CoGIS: a cooperative geographical information system

Wesley W. Chu Hua Yang Kuorong Chiang Berthier Ribeiro Gladys Chow
Computer Science Department
University of California at Los Angeles
Los Angeles, CA 90024
e-mail: wwc@cs.ucla.edu and hua@cs.ucla.edu

ABSTRACT

Knowledge-based cooperative query answering facilities have been introduced to the GIS for approximate query answering. We use context-sensitive spatial type abstraction hierarchies to provide such cooperative capabilities. As a result, the ranges of the approximate spatial operators can be automatically determined by STAHs, rather than specified by domain experts. Thus these operators are context-sensitive. Further, STAHs can be generated automatically from geographic databases and thus is scalable. A prototype system - CoGIS - has been implemented to incorporate these cooperative capabilities.

1 INTRODUCTION

Geographical Information Systems (GIS) are gaining popularity in recent years. A conventional GIS incorporates digitized maps with database systems. This integration of the geographical/spatial and factual information in GIS greatly facilitates the information retrieval and analysis. In particular, it is useful in a wide range of applications dealing with geographical and topological data, such as logistical planning, oil/mineral exploration, environmental monitoring, archeology, epidemiology, and so on.

Early research on GIS is primarily concerned with the methodologies of efficient integration of maps and databases. Map-based spatial operators are later introduced to improve the query answering capabilities. Such operators are *area*, *length*, *centroid*, *perimeter*, *located*, *within*, *contain*, *union*, *intersection*, and *difference* [1, 6]. In addition, approximate spatial operators, such as *near-to*, *between*, and *south-of*, are also studied and implemented in some systems [9, 13]. The advent of knowledge based systems promises many new ways to enhance GIS. In this paper, we describe a knowledge-based GIS which provides cooperative query answering, particularly, it provides approximate answers when no exact answers are available.

The rest of the paper is organized as follows. First, we describe an implementation of GIS in Section 2. Then the cooperative query answering is introduced in Section 3. Section 4 shows how to relax geographical queries when no exact answers are available. Section 5 presents a set of context-sensitive approximate spatial operators resulted from the use of cooperative methodologies. Finally, Section 6 discusses the benefits of our cooperative GIS.

2 A GEOGRAPHICAL INFORMATION SYSTEM

A geographical information system allows a user to pose queries based on geographical objects such as countries, states, cities, airports, rivers, *etc.* Further, it provides spatial operators for constraining the geographical queries. In this section, we discuss the basic elements of a geographical information system.

2.1 Geographical Objects and Representations

There are three types of geographical objects: *points*, *lines*, and *areas*. For example, cities, schools, or airports can be shown as points on a maps. Highways, streets, or rivers can be lines; countries, states, or lakes can be areas. These objects can be represented in one of the two forms:

1. *Raster*: raster representations use a set of small and regular grids (or pixels). Each grid has a value to represent its status: a point, a line, a lake, *etc.*

2. *Vector*: vector representations use the geometric Cartesian coordinate system, specifically, *longitude* as X and *latitude* as Y.

We use the vector representation in our system. A point is represented as a (X, Y) pair. A line is represented by a sequence of points that define a set of line segments approximating the actual shape of the line. An area is represented by a set of line segments that forms a closed loop approximating the shape of the area.

We apply the object-oriented technique to implement the geographical objects. With this technique, more specific object types are derived from the three basic types, such as *country*, *state*, *city*, *airport*, *highway*, *etc.* They are the real entities that can be found in the databases. Since the object-oriented technique supports inheritance, derived classes also have all the features defined in their parent classes including the spatial features from the classes such as points, lines and areas. Thus, these objects have both geographical/spatial information from the map and specific factual information from databases. Further, the encapsulation capability provided by the object-oriented technique allows the differences of the various objects to be hidden so that they can be manipulated in a similar manner. Thus, each object can be easily selected, retrieved, and displayed on the map. This greatly facilitates the querying process.

2.2 Spatial Relationships and Operators

An important feature of these objects is that they are frequently *spatially* related. For example, a city is always in a state, which in turn is in a country. A river can go through several states. Two states are adjacent to each other. These spatial relationships can be used to constrain geographical queries as well as to infer more geographic information. Thus, they are often used in geographical queries as *direct* or *indirect* spatial conditions: such as "*find the populations of those states that River X goes through (or intersects)*," where the states are not specified but can be inferred from the information about River X. Using spatial relationships can significantly enhance the querying capability.

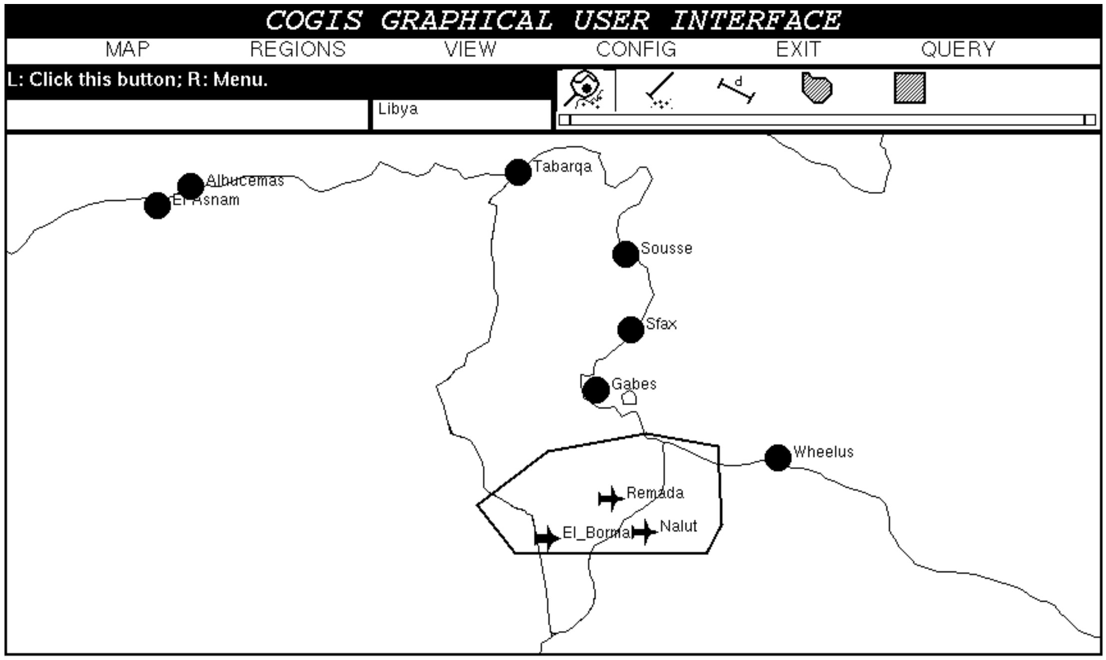

Figure 1: Find all the airports in a user-specified area.

Some of the relationships are precisely defined and can usually be inferred or calculated from existing geographical information based on geometry theory [12]. Our system provides the following spatial operators to express these spatial relationships:

- *located*: point vs point.
- *within*: point vs line/area; line/area vs area.
- *contain*: area vs point/line/area.
- *intersect*: line/area vs line/area.
- *union*: area vs area.
- *difference*: area vs area.

These spatial operators can be used in our system to specify the spatial conditions in geographical queries. In particular, it allows the user to draw a polygon on the map and ask "*find all the airports in this area*," Since this polygon is an arbitrary area, there may not be any pre-defined real entities in the databases to match it. Thus, the system has to take several steps to process this query. First, it creates a rectangular bounding box to replace the polygon, and then poses the query based on the bounding box. After answers are returned from the databases, it only returns the ones *within* the polygon (see Figure 1). Further, with these spatial operators, geographical objects can be dynamically created when needed. For example, a query such as "*find all the cities in Tunisia*" retrieves all the cities in Tunisia from databases, creates the city objects to record the answers and related information, and displays the objects on the map.

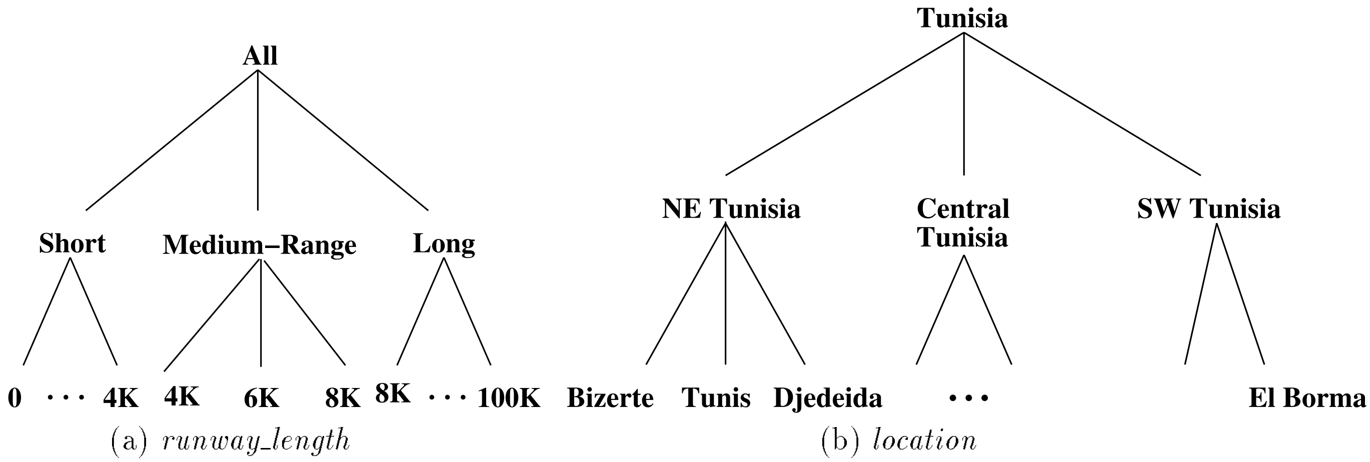

Figure 2: Type Abstraction Hierarchies

3 COOPERATIVE QUERY ANSWERING

In a conventional database, if an exact answer is unavailable or if a query is not well-formed with respect to the schema, the database returns a null answer or an error, which is usually not what a user wants. A database system can be made to be much more resourceful and cooperative by permitting conceptual level queries (containing concepts that may not be expressed in the database schema) when the user does not know the precise schema, providing approximate answers when requested data is missing, or volunteering associative (relevant) information to the query answer. A knowledge-based system – CoBase [2, 4] – is being developed to incorporate these cooperative query processing capabilities.

3.1 Query Relaxation Via Type Abstraction Hierarchies

CoBase can provide approximate answers when no exact answers are available. To do so, it relaxes query conditions by utilizing the domain knowledge – Type Abstraction Hierarchies (TAHs). TAHs represent the approximation knowledge at different levels of abstraction. For example, the *Medium-Range* (*i.e.*, from 4,000 to 8,000 ft.) in the TAH of *runway-length* (in Figure 2) is a more abstract representation than a specific runway length in the same TAH, *e.g.*, 6,000 ft. Abstractions of several attribute values can be *combined* to form abstraction of tuples. For example,

```
airport(NE-Tunisia, Medium-Range, Runway-Width:Wide-Range, ...)
```

is an abstract representation of

```
airport(Bizerte,   6000 ft,   50 ft, ...)
```

A query can be modified by relaxing the query conditions. First, an attribute value from the query condition is selected for relaxation; it is located at the bottom of its TAH. The query condition is relaxed by *generalization* (moving up the TAH) and *specialization* (moving down the TAH), *e.g.*, from 6000ft to Medium-Range to 4000ft, 6000ft, or 8000ft. TAHs provide a structured way for query relaxation which is different from the rule-based approach [5]. Moreover, we have developed methodologies [3, 10] to automatically construct TAHs from databases. Thus, this knowledge structure for query relaxation is scalable to large information systems.

3.2 Cooperative Operators

Although CoBase uses relaxation when no answer is found, query modification may also be specified explicitly by the user through a set of cooperative operators. For example:

- *approximate operator* ($\wedge$) relaxes the specified values within the approximate range that is predefined by the user. For example, $\wedge$9am will be changed to the interval (8am, 10am). Further, (3pm,$\wedge$5pm) can be replaced by (3pm, 6pm).

- *similar-to* requests a set of objects semantically similar to the given object. The similarity is computed over a set of query-specified attributes. Thus, *similar-to* is a multiple-attribute operator with a weight assigned to each attribute in accordance with the relative importance of that attribute. The similarity of each object is evaluated against the given object by calculating the weighted mean-squared error over the list of the attributes [11]. The similar objects are listed in ranked order, along with their measures.

- *Nonrelaxable list* specifies the query conditions that may not be relaxed.

- *relaxation-order* specifies the order that query conditions should be relaxed. For example, an airport's runway-length can be asked to relax before relaxing the runway-width.

- *Unacceptable lists* allow users to inform the system not to provide certain answers. This can be accomplished by trimming parts of the TAH from searching. For example, *avoid airlines X and Y* tells the system that airlines X and Y should not be considered during relaxation. It not only provides more satisfactory answers to users, but also reduces search time.

4 GEOGRAPHICAL QUERY RELAXATION

We have implemented a prototype of GIS on the top of CoBase. This cooperative GIS (CoGIS) provides regular GIS functionalities. In addition, when there are no exact answers for a geographical query, both its spatial and non-spatial conditions can be relaxed to obtain the approximate answers. For the non-spatial conditions, they can be relaxed by the method described in Section 3. In this section, we will describe the methodology to relax spatial conditions.

4.1 Spatial Relaxations

As we discussed in Section 2, spatial conditions are usually expressed by the spatial operators. Each operator has a distinct meaning, thus it should be relaxed differently from others. In the following, we present several examples of spatial relaxation.

- *Location relaxation*: the distance between two objects o_1 and o_2 can be represented by their centroids, $\rho(o_1,\ o_2)$, and their angular distance, $\theta(o_1,\ o_2)$. To relax the spatial location of o_1

and o_2, ρ and θ can be relaxed to $\rho + \Delta\rho$ and $\theta + \Delta\theta$, where $\Delta\rho$ and $\Delta\theta$ are the relaxation ranges.

- *Containment relaxation*: an object o_1 containing another object o_2 can be represented by the objects' area ratio, $\eta = \frac{A(o_1)}{A(o_2)}$, and the distance between their centroids, $\rho(o_1, o_2)$, where $A(o_i)$ is the area of object o_i. To relax the containment feature, ρ and η may be relaxed to $\rho + \Delta\rho$ and $\eta + \Delta\eta$, where $\Delta\rho$ and $\Delta\eta$ are the relaxation ranges for ρ and η.

- *Intersection relaxation*: an object o_1 overlapping with o_2 can be represented by the overlap ratio (the ratio of overlapped regions over the union of both regions) $\zeta = \frac{A(o_1 \cap o_2)}{A(o_1 \cup o_2)}$, and the distance between their centroids $\rho(o_1,\ o_2)$, where $A(o_1 \cap o_2)$ is the intersection region between o_1 and o_2, and $A(o_1 \cup o_2)$ is the union of the areas covered by o_1 and o_2. To provide intersection relaxation on o_1 and o_2, ζ and ρ can be relaxed to $\zeta + \Delta\zeta$ and $\rho + \Delta\rho$ where $\Delta\zeta$ and $\Delta\rho$ are the ranges of relaxation.

The relaxation ranges such as $\Delta\rho$ and $\Delta\theta$ can be manually determined by a domain expert. Since it is application and context dependent, improper selection of the relaxation ranges may produce too many answers or none at all. For example, if we cannot find a hospital at location X, we can dilate the search area by a default radius such as 50 miles. This default value may not be sensitive to the context: if the location X is in a rural area, we could not find any hospitals within 50 miles; if the location is in a city area, we may find too many hospitals. Thus, two different ranges are needed for the rural and city situations respectively. But it is impossible to provide context-sensitive ranges for every situation. To remedy this problem, we use the Spatial Type Abstraction Hierarchies (STAHs).

4.2 Spatial Type Abstraction Hierarchies

In CoBase, we have developed a method to construct TAHs based on the data distribution [3]. We extend this method to create STAHs. A STAH differs from a regular TAH in that it is based on multiple attributes rather than a single attribute. For example, the spatial distance between objects can be more precisely represented by (ρ, θ). As a result, more accurate relaxation, $(\Delta\rho, \Delta\theta)$, can be provided by a STAH. Similarly, the STAHs for containment relaxation can be generated based on the containment relationship ρ and η which depends on the domain and application context. Thus, they provide more expressive capability for spatial relaxation.

More importantly, a STAH is constructed based on the data distribution, making it context-sensitive. The clustering method recursively partitions the objects into smaller spaces. The final sizes depend on the local density of the objects: the denser the objects, the smaller the spaces. To relax a spatial condition, the space containing the object is first identified in the STAH. Then, all the objects in that space are returned as the approximate match or the approximate answers to the query. Using STAHs, therefore, provides a more accurate and context-sensitive relaxation.

For example, let us consider the query: "*find an airfield at the city Sousse.*" Since there is no airfield located exactly at Sousse, this query must be relaxed to obtain approximate answers. First, we locate the city Sousse with latitude 35.83 and longitude 10.63. Using the STAH in Figure 3, we find Sousse is covered by Area 4. Thus, the airport Monastir is returned. Unfortunately, it is not an airfield. So the query is further relaxed to the neighboring cluster – the four airports in Area 3 are

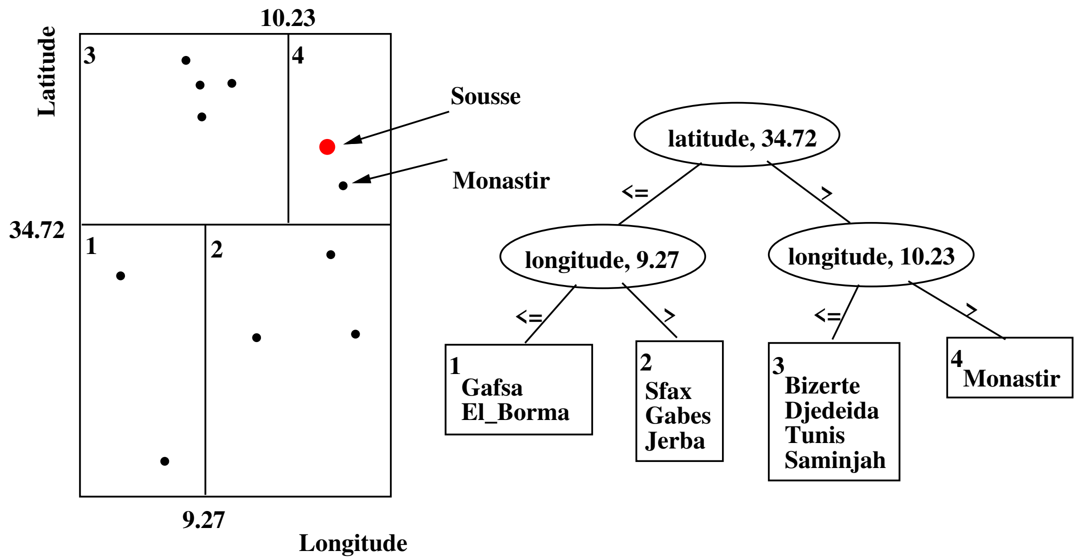

Figure 3: A STAH for the airports in Tunisia and its corresponding two-dimensional space.

returned: Bizerte, Djedeida, Tunis, and Saminjah. Since only Djedeida and Saminjah are airfields, these two will be returned as the approximate answers. In the STAH, the default neighborhood distance for each area can be determined by the maximum pair-wise distance between the airports in the area. Because STAH is generated based on the distribution of airports, the default neighborhood distance determined in this way is context sensitive: the denser the airport location distribution, the smaller the distance among the airports. In Figure 3, for example, the default neighborhood distance in Area 3 is *smaller* than the one in Area 1.

Because the spatial measure is subject dependent, we need to construct STAHs for every type of object. For example, hospitals and airports are different objects thus need different hierarchies. Based on applications, the user can pre-select the objects before generating the STAHs. The pre-selection condition can vary from simple (*e.g.*, all the hospitals in LA) to complex (*e.g.*, all the hospitals near highway I-10 in West LA that have a facility for heart surgery). These conditions are based on applications, and thus are simple to specify.

The scalability of the system can be further enhanced by the re-use of existing STAHs through generalization and composition. This characteristic is particularly useful for constructing STAHs for a cluster of objects where the STAHs of the sub-clusters already exist. For example, suppose we want to generate STAHs for West LA, and the STAHs already exist for the cities of West LA (Westwood, Santa Monica, etc). By generalizing each city in West LA on (latitude, longitude), the STAHs for the individual cities can be merged into a larger STAH for West LA. We also provide STAH editing facility so that individual STAHs can be 'customized', making the entire STAH more context and user sensitive.

5 CONTEXT-SENSITIVE APPROXIMATE OPERATORS

Often, approximation can be very useful in describing spatial relationships. For instance, it is common to say that *the bookstore is near to the university.* Approximate spatial operators are developed for the approximate relationships. CoGIS supports approximate spatial operators such as *near-to*, *between*, *south-of*, *north-of*, *east-of*, and *west-of*.

The difficulty of implementing such approximate spatial operators is the *impreciseness* of these operators – their meanings vary depending on *contexts* and *domains* [9]. For example a hospital *near to* LAX is different from an airport *near to* LAX. Likewise, the nearness of a hospital in a metropolitan area is different from the one in a rural area. A simple and common approach to this problem is the use of pre-specified ranges. But it requires experts to provide such information, thus making it difficult to scale up to larger applications or to be extended to different domains. Subramanian and Adam [13] propose an AI approach to automate this process. However, their analogical-reasoning approach transforms an approximate spatial operator into another equally-imprecise operator, such as *near-to* into *short-drive* or *short-flight*.

As we discussed in Section 4, we have developed a method to cluster objects based on data distribution in STAHs. When a set of hospitals are clustered based on their locations, the ones in the same cluster of the STAH are much closer to each other than to those outside the cluster. Thus they can be considered *near-to* each other. Clustering of the STAH is based on the distribution, thus it is *context-sensitive*. Since the knowledge acquisition process is domain independent and fully automatic, our approach is scalable. We can apply the same approach to other approximate spatial operators, such as *between* (*i.e.*, a cluster *near-to* the center of two objects), *south-of* (*i.e.*, a cluster in the south side of the given object), *etc.*, and provide spatial relaxations as well.

The integration of spatial operators and cooperative operators in GoGIS provides more expressiveness and context-sensitive answers to the GIS. For example, the user is able to pose such queries as "*find the airports similar-to LAX and near-to City X*" in CoGIS. To process this query, it requires the processing of both CoGIS and CoBase. First, CoGIS finds a set of airports *near-to* City X by using the STAH based on the airport location. CoBase then ranks the answers based on the similarity criteria. When there are no answers available, both *near-to* and *similar-to* can be relaxed based on the user's preference. To relax *near-to*, airports from neighboring clusters in the STAH are returned. To relax *similar-to*, the multiple-attribute criteria are relaxed by their respective TAHs.

6 DISCUSSION

Cooperativeness in geographic databases was studied in [8]. A rule-based approach is used in their system for approximate spatial operators as well as query relaxation [7]. For example, they define that "P is *near* to Q iff the distance from P to Q is less than *n*length_unit*, where *length_unit* is a context dependent scalar parameter, and *n* is a scalar parameter that can be either unique for the application and thus defined in domain model, or specific for each class of users and therefore defined in the user models". This approach requires the parameters (*i.e.*, *n* and *length_unit*) be set by domain experts. Thus it is difficult to scale up.

Our CoGIS/CoBase uses a structured knowledge base — TAHs and STAHs, which can be generated automatically from databases based on contexts. It also provides a structured way to relax queries. Thus it is scalable to large applications. Further, the correlation coefficient for each node is calculated during the construction of TAHs and STAHs. It can be used to evaluate the quality of relaxations and to rank the answers in accordance with the nearness to the exact answer.

TAHs/STAHs can be expressed with concepts. Given the feature distribution of ρ and η, we can construct a STAH for the intersection relationship with concepts (or classes) such as *slightly overlapped*, *largely overlapped*, *skewed overlapped*, *overlapped in the north region*, *etc.* These concepts can be used for conceptual queries, making the system more expressive. We can also create different TAHs/STAHs for different contexts. For example, it is critical to distinguish a friendly airport from an enemy airport. The use of a STAH for friendly airports restricts the relaxation only within the set of friendly airports, even though some enemy airports are geographically nearby. This significantly improves the accuracy and flexibility of a GIS. Further, TAHs and STAHs can be edited by users or domain experts to suite the applications. Because each TAH/STAH is context-sensitive, editing a particular TAH/STAH does not affect other TAHs/STAHs – *e.g.*, updating the STAH of the friendly airports does not change the STAH of the enemy airports. It ease the knowledge maintaining and updating.

Our CoGIS can display approximate answers with icons on a map and graphically show the relaxation process. It allows a user to enter queries graphically and incrementally. For example, the user can first ask a query "*find hospitals near-to location X*" by pointing to a location X on a map. Several hospitals may be retrieved from databases via STAHs. Their objects are created dynamically and shown up on the map. Then, the user can select a hospital and continue to ask the next query "*find the shortest route from X to this hospital.*" Such interactive querying capability allows an average user to decompose a complex query into a set of simpler and easier-to-ask queries, which is convenient to the user.

7 CONCLUSION

Integrating CoBase methodologies to the GIS yields the following valuable features:

1. provide context-sensitive approximate spatial operators such as *near-to*, *south-of*, and *between* via STAHs.

2. provide a structured way for the relaxation of such spatial relationships as locations, containments, interactions via STAH.

3. provide the non-spatial cooperative operators (*e.g.*, *similar-to*) to the GIS, thus improving the expressive power of the query language.

4. provide graphic and increment querying capabilities. As a result, we are able to decompose a complex query into a set of simpler queries, greatly improving the effectiveness in query answering.

The STAHs can be automatically generated from databases. Since they are context-sensitive, any changes in a STAH only affect locally. Thus, the knowledge maintenance cost is lower than that of a rule-based system. We have constructed a prototype CoGIS for a transportation domain. Our experience reveals that cooperative methodology provides a valuable enhancement to the GIS, and yet it only requires a moderate cost.

8 ACKNOWLEDGEMENTS

This research was supported by ARPA contract N00174-91-C-0107. We thank Jason Robbins and Henrick Yau for their implementation efforts.

References

[1] S.K. Chang, C.W. Yan, D.C. Dimitroff, and T. Arndt. An intelligent image database system. *IEEE Transactions on Software Engineering*, May 1988.

[2] Wesley W. Chu, Qiming Chen, and Rei chi Lee. Cooperative query answering via type abstraction hierarchy. In *Proceedings of the International Working Conference on Cooperating Knowledge Based Systems*, October 1990.

[3] Wesley W. Chu and Kuorong Chiang. A distribution sensitive clustering method for numerical values. Technical Report 93-0006, UCLA Computer Science Department, 1993.

[4] Wesley W. Chu, M. A. Merzbacher, and L. Berkovich. The design and implementation of cobase. In *Proceedings of ACM SIGMOD 93*, pages 517–522, Washington D. C., USA, May 1993.

[5] F. Cuppens and R. Demolombe. How to recognize interesting topics to provide cooperative answering. *Information Systems*, 14(2):163–173, 1989.

[6] R.H. Guting. Geo-relational algebra: A model and query language for geometric database systems. In *Proceedings of the EDBT conference*, 1988.

[7] A.S. Hemerly, M.A. Casanova, and A.L. Furtado. Cooperative behaviour through request modification. In *Proc of the Tenth International Conference of the Entity-Relationship Approach*, San Mateo, CA, USA, 1991.

[8] A.S. Hemerly, A.L. Furtado, and M.A. Casanova. Towards cooperativeness in geographic databases. In *Proc of the 4th International Conference on Database and Expert Systems Applications*, Prague, Czech Republic, 1993.

[9] D.M. Mark and A.U. Frank. Concepts of space and spatial language. In *Proceedings of the 9th International Symposium on Computer-Assisted Cartography*, pages 538–556, Baltimore, MD, April 1989.

[10] Matthew Merzbacher and Wesley W. Chu. Pattern-based clustering for database attribute values. In *Proceedings of AAAI Workshop on Knowledge Discovery*, 1993.

[11] A. Motro. Vague: A user interface to relational databases that permits vague queries. *ACM Transactions on Office Information Systems*, 6(3):187–214, 1988.

[12] F.P. Preparata and M.I. Shamos. *Computational Geometry: An Introduction*. Springer-Verlag New York Inc., New York, 1985.

[13] R. Subramanian and N.R. Adam. Ill-defined spatial operators in geographic databases: their nature and query processing strategies. In *Proceedings of the Geographical Information System Workshop*, Washington, DC, October 1993.