UTILISATION OF QUALITATIVE SPATIAL REASONING IN GEOGRAPHIC INFORMATION SYSTEMS

Carl P.L. Schultz, Timothy R. Clephane, Hans W. Guesgen, Robert Amor

Department of Computer Science

University of Auckland, Auckland, New Zealand

Abstract

Spatial reasoning is a fundamental part of human cognition, playing an important role in structuring our activities and relationships with the physical world. A substantial body of spatial data is now available. In order to make effective use of this large quantity of data, the focus of GIS tools must shift towards helping a user derive relevant, high quality information from the data available. Standard GIS tools have lacked focus in this area, with querying capabilities being limited, and requiring a user to have specialised knowledge in areas such as set theory, or Structured Query Language (SQL). A fundamental issue in standard GIS is that, by relying entirely on numerical methods when working with spatial data, vagueness and imprecision can not be handled. Alternatively, qualitative methods for working with spatial data have been developed to address some key limitations in other standard numerical systems. TreeSap is a GIS application that applies qualitative reasoning, with a strong emphasis on providing a user with powerful and intuitive query support. TreeSap's query interface is presented, along with visualisation strategies that address the issue of conveying complex qualitative information to a user. The notion of a relative feature is introduced as an alternative approach to representing spatial information.

Introduction

An immense volume of spatial data is now available [1]. Modern GIS commonly provide powerful tools that allow a user to manipulate, query and view this geographic information, however, many limitations have emerged relating to user interaction and query expressiveness [1,2]. To make effective use of the spatial information available, it is not enough that GIS simply display data to a

user [1,2]. People need accessible, intelligent query tools that allow the extraction of specific, relevant information from the raw data provided [1,2]. Standard GIS tools have lacked focus in this area, with querying capabilities being limited, and requiring a user to have specialised knowledge in areas such as set theory, or Structured Query Language (SQL).

A fundamental short coming of current GIS is that they rely entirely on numerical approaches when working with spatial data [1,2]. People find numerical methods non-intuitive, for example, a statement such as "The café is at latitude 23 minutes, 8 degrees, and longitude.." is far less natural than "The café is opposite the art gallery on Symonds St" [1,3]. Further to this, numerical approaches cannot handle uncertainty in information, despite uncertainty being an intrinsic property of information about the physical world [3]. For example, it is impossible to define the boundaries of a coastline with absolute numerical accuracy, due to physical limitations of measurement precision, and the issue of information becoming out of date [4,5]. Another example is the inherent vagueness in a statement such as "The Forest is *near* the Pond". Despite this, humans still reason with imprecise and vague spatial information [3].

In everyday situations, humans often reason about spatial information in a qualitative manner, in particular, working with uncertainty [1,5]. A number of formalisms have been developed that apply qualitative techniques to reason about space. These approaches have been strongly influenced by Allen's qualitative temporal logic [3,4,6]. Allen presents a set of thirteen atomic temporal relations that describe relationships between time intervals. He describes key attributes for effective qualitative reasoning, that have been extended to the spatial domain [5]:

- The logic must handle *imprecision* in the data, given that people often express spatial information in a relative manner, with no reference to an absolute coordinate [5].
- *Uncertainty* in the data must be handled, so that a partial relationship between two features is accepted by the calculus, if the exact relationship is not known [5].

Freksa in [7] presents a generalised approach to Allen's temporal logic, by introducing semi-intervals as the basic reasoning unit, along with the notion of conceptual neighbours [7]. This approach supports reasoning with incomplete temporal information, and reduces the computational effort required during the inference process, by compacting the underlying knowledge base [7].

In [8] we introduce a one-dimensional spatial logic directly based on Allen's original temporal logic [8]. The central idea is to represent relative spatial relationships between objects rather than using absolute object positions [8]. This approach is extended to represent spatial relationships of higher dimensions by

using an n-tuple of relations between each pair of objects [8]. Each component of the tuple represents a different dimension of the modelled scene [8].

Region Connection Calculus (RCC) proposed by Randell et al. [9] is another approach to qualitative spatial reasoning. RCC describes relationships between different spatial regions based on their topological properties, and is thus independent of any coordinate system [9]. Regions are defined to be the primitive type, with a primitive relation 'X connects with Y': C(X,Y) [4,9]. RCC8 is a system which selects a set of eight of these basic relations, such that the set covers any possible physical relation between two regions, and such that there can be no physical relation which is an instance of two basic relation types [4,9].

While the above approaches address the issue of imprecision, and attempt to provide a more human-friendly system for working with spatial data, they do not typically address the issue of vagueness in spatial relations [3,4]. In order to overcome the limitations of either exclusively numerical or qualitative approaches to spatial reasoning, AI techniques have been applied, such as fuzzy logic, to qualitative formalisms [3,4]. By applying fuzzy logic, the formalisms manage both imprecision and vagueness in spatial relations, and allow qualitative relations to be combined with numerical data [3,4].

Qualitative formalisms can extend a standard GIS by providing more intuitive, sophisticated and powerful querying tools. The primary aim of this work is to show that key issues in GIS (non-intuitiveness, imprecision and vagueness) can be resolved through the use of qualitative spatial reasoning techniques in a software application, and that these formalisms are suitable for practical application to real geographic information.

Qualitative Proximity Formalism

Qualitative methods are a coarser, language based approach to working with information, and have been used to specify spatial relationships and properties [3,4,5]. The Qualitative Proximity (QP) formalism is an adapted version of the Fuzzy Proximity formalism described in [10], and is used to reason about **distance relationships** between spatial objects. The possible relationship types, in order of increasing distance, are: touching, very near, near, moderately near, moderately far, far, very far. Figure 1 illustrates two example relationships between a pair of objects, A and B.

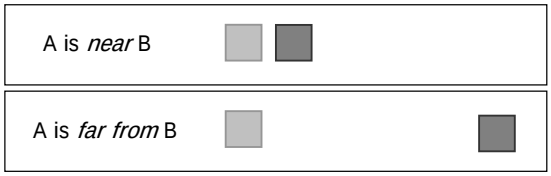

Fig. 1. Subset of the distance relationships defined in QP, where A and B are objects or regions

To address the issue of vagueness in spatial information, we have combined qualitative methods with fuzzy logic [3,4]. To illustrate this, consider the following query: "Find all objects *near* A". As shown in Figure 2, a "near" membership value is assigned to every distance relationship that A shares with some other object, indicating how closely each relationship matches the "near" relationship type. More generally, the standard alpha notation can be used [3,4,10], where α_0 indicates the highest possible membership (a value of 100%, where the relationship is definitely considered a "near" relationship), and:

$$\alpha_1 > \alpha_2 > \alpha_3 > \ldots$$

indicating decreasing membership values, where the exact values of α_1, α_2, α_3,… can be determined according to the application [3,4,10].

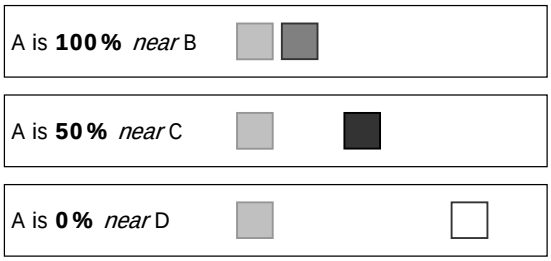

Fig. 2. Results of the query "Find all objects *near* A". A fuzzy membership value is assigned to every relationship that A shares with another object, representing how well each relationship matches the definition of "near". A is considered definitely "near" B, however A is definitely not "near" D. A is considered partially "near" C.

Fuzzy membership values are thus assigned to relationship types of the QP formalism using the conceptual neighbourhood approach proposed in [3,4]. Membership grades are assigned to relations according to the distance the relation is from a reference relation in the conceptual neighbourhood graph [3,4]. The further away a relation is from the reference relation, the lower its membership

grade [3,4]. Figure 3 illustrates the assigning of membership grades to relations with respect to the "near" relationship type.

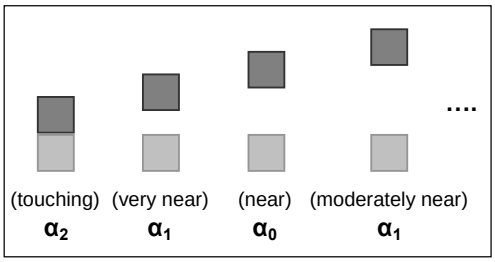

Fig. 3. Extract from the set of QP relations, arranged according to their conceptual neighbourhood. Membership values (alpha notation) have been assigned with respect to the "near" relationship type.

A complete network of qualitative relationships is constructed [3,4], based on the raw numerical data [10]. The first step is to take the distances between each pair of objects. For example, in a 2D scene, this can be accomplished by computing the minimum distance between each pair of objects, a and b, using:

$$\text{distance}(a,b) = \min\left(\sqrt{(x_b - x_a)^2 + (y_b - y_a)^2}\right)$$

The next step is to normalise the distance values to a number between 0 and 1. Thus, a decision needs to be made as to what value is considered a "near" distance. The normalised distance value (d) is then transformed into a fuzzy membership value. If the normalised distance value is 0, then the two objects are touching [10], resulting in a fuzzy value of 1. As the normalised distance value increases, we consider the two objects to be increasingly further apart [10], resulting in a decreasing fuzzy value (between 1 and 0). The following function [10] was used in this case, however, any function with similar properties may also be suitable:

$$\text{Fuzzy membership value} = \frac{1}{1+d^2}$$

This fuzzy value is then mapped to one of the seven QP relation types (from "touching", to "very far away"). The set of qualitative relationships (between every pair of objects) makes up the complete relationship network, which can then be referred to, in order to facilitate more advanced query support [3,4].

GIS Interface and Usability

Usability and human-computer interaction is a key aspect in introducing qualitative spatial reasoning into GIS [1]. A significant part of the limited usability of current tools in GIS is that they are not intuitive to the predominance of users [1].

To extract any specific piece of information from a large dataset, the user must be able to express arbitrarily complicated queries. That is, if the query tool is not expressive enough, certain users will not be able to present the appropriate criteria to the system. To ensure that **all** users can express their criteria, it is best to not place any restrictions on the complexity of a query. Further to this, the query tool must be intuitive and simple to use, to ensure that it is accessible to all users, not just experts. Many issues arise when a user must provide complicated input:

- A query may be malformed, such as a syntax error in SQL.
- The input may be erroneous or semantically nonsensical, such as incorrect data types being entered into a field.
- A challenge is also in teaching a user how to operate a query interface that allows arbitrarily complex input, at the same time minimising the potential for a misunderstanding of the system. The user thus requires constant feedback, and reassurance that the intended query is accurately represented by their actual input.
- A user may be required to learn and remember numerous commands and keywords (such as “select”, “where”, “drop index” from SQL), increasing learning time, along with the chance of a misunderstanding or a syntax error.
- A direct reflection of the underlying formalisms is also desirable, as it allows a user to develop an accurate understanding of how the spatial information is being managed. A user can then benefit from the full potential of the formalisms.
- A further issue relates to the structuring of a query. If a complicated query has poor structure, or has a format that is too general, then the query may become either ambiguous, or far too difficult to understand. On the other hand, if a query format is too strict, then it may lose the desired expressiveness.

The TreeSap GIS application was produced to address these issues, by demonstrating a qualitative reasoning approach along with different visualisation methods. TreeSap’s querying and visualisation approaches are discussed in the following sections.

TreeSap – Qualitative Reasoning GIS

TreeSap (Topographic Reasoning Application) is a desktop GIS application, that provides powerful spatial reasoning tools, with a strong emphasis on usability. TreeSap was produced as an example of how the interface into a powerful query tool can be intuitive and simple to use, without requiring an understanding of mathematics, computer science, or artificial intelligence. Effectively conveying a mixture of qualitative and numerical information to a user is also an important issue. Standard geographic information can be extremely detailed. This is further complicated when qualitative and numerical information are combined, as the data can then express uncertainty and imprecision, along with the standard numerical details. TreeSap was also developed to help address this issue, by demonstrating visualisation strategies that convey complex qualitative information to a user.

TreeSap provides standard functionality found in current GIS, including presentation and organisation of geographic data. In addition to this, two qualitative spatial reasoning tools have been introduced: qualitative querying and relative features. This was accomplished by applying the Qualitative Proximity formalism.

Qualitative reasoning has been applied in two stages of the reasoning process; both in how the query is specified, and in how the system determines which features satisfy a query. Queries are specified in a qualitative manner, that is, the criteria which the query consists of are described using qualitative constraints on geographic features. For example a qualitative query might be "Find all Roads *near* all Railways", rather than providing some numerical distance value. Qualitative reasoning is also used in the processing of the qualitative queries. The generated relation networks are used to find solutions to the qualitative queries, while also providing the user with an indication on the viability of the results.

Qualitative Querying

TreeSap allows a user to specify a qualitative query with an arbitrary number of conditions. The query interface, illustrated as a screen shot in Figure 4, places a strong emphasis on being intuitive and simple to operate, while providing a user with the full benefits of the underlying qualitative formalisms. The query tool is natural language driven, attempting to reflect the underlying qualitative formalisms as directly as possible. The user consequently does not require specialised knowledge in areas such as set theory, or SQL.

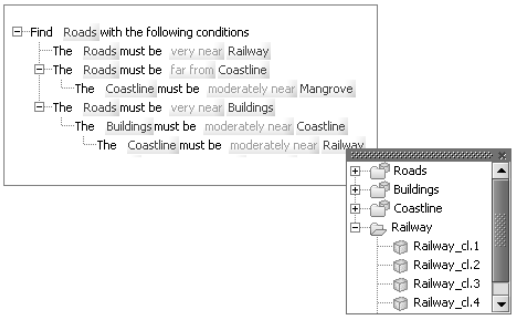

Fig. 4. Screenshot of the interface used to specify a query. The user builds a natural language query tree using the mouse.

The user builds their query as a hierarchical tree structure of conditions, where nodes of the tree are search criteria, or constraints. These criteria are described in terms of a subject, and its qualitative spatial relationship with another object. The query tree allows nested relationships to an arbitrary depth, thus allowing for a query of arbitrary complexity. The hierarchical structure allows the complicated queries to be organised in a natural and intuitive manner.

Each nested condition acts to constrain its parent. For example, consider the condition from Figure 4 "The Roads must be *very near* Buildings". The buildings are, in turn, constrained by the nested condition that states: "The Buildings must be *moderately near* Coastline". A further level of nesting now constrains the coastline: "The Coastline must be *moderately near* Railway".

A query is built up in stages, and at each stage the user is presented with the results of the partial query, visualised on a map. This constant feedback is important, as it allows a user to develop a thorough and accurate understanding of how the query tool works.

The query building process is entirely mouse driven, and thus has a number of usability advantages as follows:

- A query can never be malformed, due to the nature in which it is built. This is not the case for other approaches such as SQL, where the query could have syntax errors.
- There is no possibility for erroneous or invalid input, such as incorrect data types being entered into a field. Input is always valid, thus minimising the opportunity for a misunderstanding to develop.
- The user is given immediate feedback through the mouse-driven interface. When the mouse moves over an interactive component (such as a button), the component lights up, as illustrated in Figure 5. This implicitly suggests to the user that the component is interactive. This is reinforced by a message that explicitly tells the user what interactions the component supports. This feed-

back encourages a user to explore and familiarise themselves with the interface.

- All user communication and interaction (such as component highlighting, tool-tips, popup selection boxes, and messaging) happens near the mouse pointer. This simplifies the query building process, as it is likely that the user's attention will be focused on the mouse. It also reduces user effort, by avoiding large eye and mouse movements between targets.
- Being mouse driven, the interface is easy to operate, as it does not require a user to learn or remember commands or keywords.

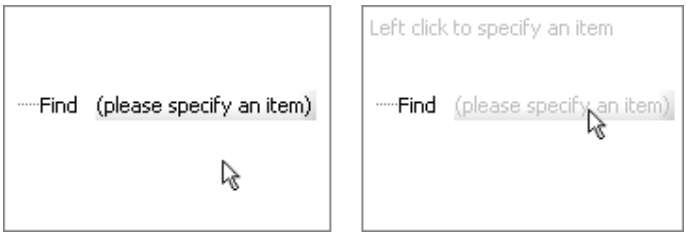

Fig. 5. Screenshots illustrating how interactive components provide the user with feedback. As the mouse moves over an interactive component, the component changes colour (right).

The nature of progressively building a tree which describes each of the user's desired constraints ensures that the query building process is simple, while also providing constant feedback (the results of the query are immediately displayed whenever a condition is specified). Further to this, the use of qualitative formalisms to describe the spatial relationships in the query make it intuitive and easy to learn. Future work will involve allowing a user to combine numerical statements with qualitative statements in a query, for example, "Find all Cafes *near* the Railroad, such that the Railroad is *within 5km* of Downtown". Also, the current query structure uses an implicit AND to join the conditions. This could be extended to include other conditional operators, such as OR, and XOR.

Qualitative Visualisation

Standard geographic data is often large and detailed. The data that results from a qualitative query is even more complicated, with the introduction of fuzzy values on top of the large, detailed datasets. Further to this, there is a need to reflect the innately ambiguous qualitative notions that are present in the underlying formalisms. The key challenge is to present this information to a user in an intuitive manner, while avoiding an approach that requires knowledge in discip-

lines such as mathematics, artificial intelligence, or database languages. TreeSap addresses the issue of conveying complex qualitative information to the user by demonstrating two visualisation strategies: using transparency, and using a display threshold.

Transparency

In the first strategy, transparency is used to represent how well a feature fulfills the query criteria, as demonstrated in Figure 6. Features that fulfill the query criteria to a high degree are displayed completely opaquely, while features that are less relevant to the query are displayed transparently. The level of transparency used is proportional to how poorly a feature fulfills the query criteria.

Fig. 6. Screenshot of the transparency method used to visualise results of the query: "Find all Roads *near* a Specific Building (black circle)".

Using this method, all elements displayed to the user directly convey spatial information. Opaque features represent the solution to the query, and are therefore the most important pieces of information being displayed. These features appropriately attract a user's attention, by being displayed more distinctly than non-solution features. By displaying neighbouring, non-solution features very faintly, the user is implicitly given some spatial context, to assist in the interpretation of the solution. In this respect, transparency offers an intuitive and visually efficient technique for conveying qualitative information.

This strategy presents the user with a static snapshot of the solution, with all the information relating to the query result being provided in a single image. This method is thus ideal if it is required that the query solution be ported onto a hardcopy medium, such as a hardcopy report document, or a newsletter.

Threshold Display

A limitation of the transparency approach is that, while it can provide an instant overview of a query result, it does not effectively convey subtle trends and details. For example, the exact location with the highest solution quality is not always obvious, as subtle differences in transparency can be difficult to recognise. To address this issue, a second approach is proposed that uses a threshold to determine how much information is presented to the user at a given point in time.

Some features fulfill a query's criteria more than others. The notion of a solution quality is thus used to indicate how well each feature meets the given criteria. 100% indicates that a feature meets all the criteria, while 0% indicates that a feature does not meet the criteria in any way. The user can then control a display threshold by dragging a slider. All features that have a solution quality above the threshold are displayed opaquely, and any features with a solution quality that do not meet this threshold are not displayed at all. A scenario is illustrated in Figure 7, where more roads are displayed as the threshold is lowered, revealing an underlying trend.

Fig. 7. Screenshot of the display threshold method used to visualise results of the query: "Find all Roads *near* a Specific Building (black circle)" for differing thresholds.

This strategy is a dynamic representation of a query result, with different parts of the solution being revealed at different points in time. A key aspect of this

method is that the user has control over the dynamic property by adjusting the threshold, thus revealing trends and patterns in the way that the solution unfolds. For example, consider the scenario that a city council is looking for a site to transform into a reserve for growing native New Zealand kauri trees. After applying the appropriate query, it is observed that one small area meets the criteria by 100%, but a much larger area meets the criteria by around 78%. This second part of the solution will be expressed as a small, independent pocket appearing and growing rapidly, once the threshold has dropped to 78%. This suggests that, with minor adjustments, the larger area may be a more appropriate, long term solution. Thus, the display threshold approach offers a deeper understanding of the query solution.

Relative Features

People often describe spatial features in a relative manner [1,2]. Despite this, standard GIS only allow a user to describe the location of a feature with absolute numerical coordinates [1,2]. In order to support a more intuitive method for expressing the location of features, TreeSap introduces the notion of a relative feature. The difference between a standard feature and a relative feature is that the position of a relative feature is described solely by qualitative relationships that it has with other features.

To describe the location of a relative feature, the user builds a relationship tree. This process is based on the procedure for specifying a query, and as a consequence, is also natural language driven. The user defines relationship constraints, such as "The Café is *near* some Roads". The relationships can be arbitrarily complex, for example "The Café is *near* a Coastline, such that the Coastline is *far away* from a Port". Once a relative feature has been defined, TreeSap can search for a possible numerical location that fulfills the criteria given. The feature is then positioned on the map, and the user is provided with a percentage indicating how well the location meets the relationship criteria. Figure 8 illustrates an example where a user is looking for an appropriate location to build a day care centre.

Fig. 8. Defining a day care centre as a relative feature. The relationship constraints that define the location of the day care centre are described using a relationship tree (top). TreeSap can then present a possible instantiation of the relative feature, along with a percentage indicating how well the criteria have been met (bottom).

A key aspect of this method is that it allows a user to determine locations that are consistent with partially defined, or incomplete, qualitative spatial information. For example, consider that a police service uses TreeSap to help isolate the exact position of an emergency. The police station receives a message that an accident has occurred in the Auckland Domain, near the motorway. A user of the application creates a new relative feature to represent this information, and assigns it the label "EMERGENCY_332", along with the partial information relating to its location. TreeSap then attempts to position the relative feature according to the given criteria. As more partial qualitative information is received at the station, the actual position of the emergency is refined. This process would be considerably more complicated if the user could only specify features using numerical coordinates.

This tool provides a user with an intuitive approach to specifying the location of features, compared to standard numerical methods. It allows a user to describe a relative feature with arbitrarily complex relationships, without requiring a user to have specialised knowledge in areas such as SQL or set theory. Further to this, it can handle incomplete spatial information, providing the user with a more powerful system for describing features.

Future Work

Currently, TreeSap can only handle small to medium sized datasets. This is due to limited available memory, as the relations network is stored in local RAM. TreeSap could be extended to handle large datasets by storing the network in a database.

Further qualitative spatial formalisms could also be incorporated into TreeSap. For example, the current system generates a relationship between every pair of features in the data. This will cause problems when moving to larger datasets. A formalism could be implemented that groups the data into clusters, and then layers these clusters, producing more sophisticated relation networks. For example, a query such as "is London near New Zealand?" can be translated into two query steps:

1. Where is London? – England
2. Is England near New Zealand?

This would avoid the need to have inter-layer relationships, such as a relationship between London (a city) and New Zealand (a country), vastly reducing the size of the network.

Another example is the Region Connection Calculus (RCC), which specifies containment relationships between regions, such as "The Wharf is *within* Downtown", or "A *overlaps* B" [4]. This would provide a natural extension to TreeSap's querying capabilities.

The notion of relative features could be extended into a form of automated design. Rather than basing relative features on absolute numerical data, a spatial specification could consist entirely of qualitative spatial constraints between different entities. This abstract specification could then be implemented by a computer system, to determine a number of possible numerical configurations that meet all the criteria to some fuzzy degree. A user could then make small modifications to the automatically generated design, and receive feedback on how well the adjusted designs meet the initial qualitative constraints. For example, this could be used in town planning, product design, packaging, bin packing, and other areas that involve spatial reasoning.

The concepts demonstrated through TreeSap (particularly qualitative reasoning and more natural human-computer interaction) are not restricted to GIS, or even the spatial domain. Future work could involve researching the applicability of qualitative reasoning in a wide variety of different scenarios and disciplines, particularly areas that work with large amounts of data, such as manufacturing, business services, plan verification, bioinformatics, and others.

Conclusions

The amount of stored geographic data has grown significantly [1]. Several key problems exist in standard GIS, due to the reliance on numerical approaches when working with spatial information, including the problems of non-intuitive interfaces (including user controls and data visualisation), and the inability to reason under vagueness and imprecision [1].

The area of qualitative spatial reasoning provides powerful formalisms for reasoning about spatial objects, and their relationships, in a 'natural' and intuitive manner. The application of fuzzy logic allow these formalisms to reason under vagueness and imprecision [3,4]. As a consequence, qualitative reasoning formalisms offer benefits to both usability and querying capability in GIS, as demonstrated by TreeSap, thus resolving many of the existing key issues. The notion of the 'relative feature' offers a completely new approach to representing spatial information, that can be effectively combined with possibly incomplete, absolute numerical data, to derive further relevant information. Qualitative spatial reasoning formalisms are a promising approach for improved power and flexibility when reasoning about, and interacting with, geographic data.

References

[1] Cohn, A. G., Hazarika, S. M. (2001) "Qualitative Spatial Representation and Reasoning : An Overview". Fundamenta Informaticae Vol. 46, No. 1-2, pp. 1-29.

[2] Frank, A. U. (1996) "Qualitative Spatial Reasoning: Cardinal Directions as an Example". Geographic Information Systems, Vol. 10, No. 3, pp. 269-290.

[3] Guesgen, H. W. (2002) "Fuzzifying Spatial Relations". Applying soft computing in defining spatial relations, Matsakis P. and Sztandera L. (Eds.), Physica-Verlag, Heidelberg, Germany, pp.1-16.

[4] Guesgen, H. W. (2005) "Fuzzy Reasoning About Geographic Regions". Fuzzy Modeling with Spatial Information for Geographic Problems, Petry F.E., Robinson V.B., and Cobb M.A. (Eds.), Springer, Berlin, Germany, pp.1-14.

[5] Allen, J. F. (1983) "Maintaining Knowledge About Temporal Intervals". Communications of the ACM, Vol. 26, No. 11, pp. 832-843.

[6] Gooday, J. M., Cohn, A. G. (1994) "Conceptual Neighbourhoods in Temporal and Spatial Reasoning". In Pro ECAI-94, pp. 57-64.

[7] Freksa, C. (1992) "Temporal Reasoning Based on Semi-Intervals". Artificial Intelligence, Vol. 54, No.1-2, pp.199-227.

[8] Guesgen, H. W. (1989) "Spatial Reasoning Based on Allen's Temporal Logic". Technical Report TR-89-049, ICSI, Berkeley, California.

[9] Randell, D. A., Cui, Z., and Cohn, A. G. (1992) "A Spatial Logic Based on Regions and Connection". In Pro KR-92, pp.165-176, Cambridge, Massachusetts.

[10] Guesgen, H. W. (2002) "Reasoning About Distance Based on Fuzzy Sets". Applied Intelligence, Vol. 17, pp. 265-270.