \text{in}[s_1, s_2] \Leftrightarrow r_1 \leq s_1, s_1 \leq s_2, s_2 \leq r_2 \end{array}$$

The first two axioms state that **th** $I$  and **in** $I$  are equivalent to **at**  $t$  when the time period  $I$  consists of a single time point  $t$ .<sup>2</sup> Next, if a formula holds at every point of a time period, then it holds at every point in all sub-periods of that period ((**th**  $\sqsubseteq$ ) axiom). On the other hand, if a formula holds at some points of a time period then it holds at some points in all periods that include this period ((**in**  $\sqsubseteq$ ) axiom). A consequence of the above axioms is

$$(\text{in th } \sqsubseteq) \quad \text{in}[s_1, s_2] \sqsubseteq \text{th}[r_1, r_2] \Leftrightarrow s_1 \leq r_2, r_1 \leq s_2, s_1 \leq s_2, r_1 \leq r_2$$

i.e., an atom annotated by **in** holds in any time period that overlaps with a time period where the atom holds throughout.

Now we axiomatize the least upper bound  $\sqcup$  of temporal annotations over time points and time periods. For technical reasons related to the properties of **th** and **in** annotations (see [2] for details) we restrict ourselves to compute the least upper bound between **th** annotations with overlapping time periods that do not include one another:

$$(\text{th } \sqcup) \quad \text{th}[s_1, s_2] \sqcup \text{th}[r_1, r_2] = \text{th}[s_1, r_2] \Leftrightarrow s_1 < r_1, r_1 \leq s_2, s_2 < r_2$$

Finally, in order to be able to compose programs we add to the constraint theory defined above the axiomatization of the greatest lower bound  $\sqcap$  of two annotations.

$$(\text{th } \sqcap) \quad \text{th}[s_1, s_2] \sqcap \text{th}[r_1, r_2] = \text{th}[t_1, t_2] \Leftrightarrow s_1 \leq s_2, r_1 \leq r_2, t_1 = \max\{s_1, r_1\}, \\ t_2 = \min\{s_2, r_2\}, t_1 \leq t_2$$

$$(\text{th } \sqcap') \quad \text{th}[s_1, s_2] \sqcap \text{th}[r_1, r_2] = \text{in}[t_2, t_1] \Leftrightarrow s_1 \leq s_2, r_1 \leq r_2, t_1 = \max\{s_1, r_1\}, \\ t_2 = \min\{s_2, r_2\}, t_2 < t_1$$

$$(\text{th in } \sqcap) \quad \text{th}[s_1, s_2] \sqcap \text{in}[r_1, r_2] = \text{in}[r_1, r_2] \Leftrightarrow s_1 \leq r_2, r_1 \leq s_2, s_1 \leq s_2, r_1 \leq r_2$$

$$(\text{th in } \sqcap') \quad \text{th}[s_1, s_2] \sqcap \text{in}[r_1, r_2] = \text{in}[s_2, r_2] \Leftrightarrow s_1 \leq s_2, s_2 < r_1, r_1 \leq r_2$$

$$(\text{th in } \sqcap'') \quad \text{th}[s_1, s_2] \sqcap \text{in}[r_1, r_2] = \text{in}[r_1, s_1] \Leftrightarrow r_1 \leq r_2, r_2 < s_1, s_1 \leq s_2$$

$$(\text{in } \sqcap) \quad \text{in}[s_1, s_2] \sqcap \text{in}[r_1, r_2] = \text{in}[t_1, t_2] \Leftrightarrow s_1 \leq s_2, r_1 \leq r_2, t_1 = \min\{s_1, r_1\}, t_2 = \max\{s_2, r_2\}$$

<sup>2</sup> Especially in dense time, one may disallow singleton periods and drop the two axioms. This restriction has no effects on the results we are presenting.

**MuTACLP Programs and Composition Operators.** A MuTACLP program is a finite set of MuTACLP *clauses*, i.e., of clauses of the form:

$$A \alpha : - C_1, \dots, C_k, B_1 \alpha_1, \dots, B_n \alpha_n$$

where  $A, B_1, \dots, B_n$  are atoms,  $\alpha, \alpha_1, \dots, \alpha_n$  are optional temporal annotations, and  $C_1, \dots, C_k$  are constraints.

MuTACLP programs can be combined by means of two meta-level operators, i.e., union  $\cup$  and intersection  $\cap$ . Formally, the operators define the following language of *program expressions*:

$$Exp ::= Pname \mid Exp \cup Exp \mid Exp \cap Exp$$

where  $Pname$  is the syntactic category of program names, each uniquely identifying a MuTACLP program.

The union and intersection operators mirror two forms of cooperation among programs. In the case of union, either program may be used to derive, hence the union of two programs corresponds to put together the clauses belonging to each program. In the case of intersection, both programs must agree at each derivation step. More precisely, intersection allows one to combine knowledge by merging clauses with unifiable heads into clauses having the conjunction of the bodies of the original clauses as body, and the unified head annotated with the greatest lower bound of the head annotations as head.

**Spatial Representation and Spatio-Temporal Correlations.** Spatial information can be represented and handled inside our framework by means of constraints. A spatial object is modeled by a predicate and its spatial extent is expressed by adding variables denoting the spatial coordinates as arguments, and by placing constraints on such variables. For instance, a convex polygon, that can be seen as the intersection of a set of half-planes, is represented by a conjunction of inequalities each defining a single half-plane. A non-convex polygon, instead, is modeled as the union of a set of convex polygons.

Furthermore, the facilities to handle time offered by the language allow one to establish easily spatio-temporal correlations, like time-varying areas, or, more generally, moving objects, supporting either discrete or continuous changes. For instance, a moving point can be modeled by using a clause of the form:

$$moving\_point(X, Y) \text{ at } T : -constraint(X, Y, T)$$

where  $constraint(X, Y, T)$  is a conjunction of constraints involving spatial and temporal variables.

In a similar way we can represent regions which move continuously in the plane. Consider the area flooded by the water tide, and assume that the front end of the tide is a linear function of time (the example is taken from [6]). This time-varying area can be described as

$$floodedarea(X, Y) \text{ at } T : - 1 \leq Y, Y \leq 10, 3 \leq X, X \leq 10, Y \geq X + 8 - T$$

## 4.2 Semantics of MuTACLP

We define the operational (top-down) semantics of the language MuTACLP by means of a meta-interpreter, obtained by enriching the well-known vanilla meta-interpreter for logic programs in order to deal with the annotations and to give meaning to the composition operations.

Following Bowen and Kowalski [4], we employ the two-argument predicate *demo* to represent provability. Namely,  $demo(\mathcal{E}, G)$  means that the formula  $G$  is provable in the program expression  $\mathcal{E}$ . Without loss of generality, we assume all atoms to be annotated with **th** or **in** labels. In fact **at**  $t$  annotations can be replaced with **th**  $[t, t]$  by exploiting the (**at th**) axiom. Moreover, each atom which is not annotated in the object level program is intended to be true throughout the whole temporal domain, and thus can be annotated by **th**  $[0, \infty]$ .



The extended meta-interpreter is defined by the following clauses.

$$demo(\mathcal{E}, \text{empty}). \quad (1)$$

$$demo(\mathcal{E}, (B_1, B_2)) \leftarrow demo(\mathcal{E}, B_1), demo(\mathcal{E}, B_2) \quad (2)$$

$$demo(\mathcal{E}, A \text{ th } [T_1, T_2]) \leftarrow S_1 \leq T_1, T_1 \leq T_2, T_2 \leq S_2, \\ clause(\mathcal{E}, A \text{ th } [S_1, S_2], B), demo(\mathcal{E}, B) \quad (3)$$

$$demo(\mathcal{E}, A \text{ th } [T_1, T_2]) \leftarrow S_1 \leq T_1, T_1 < S_2, S_2 < T_2, \\ clause(\mathcal{E}, A \text{ th } [S_1, S_2], B), demo(\mathcal{E}, B), \\ demo(\mathcal{E}, A \text{ th } [S_2, T_2]) \quad (4)$$

$$demo(\mathcal{E}, A \text{ in } [T_1, T_2]) \leftarrow T_1 \leq S_2, S_1 \leq T_2, T_1 \leq T_2, \\ clause(\mathcal{E}, A \text{ th } [S_1, S_2], B), demo(\mathcal{E}, B) \quad (5)$$

$$demo(\mathcal{E}, A \text{ in } [T_1, T_2]) \leftarrow T_1 \leq S_1, S_2 \leq T_2, \\ clause(\mathcal{E}, A \text{ in } [S_1, S_2], B), demo(\mathcal{E}, B) \quad (6)$$

$$demo(\mathcal{E}, G \text{ wrt } \mathcal{Q}) \leftarrow demo(\mathcal{Q}, G) \quad (7)$$

$$demo(\mathcal{E}, C) \leftarrow constraint(C), C \quad (8)$$

$$clause(\mathcal{E}_1 \cup \mathcal{E}_2, A \alpha, B) \leftarrow clause(\mathcal{E}_1, A \alpha, B) \quad (9)$$

$$clause(\mathcal{E}_1 \cup \mathcal{E}_2, A \alpha, B) \leftarrow clause(\mathcal{E}_2, A \alpha, B) \quad (10)$$

$$clause(\mathcal{E}_1 \cap \mathcal{E}_2, A \gamma, (B_1, B_2)) \leftarrow clause(\mathcal{E}_1, A \alpha, B_1), \\ clause(\mathcal{E}_2, A \beta, B_2), \\ \alpha \sqcap \beta = \gamma \quad (11)$$

A clause  $A \alpha \leftarrow B$  of a plain program  $P$  is represented at the meta-level by

$$clause(P, A \alpha, B) \leftarrow S_1 \leq S_2 \quad (12)$$

where  $\alpha = \text{th } [S_1, S_2]$  or  $\alpha = \text{in } [S_1, S_2]$ .

Clauses (1) and (2) are the ordinary rules to deal with empty and conjunctive goals, respectively. Clauses (3), (5) and (6) handle the case of atomic goal reduction, implementing not only Modus Ponens but also the inference rule ( $\sqsubseteq$ ): the atomic goal to be solved is required to be labelled with an annotation which is smaller than the one labelling the head of the clause used in the resolution step. Rule ( $\sqcup$ ) is implemented by clause (4). As remarked in Section 4.1, this rule is applicable only to **th** annotations involving overlapping time periods which do not include one another. Clause (7) models message passing as a change of context, hence  $G \text{ wrt } \mathcal{Q}$  means that  $G$  has to be solved in  $\mathcal{Q}$ . Clause (8) manages constraints by passing them directly to the constraint solver.

As far as the meta-level definition of the union and intersection operations is concerned, clauses (9) and (10) simply state that a clause  $A \alpha \leftarrow B$  belongs to the union of two program expressions  $\mathcal{E}_1$  and  $\mathcal{E}_2$ , if it belongs either to  $\mathcal{E}_1$  or to  $\mathcal{E}_2$ . On the other hand, a clause  $A \alpha \leftarrow B$ , belonging to the intersection of two program expressions  $\mathcal{E}_1$  and  $\mathcal{E}_2$ , is built by taking an instance of clause in each program expression  $\mathcal{E}_1$  and  $\mathcal{E}_2$ , such that the head atoms of the two clauses are unifiable. Let such instances of clauses be  $cl_1$  and  $cl_2$ . Then  $B$  is the conjunction of the bodies of  $cl_1$  and  $cl_2$  and  $A$  is the unified atom labelled with the greatest lower bound of the annotations of the heads of  $cl_1$  and  $cl_2$ .

### 4.3 The case study

In this section we present the MuTACLP approach to the case study illustrated in Section 2.

In our experiment we focus on two problems: finding out the estimated position of the den whenever its real location is unknown and searching the animals which are likely to be a couple. To this aim we only need a subset of the whole database and we represent the information of interest for the crested porcupine as a collection of facts of the kind:

```
fix(Id,X,Y,Hour) at Date.
```

which specify the spatio-temporal localization of the animal `Id` giving the position `X,Y`, and the time, `Hour` and `Date`, of the bearing.

Let us now formalize the questions we want to cope with. From the expert knowledge we know that the crested porcupine stays in its den during the day whereas it usually spends the night far from it. Hence, in order to determine the position of the den we collect all the fixes of the animal between an hour before dawn and an hour after sunset, since in this time period the animal is probably close to its den. On the other hand, to determine if two crested porcupines are a couple we exploit the notion of contemporary fixes: two fixes are contemporary if they are within a certain distance and in a certain time interval. The effective values have to be chosen by the domain experts. At this point we estimate the number of contemporary fixes for a pair of animals and according to this number the expert can decide whether the pair is a couple or not. Notice that these are typical spatio-temporal queries where we select spatial data depending on temporal information.

In the spirit of our framework, we partition knowledge among different programs. We define four programs: `ANALYSIS` collects the rules that implement our analysis criteria; `SUN` provides the predicates that allow to determine the dawn and the sunset in the period of interest; `DATA P` contains the data of the bearings; finally `AUX` gathers together the definitions of auxiliary predicates used in the computation. In this way the analysis rules are separated from the specific data, thus the program `ANALYSIS` can be reused to perform reasoning on different sets of data.

Let us first present the program `ANALYSIS`.

`ANALYSIS`:

```
possible_loc(Id,Lloc) at T :-
  constr( findall(loc(X,Y),demo(dataP+sun+aux,(fix(Id,X,Y,Hour) at T,
    dawn_sunset(Hour) at T)), Lloc)).

prob_den(Id,Rad,Prob,L) in [T1,T2] :-
  possible_loc(Id,Lloc) in [T1,T2],
  neighbour_list(Lloc,Rad,Prob,L).

couple_in_day(Id1,Id2,R,S,N) at T :-
  constr(
    findall(c(Id1,Id2),
      demo(dataP+analysis+aux,(fix(Id1,X1,Y1,H1) at T,
        fix(Id2,X2,Y2,H2) at T,
        contem(X1,Y1,X2,Y2,R,S,H1,H2))),
      L)),
  length(L,N).

contem(X1,Y1,X2,Y2,Rad,Sec,H1,H2):- dist(X1,Y1,X2,Y2,D),
  constr(D < Rad, abs(H1-H2) < Sec).

couple(Id1,Id2,R,S,Ratio) in [T1,T2] :-
  couple_in_day(Id1,Id2,R,S,N) in [T1,T2],
```

```

couple_in_day(Id1,Id2,1000000,S,M) in [T1,T2],
constr(Ratio is (N/M))

```

The first and second clauses are used for the den analysis. The first one returns the list of positions `Lloc` of the animal `Id` between dawn and sunset in a certain day, i.e., the positions in the fixes whose hour of bearing is in the part of the day of interest. The second clause extracts from the previous list the positions which are probable dens. The predicate `neighbour_list` selects those positions whose neighbourhood (with radius `Rad`) includes a great quantity of fixes between dawn and sunset. This quantity is estimated by considering the ratio between the number of fixes in the neighbourhood and all the fixes for the given animal in the period of time of interest.

The remaining clauses are aimed at finding the (possible) couples. The predicate `couple_in_day` returns the number `N` of contemporary fixes in a day for the pair of crested porcupines `Id1,Id2`. Two fixes are judged contemporary if their spatial and temporal distance is bounded by `R` and `S` respectively. The predicate `contem` checks whether two fixes are contemporary. Finally, the predicate `couple` returns the ratio between the number of contemporary fixes of the crested porcupines `Id1,Id2` and the number of observations of `Id1,Id2` within `S` seconds at arbitrary distance (concretely this is obtained by setting a very large bound for the distance parameter) in a certain time period.

Notice that the Prolog meta-predicate `findall(X,G,L)` is used to compute the list `L` of elements `X` that satisfy the goal `G`. The predicates `neighbour_list`, `dist` and `length` are defined in the program `AUX`. The predicate `neighbour_list` returns a list of `pden(X,Y,N)` specifying the position `X,Y` of a probable den and the number `N` of fixes (between dawn and sunset) within its neighbourhood, whereas `dist` and `length` compute the distance `D` between two points and the length of a list, respectively.

We complete the presentation by showing the code for the other programs.

SUN:

```

dawn_sunset(Hour) th [T,T] :- light(D,S) at T,
                           between_ds(D,S,Hour).

between_ds(D,S,Hour):- constr(Before_dawn is D-3600,
                              After_sunset is S+3600,
                              Hour>=Before_dawn,Hour<After_sunset).

light(25470,63910) th [[1,1,1998],[31,1,1998]].
:
light(25530,62820) th [[1,12,1999],[31,12,1999]].

```

DATA:

```

fix(f1,62060.0,1669490.0,4724115.0) at [01,01,1998].
fix(f3,62120.0,1669740.0,4724100.0) at [01,01,1998].
fix(f2,76380.0,1669535.0,4724390.0) at [01,01,1998].
:

```

AUX:

```

neighbour_list(Lloc,Rad,Prob,L):-
    neighbour(Lloc,Lloc,Rad,Ltemp),
    length(Ltemp,N),
    select(Ltemp,N,Prob,L).

```

:

In the program `SUN` the predicate `dawn_sunset` checks whether the hour of the bearing is between one hour (3600 seconds) before dawn and one hour after sunset. Notice that, since the dawn and sunset times vary along the year, the predicate `light` records a monthly estimate of

such data expressed in seconds. The program DATAP collects the data on the crested porcupines provided by the biologists: they consist of 4000 fixes concerning 25 different individuals. Finally in the program AUX we show only the definition of one of six auxiliary predicates used in the computation. It is worth noting that time is expressed by dates instead of real numbers. An obvious translation converts dates into numbers before starting the computation.

We now show some queries the biologist could ask the system. We want to know whether the crested porcupines identified as `m15` and `f3` are a couple in the time period `[[26,12,1998],[29,7,1999]]`. According to the suggestions of the experts we set the contemporary parameters to 100 meters and to 15 minutes (900 seconds).

```
| ?- demo(analysis+aux,
          couple(m15,f3,100,900,R) in [[26,12,1998],[29,7,1999]]).
```

The answer is `R = 0.556`, since the contemporary fixes of the two animals are 143 and the number of observations of the two animals within 15 minutes at arbitrary distance is 257. Hence we can conclude that the animals `m15` and `f3` are probably a couple.

To confirm this hypothesis, biologists need to know whether they have the same den in that period of time.

```
| ?- demo(analysis+aux,
          prob_den(m15,100,0.45,L) in [[26,12,1998],[29,7,1999]]).
```

```
L = [pden(1669880.0,4724010.0,17), pden(1669900.0,4724020.0,17),
     pden(1669895.0,4724025.0,17),pden(1669855.0,4724040.0,17)]
```

```
| ?- demo(analysis+aux,
          prob_den(f3,100,0.47,L) in [[26,12,1998],[29,7,1999]]).
```

```
L = [pden(1669920.0,4723990.0,21),pden(1669920.0,4723990.0,21),
     pden(1669870.0,4724000.0,21), pden(1669910.0,4724030.0,21),
     pden(1669920.0,4724000.0,21), pden(1669900.0,4724020.0,21),
     pden(1669915.0,4724030.0,21),pden(1669900.0,4724060.0,21)]
```

Each of the above queries returns a set of probable den locations for the given crested porcupine. According to such answers the two animals have their dens located in an area of 60 meters diameter. Since the radio-tracking technique presents several sources of error, biologists estimated an error of about 62 meters in the provided data and thus the answer does not contradict the fact that `m15` and `f3` are a couple. Indeed, from the *homing-in* data we know that in this time period the two animals have a common den located at `X = 1669960`, `Y = 4724040`, whose distance from the computed probable dens is around 50-80 meters. This is a further proof that the obtained data are good approximations of reality.

## 5 ArcView: the GIS point of view

In order to better understand which are the opportunities and the limitations of the current GIS technology on spatio-temporal reasoning, we experimented ArcView [11] by ESRI to realize a system that satisfies some of the requirements listed in Section 2.3. The aim of this application is to provide the biologist with a system that, on the one hand, tries to automate as much as possible, the basic procedures that the user usually do “by hands”, on the other hand, it offers a set of spatio-temporal analysis functions that the user is usually not able to perform with standard tools provided by GIS. As software, we chose ArcView since it is a leading technology in geographical analysis and one of the GISs most used in the world. ArcView is a desktop solution offered by ESRI and provides data visualization, query, analysis, and integration capabilities along with the ability to create and edit geographic data. We experimented two different versions of this product, ArcView 3.1 and ArcView 8.1, and the results are presented in Section 5.1 and Section 5.2, respectively.

## 5.1 Implementation in ArcView 3.2

The spatio-temporal porcupine application developed with ArcView 3.2 exploits the extension Spatial Analyst (ESRI) that, essentially provides raster capabilities, and the package Movement [19] (USGS), that implements a set of biological/statistical functions such as Home Range estimators. Based on this software, the application has been built using Avenue, the script programming language that allows the developer to customize an ArcView application.

Spatial Analyst offers a set of sophisticated analysis functions enabling ArcView to create query and analyze cell based raster maps, to derive new information from existing data, query information across multiple data layers, the integration of cell based raster maps with standard vector data sources, and create sophisticated spatial models using ModelBuilder [1]. Among others, this extension provides features to create buffers based on distance, to generate density maps profiles and to perform area analysis and analysis of “nearness” of spatial objects. These features are at the basis of the Animal Movement scripts realized by USGS [19] that offer a collection of about 40 functions for the analysis of the movement and behavior of animals. The most popular functions in Movement are probably Home Range estimators (Minimum Convex Polygon - MCP - and the Kernel method), which are statistical functions that provide an estimate of the living area of the animal, as already explained in Section 2.2.

**Description of the system** The system [5] we realized allows the biologists to analyze the collected data and implements the spatio-temporal queries 2, 3 and 4 of Section 2.3, i.e., *Relationship among animal localizations*, *Time step for calculating the home range* and *Spatio-Temporal distance between localizations and events*.

The user interface is a customization of ArcView implemented in Avenue. Besides ArcView menus, two new menus have been added to support the two main groups of functionalities of the system:

- **View features.** These functions provide a number of standard visualization features such as the view of all the fixes of a given animal together with its den or the Home Range of an identified animal. Notice that although all these features are already provided as primitive functions by the GIS software, here their use has been automated by Avenue scripts and made easily accessible by ad-hoc menus. No details will be given about this part of the system.
- **Spatio-temporal analysis.** The most interesting and innovative part of the system supports the spatio-temporal queries introduced in Section 2.3.

Let us now describe in details, the spatio-temporal functions implemented in the system and accessible from the new *spatio-temporal* menu:

- **Analysis of the Movement of an animal.** This function provides an animation of the movement of an animal during the monitored time in the study area. The tool shows the path of the animal in the area as it is given by consecutive fixes. Obviously, this is only an approximation of the path and its accuracy strictly depends on how often the movements of the animal have been traced. The more fixes we have, the more the path will approximate the reality. This function is a customization of the *Animal Movement Path* provided by the package Movement since it offers more information than the original script as it calculates the spatio-temporal distance between each single step. Furthermore it provides the possibility to select a session of the monitoring period. This is particularly useful when the set of data is very large and spread over a long period of time.
- **Movement Analysis of two animals.** This function is an extension of the previous one which shows the movement of a pair of animals, particularly useful to study the behavior of the animals as a whole.
- **Home Range on overlapping time intervals.** The objective of this function is the detection of seasonal variations of the home ranges, both in location and size. To attain this, the idea is to calculate home ranges in a “continuous” way within a given time interval (e.g., a season).

The “continuity” can be obtained by using a temporal *sliding window*. Let  $[t_{start}, t_{end}]$  be the time period of interest, two parameters  $\delta_t$  and  $W_t$  are defined, and home ranges are computed on overlapping time intervals, each of duration  $W_t$  and each shifted of  $\delta_t$  w.r.t. the previous one, starting from interval  $[t_{start}, t_{start} + W_t]$ . In this way, even animal movements at a scale *finer* than the window size produce a detectable change in the home range – the larger and the longer is the animal movement, the larger is the change in the home range, in a continuous way –, and thus distinct movements can be distinguished between each other. Obviously, this computation is quite heavy, depending on the quantity of data available. However, usually this operation is performed only once since the output table is saved and the tool can reuse it without rebuilding it. The user can set three parameters:  $\delta_t$  the width of the time interval,  $W_t$  the shift considered between intervals and *the minimum number of fixes* to be considered in order to calculate the home range in an interval. These values depend on the specific application case and can vary from few minutes, when fixes are collected very often, to several days when fixes are collected over longer periods. The algorithm implemented in Avenue exploits the Home Range function provided by Movement and, in order to perform iterative Home Range calls to the overlapping intervals, it creates a temporary table containing all the sub-intervals inside the period of interest, determined by the input parameters. The output of this function is a new table containing the data of each single home range computed. Having the data stored in a table allows further operations on it such as visualization on screen of the areas, building statistical diagrams and graphs. As an example Figure 10 and Figure 11 represent the home range computed in two consecutive time periods.

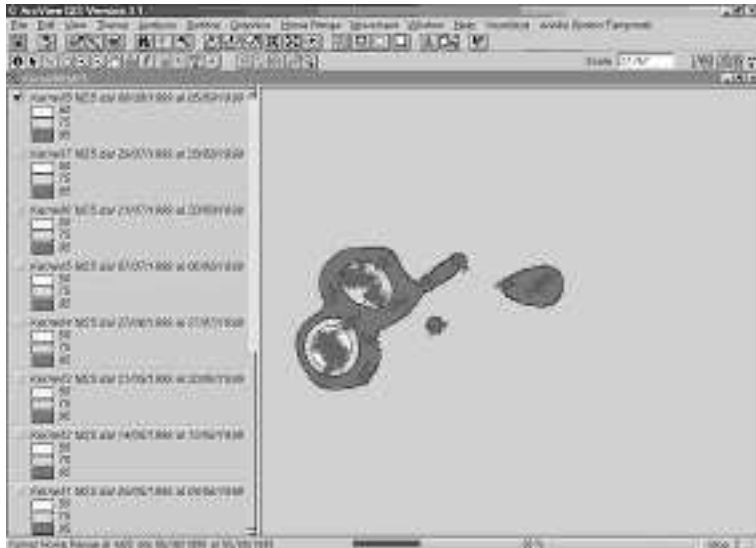


Fig. 10. Visualization of Home Range Variation - Step 1

- **Couples of Animals.** The aim of this query is to study the behavior of couples of animals. As already discussed in Section 2.3 and 4.3, to cope with this problem we consider the contemporary fixes. Indeed here before fixes are selected according to a temporal property, i.e. fixes within a certain time interval, called simultaneous interval, and then also the spatial component is considered. The duration and the scale (seconds, minutes, days) of the simultaneous interval is an input parameter and can be decided by the user depending on the context. In detail, the algorithm first sorts the table containing fixes based on the time information and then scans the records and build a new table containing pairs of fixes falling within the given simultaneous interval. By exploiting this information, a set of functions are defined:

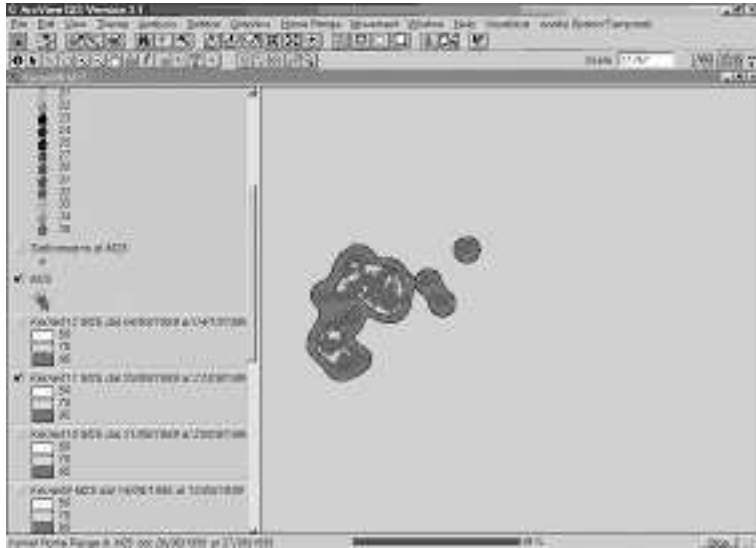


Fig. 11. Visualization of Home Range Variation - Step 2

- **Find all the couples.** This function computes all the couples, i.e. animals which are close to each other *at the same moment*, relating together the spatial and the temporal dimension.
  - **Reciprocal animal avoidance.** This function assumes that an animal avoids another one if their inter-individual distance is always greater than a minimal fixed value.
- **Effect of events on the animal movement.** The system provides functions to study the effect of events on the movement of the animal. Events could be meteorological (e.g., rain, snow) or other events related or not to human activities (e.g., fire, flood, hunting). The function computes all the distances in space and time between a given event and one or more monitored animals. The parameters defined by the user are:
- the radius of the study area, i.e. the maximal distance (in meters) from the event an animal can be affected by the event itself;
  - the time period to be analysed;
  - the temporal duration of the event expressed in number of days.

The output of this function is a table where all data related to the selected event and the animals whose fixes respect the parameters (spatio-temporal distance from the event) are stored. Based on that, the user can ask the system to visualize an animation of the movements of the animals with respect to the event. This is to give “a first feeling” of the movements of animals. At the same time, the statistical available functions offer a more detailed analysis of the event.

As a preliminary result, we found out that, when compared over the same kind of queries, MuTACLP code is much more compact and readable than the corresponding script written in Avenue. In fact, MuTACLP allows for a declarative description of the knowledge of the experts, whereas Avenue is a procedural language which does not provide high-level primitives for the handling of temporal information. Of course, commercial GISs offer many visualization and spatial handling mechanisms that are still not available in our logical language.

To this end we developed a new version of the system exploiting the more recent release of ArcView in order to integrate the GIS with the logical component implementing the deductive language MuTACLP. The resulted system aims at combining the advantages of both systems while reducing the drawbacks. It exploits the graphical user interface and visualization capabilities of ArcView together with the increased expressive power of the spatio-temporal logical component, as it is illustrated in the next section.

## 5.2 Implementation in ArcGIS 8.1

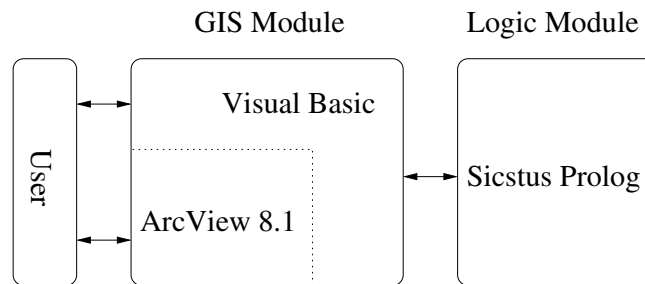
ArcGIS 8.1 is a recent version of GIS by ESRI and it is a completely new product with respect to the previous release. It is an integrated environment that includes ArcView, ArcInfo, ArcSDE and ArcIMS. One of the main features of this new GIS environment is that the programming technology now is based on Common Object Model (COM). This means that ArcGIS supports any language based on COM, such as Visual Basic or Visual C++.

We implemented the porting of the crested porcupine application into this new environment [9, 21] and it is worth pointing out that the application has been redesigned in order to

- implement a new environment where a logic module containing MuTACLP is integrated with the GIS;
- exploit the new features of the GIS platform to better meet the user requirements.

The resulting system is therefore a new integrated architecture where the user can perform analysis on animal behaviour using a customization of the standard GIS functionalities (this part called S.T.I.A. - Spatio-Temporal Individual Analysis) or using enhanced spatio-temporal capabilities provided by the logical module. In particular, STIA application is more general than the previous version since it provides the user with more configuration options (e.g. the specific format of the input data) and the the queries themselves has been designed to analyse each type of individual studied with a radio-tracking system. However, it is worth noticing that since the Movement scripts were not available for this new platform, the queries related to home range studies have not been implemented.

It is worth recalling here that the novelty of the new system is the integrated architecture combining ArcView 8.1 with the logical module, as shown in Figure 12.



**Fig. 12.** The integrated architecture

The user is presented with the ArcView interface from which she/he can interact with the system by two modalities: on the one hand, she/he can use the standard functionalities provided by the system enriched with the STIA application, on the other hand, she/he is provided with a supplementary menu in order to exploit the additional spatio-temporal logical module. Each of the primitive functionalities of the tool can be activated by the user. Furthermore, the implementation of the user requirements proposed by biologists has been realized as a Visual Basic extension called by the appropriate menus “Queries on couples” and “Events” (see Figure 13). The implementation of these functionalities follows the requisites posed by users in section 2 and the lines of the algorithms explained in the previous section concerning ArcView 3.2. Here, the application has been redesigned to better exploit the features of Visual Basic, such as a better integration with MS Windows objects, the use of ArcObjects library provided by ESRI, the use of MS Access for storing spatial objects (geodatabase) and therefore a better implementation of SQL interface. Anyway, it is worth noticing that the main lines of the algorithms have not been changed.

As an example of the new STIA application, Figure 14 shows a screenshot of the animal-events analysis function. Actually, the system displays an animation that shows how animals



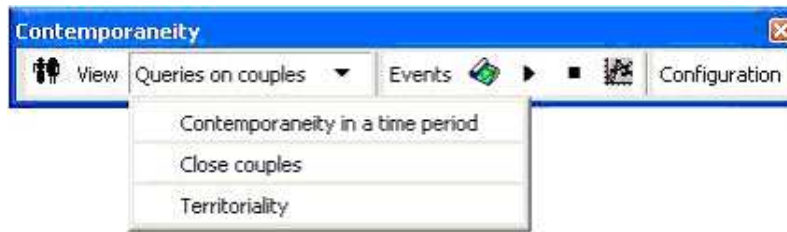


Fig. 13. The menu toolbar

move related to the selected event, thus making easier for the user to have a first impression of the animal behaviour.

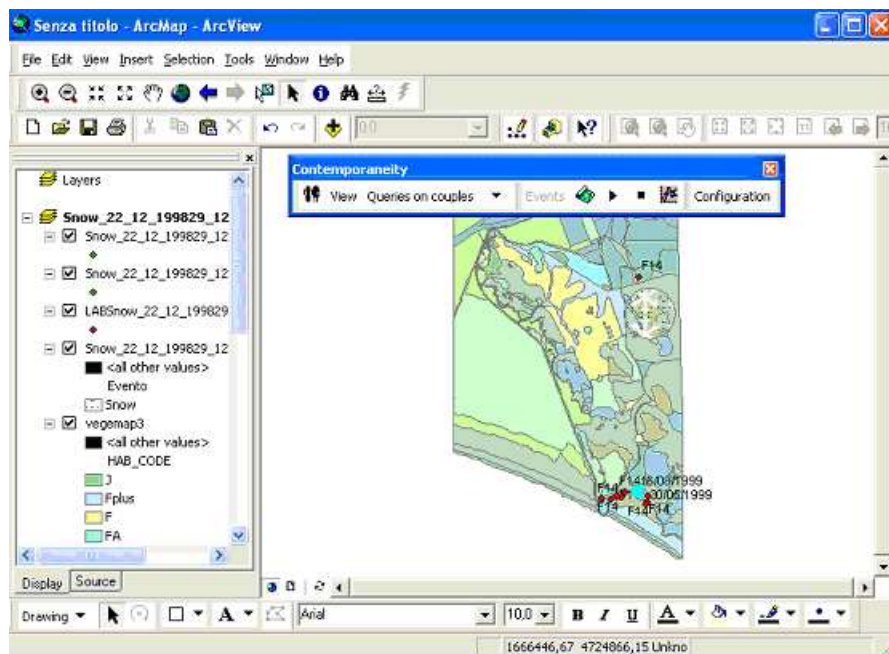


Fig. 14. Screenshot of a query on the relation between animals and events.

Now, we focus on the realization of the integration with the Logical Module that enriches the system with the capabilities of a spatio-temporal inference engine.

The integration of ArcView with the Logic Module has been realized by means of a TCP/IP connection between the Visual Basic part and Sicstus Prolog [23], that implements MuTACLP language. Each command is sent, by means of TCP/IP sockets, from the ArcGIS user interface to the MuTACLP meta-interpreter and finally to the Prolog engine that performs the computation. Whenever an operation on spatial objects is asked for (e.g., the addition of an object to a layer), the request is sent to the GIS that executes such operation. ArcView returns the identifier of the resulting object to the Prolog interpreter, which continues the deduction process. When a solution is found, it is sent back to the GIS for visualisation.

The Prolog part that sends Visual Basic messages simply writes on the standard output, previously set to the opened socket. For example, to send to the GIS the command that adds an

object to a layer, we send to the socket the code of the command ('&6'), the name of the layer, the object type (e.g. point) and the coordinates X, Y:

```
send_command('&6',Outlayer,point,X,Y) :-  
    write('&6'),nl,  
    write(OutLayer),nl,  
    write('point'),nl,  
    write(X),nl,  
    write(Y),nl,  
    flush_output.
```

The Visual Basic program that receives such a command consists essentially of a loop that waits for messages on the socket. Upon receiving a command, it executes the code that performs the corresponding operation on the GIS. The simplified (for the sake of readability) version of such a loop showing only the code that implements the addition of an object in a layer (function '&6') is:

```
Public Sub Parser(ByVal InputString As String)  
    While InputString <> ""  
        cmd = GetArg(Inputstring) 'read command from socket  
        Select Case cmd  
            Case "&6" 'add a new object to the destination layer  
                ' reading parameters from Socket  
                t1 = GetArg(Inputstring) 'layername  
                t2 = GetArg(Inputstring) 'type(point,polyline,polygon)  
                t3 = GetArg(Inputstring) ' X  
                t4 = GetArg(Inputstring) ' Y  
  
                < If the database does not exist, create it.  
                Then create the object (point,polyline,polygon)  
                and insert a record (feature) in the featureClass >  
  
                < Other cases >  
        Wend  
    End Sub
```

## 6 Conclusions

In this paper we presented a complete case study concerning the analysis of animal behaviour, especially focusing on spatio-temporal aspects which are typical of the animal ecology domain. We formalized this application from different points of view, including the conceptual model obtained by using the MADS formalism, the logical model using MuTACLP in order to express spatio-temporal queries. We have realized a prototype that performs the required analysis using recent commercial GIS technology. Finally, we have ported this system into the new ArcView version that offers enhanced customization technologies supporting languages such as Visual Basic or Visual C++. Exploiting these new capabilities we realized an integrated system that combines the standard GIS functionalities with the spatio-temporal reasoning capabilities provided by MuTACLP.

## Acknowledgments

We would like to thank Dr. A. Sforzi of the Ethology and Behavioural Ecology Group (GEEC) headed by Professor Sandro Lovari (Dipartimento di Biologia Evolutiva, University of Siena).

## References

1. ESRI - Arc View, Spatial Analyst Extension <http://www.esri.com>.

2. P. Baldan, P. Mancarella, A. Raffaetà, and F. Turini. MuTACLP: A language for temporal reasoning with multiple theories. In *Computational Logic: Logic Programming and Beyond*, volume 2408 of *LNAI*, pages 1–40. Springer, 2002.
3. A. Belussi, E. Bertino, and B. Catania. An extended algebra for constraint databases. *IEEE TKDE*, 10(5):686–705, 1998.
4. K.A. Bowen and R.A. Kowalski. Amalgamating language and metalanguage in logic programming. In *Logic programming*, volume 16 of *APIC studies in data processing*, pages 153–172. Academic Press, 1982.
5. A. Brandini. Analisi spazio-temporale con gis commerciali per lo studio del comportamento animale, May, 2001. Master thesis, University of Pisa.
6. J. Chomicki and P.Z. Revesz. Constraint-Based Interoperability of Spatiotemporal Databases. *Geoinformatica*, 3(3):211–243, 1999.
7. M.T. Corsini, S. Lovari, and S. Sonnino. Temporal activity pattern of crested porcupines hystrix cristata. *Journal of Zoology*, 256:43–54, 1995.
8. A. Felicioli and L. Santini. Burrow entrance hole orientation and first emergence time in the crested porcupine hystrix cristata l.: space-time dependence on sunset. *Polish Ecological Studies*, 3/4(20):317–321, 1994.
9. N. Grasso and A. Isolani. Un sistema che integra basi di dati geografiche e ragionamento spazio temporale, 2002. Tesi di Laurea, Dipartimento di Informatica, Università degli Studi di Pisa.
10. S. Grumbach, P. Rigaux, and L. Segoufin. Spatio-Temporal Data Handling with Constraints. In *ACM GIS*, pages 106–111. ACM Press, 1998.
11. <http://www.esri.com>. *Arc View 3.1 User manual*.
12. R. Kenward. *Wildlife radiotagging*. Academic Press, 1987.
13. M. Lucherini. Attività e uso dello spazio nell’istrice hystrix cristata, 1996. Tesi di Dottorato di Ricerca, Università degli Studi di Siena.
14. P. Mancarella, G. Nerbini, A. Raffaetà, and F. Turini. MuTACLP: A Language for Declarative GIS Analysis. In *DOOD2000*, volume 1861 of *LNAI*, pages 1002–1016. Springer, 2000.
15. P. Mancarella, A. Raffaetà, and F. Turini. Temporal Annotated Constraint Logic Programming with Multiple Theories. In [25], pages 501–508. IEEE Computer Society Press, 1999.
16. C. Parent, S. Spaccapietra, and E. Zimanyi. Spatio-temporal conceptual models: Data structures + space + time. In *Proceedings of 7th ACM Symposium on Geographic Information Systems, GIS’99*, 1999.
17. C. Parent, S. Spaccapietra, and E. Zimanyi. Spatio-temporal information systems: A conceptual perspective, November 2-6, 1999. Tutorial, Eighth International Conference on Information and Knowledge Management, (CIKM), Kansas City, Missouri.
18. G. Pigozzi. Crested porcupines hystrix cristata within badger setts meles in the maremma natural park, italy. *Säugetierkundliche Mitteilungen Band*, 33(2/3):261–263, 1986.
19. W.M. Eichenlaub P.N. Hooge and E.K. Solomon. Animal movement extension to arcview. ver. 2.0. Alaska Biological Science Center, U.S. Geological Survey, Anchorage, AK, USA. 1997.
20. A. Raffaetà and C. Renso. Temporal Reasoning in Geographical Information Systems. In *Advanced Spatial Data Management (DEXA Workshop)*, pages 899–905. IEEE Computer Society Press, 2000.
21. A. Raffaetà, C. Renso, and F. Turini. Enhancing GISs for Spatio-Temporal Reasoning. In *ACM GIS02*, pages 35–41. ACM Press, 2002.
22. L. Santini. The habits and influence on the environment of the old world porcupine hystrix cristata l. in the northernmost of its range. In *Atti del Congresso Ninth Vertebrate pest conference, Fresno, California*, pages 149–153, 1980.
23. SICS. *Sicstus Prolog User’s Guide*, 1995.
24. S. Sonnino. Spatial activity and habitat use of crested porcupine, hystrix cristata l. 1758 (rodentia, hystriidae) in central italy. *Mammalia*, 2(62):175–189, 1998.
25. S. Spaccapietra, editor. *Spatio-Temporal Data Models & Languages (DEXA Workshop)*. IEEE Computer Society Press, 1999.
26. G.C. White and R.A. Garrott. *Analysis of wildlife Radio-Tracking data*. Academic Press, Chicago, 1990.
27. B.J. Worton. Kernel methods for estimating the utilization distribution in home range studies. *Ecology*, 1(70):164–168, 1989.