A Framework for Supporting the Application of Qualitative Spatiotemporal Reasoning

Carl P.L. Schultz, Robert Amor
Department of Computer Science, The University of Auckland
Private Bag 92019, Auckland, New Zealand
csch050@ec.auckland.ac.nz, trebor@cs.auckland.ac.nz

Hans W. Guesgen
Institute of Information Sciences and Technology, Massey University
Private Bag 11222, Palmerston North, New Zealand
h.w.guesgen@massey.ac.nz

Abstract

Numerical approaches for representing and reasoning about information are ineffective when data is too imprecise or uncertain. People on the other hand cope very effectively with vague information in daily life, for example when using spatial or temporal information. This has motivated the field of qualitative spatiotemporal reasoning (QSTR), which focuses on coarse, qualitative distinctions between spatial and temporal entities and relations. A substantial body of work has emerged from the QSTR community, however, serious difficulties prevent a uniform and general qualitative treatment of data representing space and time. Without unifying principles there is no basis for comparing the various QSTR approaches, and it is not always clear when and how QSTR should be applied. These issues must be addressed before QSTR can be properly integrated into standard software tools and practices. In this paper the first author's PhD programme is outlined, covering (a) the research aim of developing a framework for supporting the design and implementation of QSTR solutions, and (b) the research approach, which is based around the analysis of case studies, two of which are discussed.

Introduction

Computers and software systems rely on numerical methods for representing and processing information, which work very effectively when data is certain and precise. However uncertainty and imprecision are inherent properties of data that we gather from the physical world, and when probability distributions are unavailable or the numerical precision is not satisfactory, quantitative analysis methods break down. On the other hand people have a remarkable capacity to reason about and operate in the continuously changing physical world, considering that the information we have is necessarily vague and uncertain. In particular, people cope very effectively with everyday phenomena without resorting to detailed numerical analysis of a system or situation [1]. For example people comfortably navigate around their environment throughout the day without taking precise numerical measurements of angles or distances covered. They instead reason and communicate with qualitative concepts such as "turn left", or "walk to the end of Symonds St". A notable and prevalent technical example is the use of asymptotic analysis and the O() notation for investigating algorithm efficiency, where implementation details and input content are abstracted away to achieve a more robust analysis [2]. This approach is called qualitative reasoning (QR) [3], where the aim is to make the smallest number of distinctions between objects and relationships in order to complete a task in a given domain [4].

Specialised qualitative approaches have focused on reasoning about time, resulting in a subfield called qualitative temporal reasoning, designed to manage coarse grained causality, action, and change in a software system. A notable and highly influential example is Allen's interval calculus [5], in which a set of thirteen atomic relations between time intervals is defined, a subset of which is shown in Figure 1. A composition table is provided which gives the possible temporal relations between the intervals t1 and t3 given relations for (t1, t2) and relations for (t2, t3), along with an algorithm for reasoning about networks of relations. For example, if:

- a cargo shipment arrives (t1) before the cargo can be inspected (t2), and
- the cargo is inspected (t2) before the distributors can be contacted (t3), then
- a cargo shipment (t1) must also arrive before the distributors can be contacted (t3).

A natural progression from qualitative temporal reasoning is to consider qualitative spatial reasoning (QSR) [1, 6, 7], where relationships between objects and regions are coarsely defined and reasoned about, concerning topology, shape, orientation, and distance. QSR can be used to answer questions like: "Are there any cafés near the university in Downtown?". However, serious doubts have been raised regarding the possibility of a systematic,

unified approach to QSR known as the poverty conjecture [8], stating the belief that no qualitative description of space exists that can be used to solve tasks in a variety of domains without problem specific metrical information. Despite this, significant progress has been made in a number of subfields, for example, Region Connection Calculus (RCC) [9,10] is a system used to reason about the topological relationships between regions, and in a similar fashion to Allen's interval calculus, defines a set of qualitative spatial relationships that can exist between region pairs. Figure 1 illustrates a subset of these relations. Composition tables are provided, along with algorithms to reason about networks of region relationships.

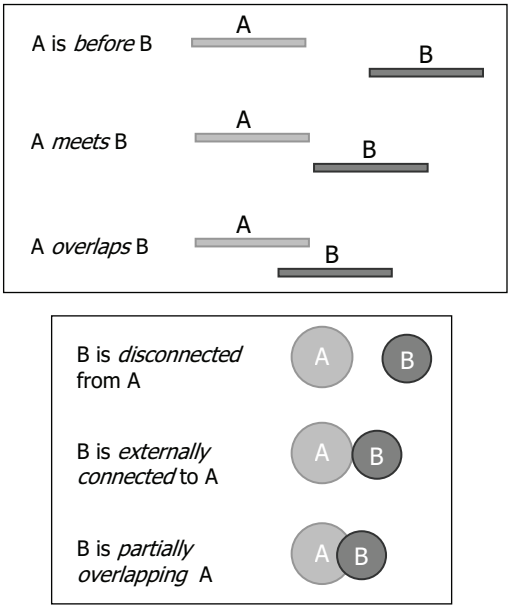

Figure. 1. Subset of the temporal relationships defined in Allen's interval calculus [5] (top), where A and B are time intervals. Subset of the region relationships defined in Region Connection Calculus [9] (bottom), where A and B are regions.

Methods for qualitative reasoning about time or space are collectively known as qualitative spatiotemporal reasoning (QSTR). While a substantial body of theoretical work exists in QSTR, along with a host of industrial applications, a central problem is the lack of a unified framework that provides a standard for the various formalisms and techniques [4]. For example, QSTR formalisms have been developed that work at different granularities, addressing different aspects of a problem, and it is not clear how the various approaches relate to one another, thus making it difficult for researchers to exchange and compare results [4]. The fundamental problem is that the lack of principles and approaches for integrating a QSTR solution with standard software systems [4, 11]. In some cases a qualitative approach will greatly assist in solving a problem. In other cases it may fail to reveal any insights, simply not apply to a domain, have no impact, or even complicate the problem. The first author's PhD research project addresses this issue, with the overall aim of developing a framework for systematising the application of QSTR methods. The framework will be based on problem and QSTR method classification schemes, classification scheme relationships, metrics for quantifying aspects of applied QSTR, and best-practice software architecture design strategies.

Supporting QSTR Software Development

The overall aim of the first author's PhD research is to support the development QSTR in software. The result will be a framework that acts as a practical guide for applying QSTR, aimed at software developers who are assumed to have little or no experience with the qualitative reasoning literature. Three main aspects will be addressed:

1. Making clear which qualitative technique is the most appropriate for a given type of problem
2. Establishing best-practice design methods in terms of software architecture
3. Quantifying the advantages, limitations and drawbacks of the proposed qualitative method, and, where possible, providing a means for measuring the potential benefits.

Objectives and Methodology for Developing the Framework

The tasks that are being undertaken towards the development of the proposed framework are (i) producing classification schemes for structuring the problem domain and the QSTR method domain, (ii) determining the associations between the two domains, (iii) developing metrics for assessing QSTR approaches, and (iv) establishing the most appropriate design strategies for applying qualitative methods in terms of software architecture.

In order to explore the possibilities of applied QSTR to determine the association between QSTR techniques and problems, a number of case studies are being undertaken, along with the analysis of other successful QSTR implementations. Conclusions drawn relating to the appropriate application and implementation of QSTR will direct the development of the proposed framework.

Classification schemes will be developed for classing QSTR methods and the problems that they can apply to. This is a necessary part for identifying which problems or tasks in general can benefit from a qualitative approach, and will be based on the common, salient characteristics shared across many similar problems and QSTR approaches. The schemes will specify which problems can be addressed by qualitative methods and will assist a person who is interested in exploring a qualitative solution. Furthermore, this will provide a platform for other qualitative spatiotemporal reasoning researchers to compare novel methods to existing ones. The sources of the data used for developing the classification schemes are the case studies that have been undertaken (discussed in

Section 3), reviewing QSTR applications in the literature, and reviewing artificial intelligence problem solving literature.

Relationships between the attributes defined in the problem and the qualitative method classifications will be determined to provide a system for associating the two domains. Relationships will be identified by considering the trends in existing qualitative applications, by reviewing qualitative formalisms, and by conducting a deeper analysis of the way in which data associated with a problem is manipulated by the qualitative approaches.

Metrics will be developed for analysing the underlying qualitative formalisms in order to determine the most suitable approach for a given task, and to verify its applicability to the problem. The effectiveness of a qualitative approach must be quantified in terms of the problem being solved so that different qualitative methods can be systematically compared. For example, important factors are the degree to which a problem has been solved and the cost incurred for applying the solution.

Integrating qualitative methods into a task environment requires a clear understanding of the software components that must exist, and how the components must interact. Without information on the best practices for software architectural design, a developer who is applying a qualitative approach may produce software that is inefficient or even faulty. Providing this information will decrease the software design and development time, and will ensure that reliable and consistent implementation results are achieved.

Case Studies

The application of QSTR covers a wide range of disciplines apart from physical systems, including education, economics, and ecological and social sciences. To help classify the various problems that can benefit from a QSTR approach, five application based case studies are being performed covering project management, robotics, astronomy, geographic information systems, and construction IT. The intention is to encounter, first hand, the issues that are raised when attempting to implement the proposed QSTR approaches. Case study analysis is conducted by referring to the current draft classification schemes, which are primarily based on more general artificial intelligence problem solving literature. From these case studies and other literature review based work patterns are being identified and used to refine the classification schemes and the problem and QSTR method domain associations. In the following sections, two of the five studies are presented.

Case Study: Qualitative Query Support for GIS

Modern Geographic Information Systems (GIS) commonly provide powerful tools for manipulating, viewing and querying geographic information, allowing the isolation and informative presentation of relevant spatial features from typically large volumes of data. An effective querying system must provide flexibility, to appropriately capture a user's desired search criteria, and usability, so that the system is appropriately accessible. Despite this, standard GIS querying capabilities are often very limited, (particularly many publicly accessible web-based GIS), or require a user to have knowledge in specialised areas such as Structured Query Language (SQL) or set theory. By relying on numerical analysis techniques, GIS struggle with uncertain and imprecise information. As people communicate about spatial concepts using qualitative

Figure. 2. Screenshot of the transparency method (top) used to visualise results of a qualitative query (bottom). The qualitative terms used to specify criteria ("very near", etc.) capture the concept of vagueness and are accessible to non-experts in GIS.

information it is desirable that a querying system support the use of such information and uncertainty. This application area raises issues regarding human-computer interaction (HCI), reasoning given uncertain and imprecise spatial criteria, and the management of large amounts of data for qualitative spatiotemporal reasoning.

A system called TreeSap GIS [12, 13] is being developed that explores the use of QSR, and demonstrates its applicability towards more sophisticated, yet widely accessible, qualitative query support, as illustrated in Figure 2.

Case Study: QSTR for Subjective Lighting Criteria in Architecture

The discipline of architecture is concerned with more than simply meeting practical criteria, such as: Can the building support the required load? Does the noise level, temperature, or airflow meet the appropriate health and safety standards? Architecture involves the study of how to direct a person's perception of their environment, for example, to evoke a mood, or to convey an abstract concept. This involves managing contradictory requirements that are often difficult to resolve through purely numerical analysis; an example of this is the subjective impression, or atmosphere of a space that can be evoked by lighting.

In such cases, numerical approaches for representing and reasoning about lighting related information are not satisfactory. For example, the level of detail at which processing is being performed is often inappropriate, particularly for early stages of design. Issues regarding usability are raised as an architect, for example, must manually determine whether the desired aesthetic and functional requirements from a lighting configuration are being met, having been given lists of numerical data that can involve a mixture of units (resulting from numerical simulation of a designed model). Thus, issues raised in this application area include the human-computer interaction issue of managing subjectivity, reasoning given vague information, and integrating various vague pieces of information (e.g. "dim lighting, with sharp shadows and striking highlights can evoke a dramatic and sophisticated atmosphere") into a reasoning framework.

A software system is being developed [14] that uses a QSTR engine for analysing a lighting installation and reporting on the subjective impressions that will be evoked.

Classifying Problems and QSTR Approaches

In order to identify and compare different problems, and different QSTR approaches, standard characteristics of the problem domain and the QSTR approach domain must be formally described. The standard characteristics will then provide a basis for the classification schemes. In this section the preliminary formal model for the problem solving process is presented.

Problem solving process

When a person uses a computer to accomplish a task, aspects of the problem are firstly modeled and then the model is reasoned about. For example the general approach for search problems is to model the solution space and then reason about the model by searching for the solution. Another example is the problem (or task) of providing an online hotel reservation service. The hotel information is modeled, along with user requests. This model is then reasoned about and the result of the user request is relayed back to the user. Aspects of the model may change frequently, such as the user requests, while others may change less frequently such as hotel contact details.

The following is a formal definition of the problem solving process.

1. Let a model (m) be a tree graph, where the internal vertices are models, leaves are statements, and branches represent containment (i.e. a model contains statements and models).

2. Let M_S be a model schema that applies semantics to models and statements (typing) and then provides constraints based on the semantics. Let M be the set of all models that adhere to a schema.

3. Let r be a reasoning function that accepts and returns models: $r{:}M \rightarrow M$

4. Let α represent the working model, that is, a variable model that is maintained throughout the problem solving process.

5. The problem solving process is described by a nondeterministic finite state automaton, with states S={startup, model development, model querying, shutdown}, inputs $I=M\cup\{go\_online, terminate\}$, outputs $O=M\cup\{accept\}\cup\{a \mid a \text{ is an assertion on } \alpha\}$, and a customisable state transition function δ: $S\times I\rightarrow S\times O$. A diagram representing the automaton with a general state transition function is illustrated in Figure 3. It must be noted that the reasoning function in the model querying state does not modify the working model α, whereas in the model development state α can change by either augmenting ($y=\alpha$), retracting ($y\subset\alpha\wedge r(x)\subset\alpha$), partially replacing ($y\subset\alpha\wedge r(x)\not\subset\alpha$), or completely replacing ($y=\varnothing$), models in α.

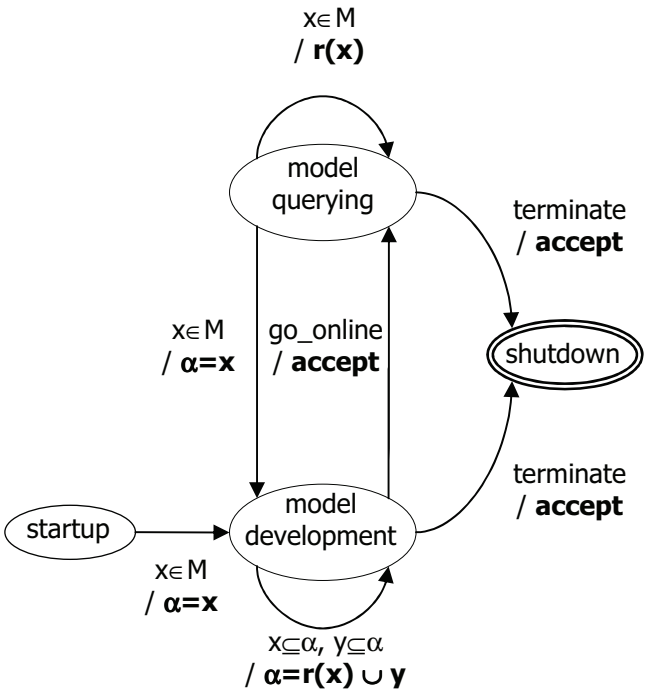

Figure. 3. Diagram of the nondeterministic finite state automaton used to describe the problem solving process. Circles represent states, arrows represent possible state transitions, and the arrow annotations indicate the inputs that cause transition and the outputs, in the format: input / **output**.

Problem characteristics are defined in terms of state sequence patterns and possible state transitions in the automaton. One problem characteristic is the duration of dependency on the working model during the problem solving process. A problem may only have initial dependency, where all the information required for processing is initially available. For example if the problem (or task) is to compile a program (e.g. using a C++ compiler) then all that is required is the source code. Alternatively other problems require information that is not initially available and thus the dependency on the working model is ongoing. For example, if the problem is to interpret a program (e.g. using a BASIC interpreter) then the code to be executed is not initially available to the system.

The notion of model dependency can be formalised by making assertions on the possible state transitions in a problem solving process automaton. A problem has initial model dependency if the automaton can not transition to the model querying state:

$$\forall s \in S, i \in I, o \in O \,.\, \delta(s,i) \neq (\text{“model querying”}, o)$$

A problem has ongoing model dependency if the automaton can transition to the model querying state:

$$\exists s \in S, i \in I, o \in O \,.\, \delta(s,i) = (\text{“model querying”}, o)$$

Other problem characteristics include the state of the working model over its lifetime (once initialised the working model may never change, or it may change either monotonically or non-monotonically), model relationships (simple, where model elements have little or no interaction, and complex, where model elements interact a lot, and strongly depend on each other), model stability (an unstable model changes frequently, whereas a stable model does not), and so on. Formal definitions are being developed for these characteristics.

Conclusions

Qualitative spatiotemporal reasoning (QSTR) is a field of artificial intelligence motivated by the way that people handle vague and uncertain information about spatial and temporal phenomena in daily life. It addresses a number of limitations that arise when a system relies entirely on numerical methods for representing and processing data. A number of successful techniques and formalisms have emerged over the last 30 years, however a lack of design and implementation support, along with questions surrounding applicability, are hindering the field's ability to broaden its scope of application. This PhD research is focused on providing a framework that will tie the various aspects of QSTR together by identifying (a) important characteristics of the problems being tackled with QSTR (b) important characteristics of the QSTR approaches being applied, and (c) the relevant interactions between problems and QSTR solutions. The framework will act as a practical guide for developers who are assumed to be unfamiliar with QSTR literature, in particular (i) making clear which qualitative technique is the most appropriate for a given type of problem, (ii) establishing best-practice design methods in terms of software architecture, and (iii) provide metrics to assess the overall solution quality by quantifying the advantages, limitations, and drawbacks of the proposed qualitative method. Development of the framework is currently being driven by five case studies, each involving the application of a QSTR method to a problem in a particular domain. Two studies were discussed: qualitative query support for GIS and QSTR engine for managing subjective lighting criteria in construction IT.

References

1. Hernandez, D. 1991. Qualitative Representation of Spatial Knowledge. *Lecture Notes in Computer Science* 804. Springer-Verlag, Germany
2. Russell, S.J., P. Norvig, 2003. *Artificial intelligence: a modern approach*. 2nd ed: 977-978. Upper Saddle River, N. J., Prentice Hall.
3. Forbus, K.D. 1996. Qualitative Reasoning. In: Tucker, A.B. ed. *The Computer Science and Engineering Handbook*, 715-733. CRC Press.

4. Bredeweg, B., Struss, P. 2003. Current Topics in Qualitative Reasoning. *AI Magazine* 24(4): 13-16.
5. Allen, J.F. 1983. Maintaining Knowledge about Temporal Intervals. *Communications of the ACM* 26(11): 832-843. ACM Press, New York.
6. Hernandez, D., Jungert, E. eds. 1998. Special Section on Qualitative Spatial Reasoning (Guest Editors' Introduction). *Journal of Visual Languages and Computing* 9(1): 1-3. Academic Press.
7. Cohn, A., Hazarika, S.M. 2001. Qualitative Spatial Representation and Reasoning: An Overview. *Fundamenta Informaticae* 46(1-2): 1-29.
8. Forbus, K.D., Nielsen, P., Faltings, B. 1987. Qualitative kinematics: a framework. In Proceedings of the Tenth International Joint Conference on Artificial Intelligence: 430-435. Milan, Italy. Morgan Kaufmann.
9. Cohn, A.G., Bennett, B., Gooday, J.M., Gotts, N. 1997. RCC: A Calculus for Region Based Qualitative Spatial Reasoning. *Geoinformatica* 1: 275-316.
10. Cohn, A.G., Bennett, B., Gooday, J.M., Gotts, N. 1997. Representing and Reasoning With Qualitative Spatial Relations About Regions. In: Stock, O. ed. *Temporal and Spatial Reasoning*: 97-134. Kluwer.
11. Forbus, K.D. 1995. Qualitative spatial reasoning: Framework and Frontiers. In: Glasgow, J., Narayanan, N., and Chandrasekaran, B. eds. *Diagrammatic Reasoning: Cognitive and Computational Perspectives*: 183-202. MIT Press.
12. Schultz, C.P.L., Clephane, T.R., Guesgen, H.W., Amor, R. 2006. Utilization of Qualitative Spatial Reasoning in Geographic Information Systems. In Proceedings of the Sixth International Symposium on Spatial Data Handling: Progress in Spatial Data Handling, 12: 27-42. Riedl, A., Kainz, W., Elmes, G.A. eds. Springer-Verlag, Berlin Heidelberg
13. Schultz, C.P.L., Guesgen, H.W., Amor, R. 2006. Computer-Human Interaction Issues When Integrating Qualitative Spatial Reasoning Into Geographic Information Systems. In Proceedings of the Seventh International Conference ACM SIGCHI-NZ: Design Centered HCI: 43-51. ACM Press.
14. Schultz, C.P.L., Amor, R., Guesgen, H.W., Lobb, B. 2007. A Decision Support Software Tool for Reasoning about the Subjective Impressions of a Lighting Installation. To be presented in CIB Twenty-fourth W78 Conference: Bringing ITC Knowledge to Work.