Handling Vagueness in Ontologies of Geographical Information

by

David Mallenby

Submitted in accordance with the requirements for the degree of Doctor of Philosophy.

UNIVERSITY OF LEEDS

School of Computing

September 2008

Declarations

Some parts of the work presented in this thesis have been published in the following articles. In each case, details of the contributions by myself and other authors are detailed, as well as which chapters these articles were used within.

Mallenby, D., “Grounding a Geographic Ontology on Geographic Data”, *8th International Symposium on Logical Formalizations of Commonsense Reasoning (Commonsense 2007)*, Stanford, USA (2007) 101–105.

Chapters based upon work: Formed initial of parts of Chapters 3 - 6, in particular; using Medial Axis, measuring linearity, segmenting polygons.

Third, A. and Bennett, B. and Mallenby, D., “Architecture for a Grounded Ontology of Geographic Information”, *Second International Conference on GeoSpatial Semantics (GeoS 2007)*, Mexico City, Mexico, (2007) 36 – 50.

My contributions: Discussion of grounding layer, development of definitions, development of software used.

Other author contributions: The Semantics of Vagueness, ontological architecture, development of general layer.

Chapters based upon work: Referred to in Chapter 2 for background. Referred to in Chapters 3 and 5 for some terms. Referred to in Chapter 6 for grounding.

Mallenby, D. and Bennett, B., “Applying spatial reasoning to topographical data with a grounded geographical ontology”, *Second International Conference on GeoSpatial Semantics (GeoS 2007)*, Mexico City, Mexico, (2007) 210 – 227.

My contributions: Principal author, wrote all sections.

Other author contributions: Checking of sections, author of previous work used in paper, vagueness in geography.

Chapters based upon work: Chapter 4 uses the polygon segmentation approaches, Chapter 5 uses the RCC calculation approaches, Chapter 6 uses the results generation.

Bennett, B. and Mallenby, D. and Third, A., “An Ontology for Grounding Vague Geographic Terms”, *Fifth International Conference on Formal Ontology in Information Systems(FOIS 2008)*, Saarbrücken, Germany (2008) 280 – 296.

My contributions: Prototype GIS implementation, contributions to development of Section 4, such as determining requirements and issues

Other author contributions: Overall wording of paper, Sections 1 - 4 (with some input from myself)

Chapters based upon work: Chapter 6 refers to the domain quantification approaches, as well as GEOLOG

Abstract

This thesis presents a novel approach to the problem of handling vagueness in ontologies of geographical information. Ontologies are formal representations of a set of concepts and the relationships that hold between those concepts. They have been proposed as a method of representing geographical information logically, but existing limitations in ontology languages and approaches fail to handle aspects of the geographical domain adequately, such as vagueness.

The technique introduced in this thesis does not seek to remove or ignore the inherent vagueness when reasoning about geographic features, but rather seeks to incorporate it into decisions made about features during this process. By improving the manner in which vagueness is handled in geographical information systems, we improve the usability and the functionality of such systems, and move towards a more natural method of interaction.

A comparison of the principal vague reasoning approaches is presented, to show how there is not at present a universal approach that handles all forms of vagueness. Rather, there exist different forms of vagueness as well as different required outcomes of vague reasoning, which means we should instead consider the problem at hand and determine the most appropriate approach accordingly.

The technique for handling vagueness proposed here is to provide a system for grounding an ontology upon a geographic dataset. This data is assumed to take the form of a set of 2-dimensional polygons, each of which may be associated with one or more labels describing the type of region that polygon represents and the attributes associated with it. By grounding the ontology onto the data, an explicit link is made between the ontology and the data. Thus, vagueness within the definitions at the ontology level can be handled within the context of the dataset used; “large” can be defined in terms of what it means to be “large” in this dataset.

Further, I developed a system that allows querying of the data and returns features through spatial reasoning. This allows the extraction of new features of interest, including constructing new regions in addition to the existing set of regions within the dataset.

Acknowledgements

I would like to thank my supervisor, Brandon Bennett, for all his support and guidance provided throughout the course of this PhD. I would also like to thank my colleagues in the Knowledge Representation and Reasoning Group, particularly Allan Third. The group has provided me with data to use in my project, as well as advice and suggested improvements whenever I needed them.

I would like to thank Ordnance Survey for providing financial funding as well as providing inspiration for this PhD. The Ordnance Survey provided me with help whenever required whilst allowing me to continue my research with minimal interference.

I'd like to thank all the members of Leeds Freestyle Kickboxing, for helping me stay fit and providing me with an outlet for frustrations built up during my research. Thanks are also due to all my friends who provided me with a similar outlet for relaxation via the pub. Special thanks go to Gary Hamilton for various trips around Europe that provided breaks when I needed them most.

I'd like to thank my parents for being so supportive throughout my academic career, encouraging me to continue with my studies and to achieve all that I could.

Finally, very special thanks are due to my partner, Hannah Featherstone, for all that she has done for me during this PhD. This ranges from doing my washing and cooking to let me get on with my work, through to putting up with endless explanations of my research which helped me more than she may ever know.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction

Maps have been used for thousands of years, offering a method of representing geographical data in a useful and understandable format, to aid navigation, amongst other things. At its most basic, a topographic map merely represents the shape of the land, tracing the outlines of countries for example. However, additional layers of information can enhance and expand the usefulness of such maps; for example marking and labelling of towns, road networks and useful buildings, such as hospitals. We are thus accustomed to the notion of maps containing geographical information, whereby we are applying information as a layer on top of the map. This is the basic principle behind geographical information systems (GIS).

In a primitive sense, any such labelled map is a simple form of GIS, since it combines additional layers of information to the topographical data. However, labelling of features is not the only information which can be layered upon data. A famous early example of this was John Snow's well known study of the spread of cholera (Snow, 1855); in his paper on the subject he included a map showing the location of water pumps in a particular district as well as the houses where people struck by cholera lived. This compellingly showed a concentration of cholera cases around a particular pump, backing up other evidence in the paper that infected water supplies were responsible. By representing the information in this manner, Snow was able to put forward ideas in an accessible manner, more so than simply listing the addresses or by graphing the distance to a pump.

In recent years, the study and usage of GIS on computers has become more prevalent. In addition to adding layers of data to maps, GIS now offer a method of reasoning about the data logically. By adding a logical reasoning layer, reasoning about the data and the relations contained within is enabled; for example, the spatial relations between different features. To do this effectively, therefore, a method of representing logically the information and relations contained within the data is needed, to enable queries to be evaluated over the data. Ontologies have been cited as a method of handling this problem (Varzi, 2001b; Fonseca et al., 2002; Guarino and Welty, 2002), by offering a method of defining the concepts within a domain and the relations that hold between them. Thus a geographical ontology may store information on concepts such as 'rivers', 'mountains' and 'roads', as well as relationships such as spatial relations that may hold between such concepts.

However, existing methodologies for construction of ontologies do not adequately handle vagueness, which is inherent to the geographical domain. Indeed, many of the geographical features around us do not have an exact definition and are dependent upon the context in which they are used, with local knowledge and views affecting the definitions. For example, a general definition of the distinction between a 'lake' and a 'pond' is that lakes are larger. However, as noted by the New Hampshire Department of Environmental Services[1]:

> In general, lakes tend to be larger and/or deeper than ponds, but numerous examples exist of "ponds" that are larger and deeper than "lakes". For example, Echo Lake in Conway is 14 acres in surface area with a maximum depth of 11 feet, while Island Pond in Derry is nearly 500 acres and 80 feet deep. Names for lakes and ponds generally originated from the early settlers living near them, and the use of the terms "lake" and "pond" was completely arbitrary.

Related to this is the problem of *individuation*, where, when given a single large feature, a method of specifying smaller parts that are also specific features is required. When presented with an initial dataset, there may be no boundaries or regions contained within. Instead, the dataset in question may, for instance, be a single large polygon. From this, it may be desirable to *individuate* smaller regions within this polygon that represent features that are of interest. Thus, individuation can be seen as the need to pick out features of interest from a larger unified set. Geographical features are often part of a larger feature

[1]New Hampshire Department of Environmental Services, Lake Biology Environmental Fact Sheet BB-49 http://www.des.state.nh.us/factsheets/bb/bb-49.htm (Visited April, 2008)

(Smith and Mark, 1998; Fonseca et al., 2002); for example, a particular water feature such as a ‘river’ or a ‘lake’ is in fact part of a larger water network, or a ‘mountain’ may be part of a larger range of mountains. A method is therefore needed of extracting such features from the larger overall set.

At present, most approaches ignore or attempt to remove this inherent vagueness, which is not ideal. Vagueness is not a defect of our language but rather a useful and integral part. It is a key research area of the Ordnance Survey[2], where it has been noted that GIS does not handle multiple possible interpretations adequately. By improving the handling of vagueness, the functionality of GIS is also improved, allowing vague features to be reasoned about in an effective manner.

This thesis will investigate incorporating vague reasoning approaches into the reasoning stage, enabling vagueness to be handled more effectively. This will be conducted in the form of a case study, investigating the domain of inland water networks, looking at producing a system that takes topographical data as an input and allows logical queries to be run upon this data to generate geographical features. The approach proposed in this thesis is that of grounding the ontology upon the data (Jakulin and Mladenić, 2005; Mallenby, 2007), thus generating an explicit link between the ontology and data levels.

1.2 Motivating Scenario

GeoCrossWalk[3] is a recent project with the intention of storing a database of geographical features with name and location, as opposed to simply storing points. Rather than a city having a point as its stored location, the spatial footprint of the extent of the city is stored instead. This increases the power of searches upon geographical data, as it allows the use of spatial reasoning. An example query might be:

“Find all hotels within 5 miles of the River Tyne.”

Hotels and rivers are features within the GeoCrossWalk database and the system can easily perform a search based upon distance, thus results to this query can be found.

However, the footprints assigned to such features are predetermined by the designers of the system, with no option to modify based upon personal requirements or preferences. Thus the River Tyne will be represented by a fixed polygon, which the user has to accept. The basis of this definition is also not stored, thus the user is unable to ascertain how this

[2]Ordnance Survey Research: Data modelling: http://www.ordnancesurvey.co.uk/oswebsite/partnerships/research/research/data_fuzzy.html (Visited, April 2006)

[3]GeoCrossWalk home: http://http://www.geoxwalk.ac.uk/index.html (Visited, February 2008)

feature was defined in the first place; whether it was based upon specific measurements or just a generally agreed consensus of what constitutes the boundaries.

Suppose the user had their own definition for ‘river’, one which differed dramatically from the definition used in GeoCrossWalk. They use the system to search for rivers in their area, but find that results differ from what they were expecting. Because they do not know what the definition was that they were using, they are unable to work out why they have been given different results. In addition, because they are unable to enter in definitions themselves, they are unable to return results closer to their needs.

A similar example is if they wanted to find a feature that had not previously been marked up but was part of an existing feature. This may be because the designers were unaware of such a feature, or it was not part of their intended usage, and hence was not included. Suppose the user was interested in determining the extent of an estuary, and attempted to search GeoCrossWalk for this. Because estuary is not a pre-defined feature, a feature would not be returned, and the user is once again left unable to rectify the situation.

By grounding the ontology upon the data, situations such as those listed above could be rectified, as there is now a direct link between the data level and the ontology level. The definition of river is now based upon the context of the data, with the definition being made available to (or even written by) the user. If the definition incorporates vague predicates, the user would be able to determine the precision of such predicates in relation to the data, for example stating at what value something is considered ‘large’. Thus the user would be able to work *with* the inherent vagueness, instead of ignoring it.

Further, if another user has a slightly different interpretation of these predicates, they would be able to compare the two to see for example where they are in agreement or where they differ. In fact, a person may not have a single interpretation of vague concepts themselves, but instead may have multiple interpretations that depend upon the context of usage. They could thus compare these different interpretations to note the similarities and differences, and determine which interpretation they feel best suits their present needs. By allowing the entering of their own definitions, the system can expand and include new features previously not included, allowing the user to segment the data into the parts they require. Such features may, in fact, not correspond to recognised geographical features such as rivers, but may be an artificial region that matches some defined requirements of the user.

The focus of this thesis, therefore, is to develop a system that can take topographical data as an input and allow a user to query this data to return geographical features as an output. This requires consideration of how to enter and represent datasets effectively, how to model vagueness within such a system and how to answer queries upon the data in an efficient and accurate manner. The thesis does not focus on producing a geographical ontology, but rather on the mechanism that would allow a geographical ontology to reason about datasets using vague predicates.

1.3 Research Questions

The main goal of this research is to develop a method of incorporating vague reasoning into geographical information systems, by developing a system that would allow an ontology to be grounded upon a topographic dataset. The datasets in question form a set of 2-dimensional polygons, each of which may be associated with one or more labels describing the type of region that polygon represents and the attributes associated with it. I believe that grounding an ontology upon such datasets allows the explicit linking of the two levels, and thus offers a useful method of accounting for vagueness within the dataset. For example, vague predicates can be defined in relation to the underlying dataset, making more precise their meaning without fixing the definition permanently to a particular value.

To achieve this goal, I will first consider the principal approaches to vague reasoning to determine the most appropriate for this situation. I will then develop a system that represents information about the data efficiently, and is able to answer spatial queries about the data to return segmented features. The system will therefore handle all these aspects in a single, cohesive program. The following principal research questions are addressed in this thesis:

- What is the most suitable approach to vague reasoning with geographical data? Is there a single solution that fits all of the domain, or are there different suitable solutions depending upon the problem at hand? How should somebody go about making such a decision, and what requirements should be considered in order to choose the most suitable approach?

- How can topographical data be represented effectively, enabling the grounding of an ontology upon this data? What methods of representation can be used, and what information will this allow us to extract? How can the spatial relations between parts of the data be modelled, and can this be done efficiently?

- How can the ontology be grounded upon the data? What work is required at both levels to ensure the levels can be connected through grounding? What is the most appropriate logical language to use to represent our ontology, and how can this be implemented to work with the data? How can vague reasoning approaches be incorporated into the representation and segmentation of the data? What features can be extracted from the data using this approach?

1.4 Achievements

The work resulted in the development of a system I have named GEOLOG, as it allows geographical data to be reasoned with using Prolog. GEOLOG takes topographical data as an input and allows first order logic queries to be asked about the data, returning regions that match the queries. GEOLOG therefore represents an important connection between the data and ontology levels. The principal achievements of this work therefore are:

- *Development of a method of applying vague reasoning to geographical data:*
  By considering the problems posed by vagueness in the geographical domain, I was able to expand previous work to produce a method of working with the vagueness in the domain, as opposed to ignoring or attempting to remove it. This allowed for potentially more natural and detailed predicates to be used to define our features.

- *Development of a framework for grounding an ontology upon data:*
  The process of grounding an ontology upon data requires work at multiple levels, both in consideration of what predicates need to be grounded and how the data can be represented. The main concentration of this work has been at this grounding level, as this has largely been ignored previously. I have shown the importance of this level, including considering how to represent the data effectively and how to then reason with this data to return regions of interest. Of particular importance to this level is geometric computation, including representing the initial dataset effectively and calculating spatial relations or new regions. The development of efficient and simple methods of geometric computations for this stage is therefore also an achievement of the work.

- *Development of a model checking approach for a changing domain:*
  The domain of regions that are considered may change depending upon the interpretation of vague predicates, thus any method of handling logical queries upon the data needs to be able to handle this. Model checking approaches typically deal with small, constant domains. In this work, I have expanded this to allow GEOLOG to work with evolving domains, whereby regions are created as required to build a

model, which is then used to answer the required query. This allows us to streamline the domain initially, with the model expanding with subsequent queries. The model generated is thus a result of the queries that precede it.

1.5 Thesis Overview

This thesis is organised as follows:

Background: In Chapter 2 I will discuss the background to the thesis, giving an overview of the research that was required to form this thesis. This includes a general consideration of vagueness in the geographical domain, the problem of grounding an ontology upon data, and qualitative spatial reasoning (both the different forms of reasoning and how to apply this to actual data). This chapter will also review some of the existing geographic ontologies to determine what areas they are deficient in. These sections will be expanded upon in later chapters to form my approach to handling the vagueness in the domain being considered.

Approaches to Vague Reasoning: In Chapter 3 I will discuss vagueness in geography in more detail, by considering the two principal approaches to handling vague reasoning; Fuzzy Logic and Supervaluation Theory. I will examine how the two approaches can be applied to the geographical domain, and how their differences will affect their usage. Finally, I shall consider how the two approaches would fare given different forms of data, to determine which would be the most effective approach for my particular problem domain.

Data and Attribute Representation: Chapter 4, together with chapters 5 and 6, forms the core of this thesis. In these chapters, I will look at the steps needed in order to take topological data, collect attributes and apply vague reasoning to return geographical features. In Chapter 4, I will consider the first part of this, which is the representation and storage of the input data in a format that will allow segmentation and reasoning to take place at a later stage. I will consider the problem of extracting a ‘skeleton’ of the data to aid representation, and how I can then store this information efficiently for future purposes.

Attribute Collection and Data Segmentation: In Chapter 5, I will examine how I can collect the required attributes from the data to allow us to segment the data into appropriate regions. In particular, I will consider how I can apply qualitative spatial reasoning to quantitative data, taking into consideration the performance and efficiency of the algorithms used. Whilst the speed of the algorithms used is important, it is not the primary concern of this project. Rather, I will discuss approaches that are easy to understand, whilst performing the required calculations as efficiently as possible. I will then consider what attributes I may wish to collect, and how these would be collected from my data.

Grounding an Ontology upon the Data Level: In Chapter 6, I will examine how I can link the ontology level to the data level by grounding the ontology upon the data. In order to do this, I need to consider the choice of logic language in which to develop the reasoning system in, and the impact these choices will have upon the reasoning process. The choices here are affected by the need to handle spatial queries, thus a language is required that is expressive enough to handle such queries. I will also evaluate possible outputs for the data that would allow integration with other systems and approaches. Finally, I will develop definitions for features that could be extracted from input datasets.

Results of using the System: In Chapter 7, I will examine the system built using the approaches discussed in the previous chapters, to see how effective my approach is at handling vagueness in the geographical domain, by analysing the results of applying the definitions from Chapter 6 to datasets representing inland water networks. The main dataset considered is of the Humber Estuary, where I will evaluate queries to segment the data into features of interest. I will then look at how these results compare with two other datasets, the River Tyne and the Stour-Orwell Estuary. I will evaluate the successes and problems raised with the different data sets, and determine whether my approach is a viable solution.

Conclusion and Future Work: In the final chapter I will summarise the main aspects of this research, highlighting the main contributions presented. I will review the strengths and limitations of my approach, as well as consider how future work could be performed in this area, for example considering how I may be able to apply the approaches detailed here to other parts of the geographical domain.

Chapter 2

Background

2.1 Introduction

In this chapter an overview of the background to the various aspects of this work is presented, which later chapters will expand upon. Section 2.2 discusses vagueness in the geographical domain. Section 2.3 discusses geographical ontologies and considers some of the existing ontologies available. Section 2.4 discusses ontology grounding. Section 2.5 discusses qualitative spatial reasoning and the principal approaches that are used.

2.2 Vagueness in the Geographical Domain

As has been discussed previously (Fisher, 2000; Bennett, 2001b; Duckham et al., 2001; Varzi, 2001b), vagueness is ubiquitous in geographical concepts. Both the boundaries of geographical objects and the definitions of geographical concepts are often vague, as well as resistant to attempts to give more precise definitions. For example, the definition of a river as given by the Oxford English Dictionary (2004) is:

> "A large natural flow of water travelling along a channel to the sea, a lake, or another river."

This is clearly vague, with the most obvious example being the use of 'large', although there are other parts of the definition which are also vague. However, this is not the only definition for a river; some may differ dramatically from this, others may be more or less restrictive. In comparison, OpenCyc[1] is the open source version of Cyc (Lenat, 1995), which is intended to be the largest and most complete general knowledge base in the world. OpenCyc defines river and stream in as:

> "A *River* is a specialisation of *Stream*. Each instance of River is a natural stream of water, normally of a large volume."
> "A *Stream* is a specialisation of *BodyOfWater*, *InanimateObject-Natural*, and *FlowPath*. Each instance of *Stream* is a natural body of water that flows when it is not frozen."

Again, these are vague, and also do not include the restrictions of the water flowing into a particular feature. Yet at the same time, both definitions are perfectly valid within a given context to describe rivers.

2.2.1 Context

Geographical definitions are dependent on the context in which they are made. A nonsensical example of this can be found in *Alice through the Looking Glass* (Carroll, 1872), when Alice meets the Red Queen:

> "— And I thought I'd try and find my way to the top of that hill "
> "When you say 'hill'," the Queen interrupted, "*I* could show you hills, in comparison with which you'd call that a valley."

Although this is nonsensical, examples of the importance of context are prevalent in geography and can often seem strange; the Atlantic Ocean being referred to as 'the pond' springs to mind. A statement such as "the River Tyne is large" may be reasonable in England due to the size in comparison to other rivers in the country, but when compared with the likes of the Mississippi and the Nile, the statement does not seem as reasonable. Another comparison would be claiming that the tallest hill in a country was a mountain, when in comparison to other mountains it is extremely small.

This difference may not be purely based upon the interpretation of one part of the definition such as size, but may actually be through different definitions. For example, in the UK, rivers are usually defined as permanent flows, but in Australia they may not contain water all year round (Taylor and Stokes, 2005). Whilst temporal aspects of rivers

[1]OpenCyc is the open source version of Cyc, a general knowledge base and commonsense reasoning engine: http://www.opencyc.org/ (Visited, April 2007)

are not as important to UK definitions, they are very important to Australia. Also, applying a UK definition to Australian rivers may result in many not being classified as a river, and similarly an Australian definition applied in the UK may mark too many features as rivers. Our definitions of geographical concepts therefore, appear to be dependent on the context in which they are made.

As discussed by Third et al. (2007), geographical data can also be ambiguous. This occurs where there exists a term used by different people to mean different things. For example, the term "estuary" is ambiguous. A widely used formal definition for an estuary is as follows (Cameron and Pritchard, 1963; Pritchard, 1967a):

> "An estuary is a semi-enclosed coastal body of water which has a free connection with the open sea and within which sea water is measurably diluted with fresh water derived from land drainage."

If this definition were to be used, salinity data could be used to classify such features as proposed by Pritchard (1967b). However, there are other definitions that do not necessarily rely upon the salinity of the water, such as the Oxford English Dictionary definition of an estuary (2004):

> "The tidal mouth of a large river."

This definition would instead be based upon tidal data. Also stemming from this, would be a general definition of an estuary as 'a region of the mouth of a river near to the sea'. As the region that is considered the "mouth" of a river is vague, the use of such a definition can create ambiguities, where people agree upon the definition of an estuary but not upon its boundaries because the definition itself contains other vague features. Indeed, applying opposing definitions to classify estuaries can cause disagreement. For example Herdendorf (1990) proposes that the Great Lakes (or regions of) may be considered to be estuaries, if the definition is primarily based upon the morphology of the region. However, Schubel and Pritchard (1990) rebuke this proposal, stating that:

> "In all estuaries, salt, sea salt, is present."

Although the definitions used are similar and will in some cases generate the same results, the lack of salinity in one definition has a dramatic impact upon what is agreed upon to be an estuary. Ambiguity and context are thus related, as establishing the context in which a definition is made can reduce the ambiguity in the classification.

Table 2.1: A comparison of the tallest mountains in the world, depending on how the height of the mountain is considered. For the second and third examples, only the highest in that category and mount Everest are shown for comparison.

measurement type	Everest	Chimborazo	Mauna Kea
Height above sea level / (m)	8,848	6,268	4,205
Distance from Earth centre / (km)	6,382	6,384	-
Height base to summit / (m)	8,848	-	10,200

2.2.2 Measurement of Terms

Continuing the notion of contextual vagueness, there is the issue of vagueness in terms related to measurement. This was noted with a term such as 'large', as this may be dependent on a combination of dimensions or merely a single dominant dimension. An interesting example of this is when considering the world's tallest mountain, as shown in Table 2.1.

When we are asked the question "what is the tallest mountain in the world?", we typically would answer Mount Everest. This is true if our measure of mountains is the highest summit above sea level. However, there are other potential measurements that could be used which are equally valid.

Instead of using sea level as our base for measurement, we could instead use the centre of the Earth. This would mean our tallest mountain was now the one whose summit was the furthest from the centre of the Earth. Because the Earth is not exactly spherical but instead bulges at the equator, Mount Chimborazo's summit is in fact 2km further away from the centre than Everest, despite being 2km less above sea level. Another measurement would be to imagine the Earth without any water, and thus the measurement would go from base[2] to summit. If this measurement was used, Mauna Kea would be the tallest mountain, since part of the mountain is underwater.

As illustrated vagueness can arise with the terms used within our definitions. It is thus important that the meaning of such terms is clear, to ensure it is unambiguous how such a measurement is made and thus what it means to be 'tall' for example.

[2]The base of a mountain may itself be vague, as there is no precise definition for a mountain base. This may coincide with the boundary of a mountain being vague, since the base of the mountain would form the boundary between the mountain and the surrounding area.

2.2.3 Threshold (*Sorites*) Vagueness

The previous example shows how problems can arise when dealing with vagueness in relation to the measurement used and obtaining agreement on this measurement. However, as noted by Bennett (2005), vagueness and ambiguity arise in different forms, with the above example perhaps being classified more as ambiguity (e.g. what is meant by 'tall' in this case). However, we may in fact be in agreement on which measurement we want to use to determine categories, but not the threshold at which we split categories. This form of vagueness is referred to as 'threshold or *sorites* vagueness', in respect to its relation to the *sorites* paradox (discussed in Section 3.2).

Returning to the previous example of mountains; suppose the generally agreed upon measurement of height above sea level was used as the height measurement (and thus Mount Everest is the tallest mountain in the world). The problem now becomes determining whether something is a hill or a mountain, based upon its height above sea level (there are other factors which may be considered such as the steepness of the sides, but for the purposes of this example only height is considered). This is also vague, since there does not exact a distinct height which would serve as a threshold. Therefore, even when limited to an agreed measurement, vagueness may still occur in determining the threshold between vague categories.

2.2.4 Classification Vagueness

Classification vaguness relates to where a region has been demarcated, but it is vague as to what the region corresponds to. In the mountain example previously given, for instance, a region may have already been identified as corresponding to a region which extends above the surrounding terrain (which may form the basis of a hill or mountain feature definition). Classifying this region as either a hill or a mountain remains vague though, given there is no clear distinction between the two. This may be due to *sorites* vagueness (the classification is based upon a threshold which may itself be vague), or other attributes.

For example, suppose a region had previously been marked, that corresponded to a built up area, which may be marked as a town or a city. The classification of this region is therefore vague, as it may be related to one or more aspects of the region. The classification may be related to vague thresholds such as how 'dense' the region is (which may be measured in different ways and also has no clear threshold between something being dense and sparse), or may be related to other aspects such as the presence of particular landmarks such as hospitals or cathedrals.

2.2.5 Boundary and Location Vagueness

As discussed by Hazarika and Cohn (2001), spatial vagueness can also broadly be seperated into two categories; boundary vagueness and location vagueness. With classification vagueness, a region was already demarcated, whereas boundary and location vagueness deal with determining the region in question. With boundary vagueness, the region has a known location but has vague or indeterminate boundaries, whereas with location vagueness, there is uncertainty within the spatial location of the region. The two can be seen to be closely related, since parts of a feature may have location vagueness (they are known to be part of the feature in question but they do not have a known location), whilst the feature itself may have boundary vagueness (the boundary between the feature and the surrounding area may be vague).

For example, with mountains, the exact location may be generally agreed upon as some region surrounding the 'peak' of the mountain (the peak is assumed here to have been decided upon), but the boundary that demarcates this from the surrounding region may be vague. On the other hand, there may exist parts of the mountain that are known to exist (such as ridges), but the location of which is not known. Thus, mountains could contain both boundary and location vagueness.

2.3 Geographical Ontologies

Ontologies are formal representations of a set of concepts and the relationships that hold between those concepts, or as defined by Guarino (1998):

> "A set of logical axioms designed to account for the intended meaning of a vocabulary."

Ontologies are intended as a method of adding meaning to concepts and their relationships by means of logic. This logical layer is intended to make explicit the information contained within a document, allowing a computer, for example, to search more effectively. For instance, suppose we had an ontology of 'cars', with information such as engine size, interior and other such attributes. We may then want to search for what we consider to be a 'fun car', which may for instance be a car with a powerful engine and leather interior. By posing this as a logical query to the ontology, we could return cars that matched our requirements, even if no car had been explicitly marked as 'fun' in the ontology. Thus, an ontology would allow logical queries to search for information beyond the original intention of the data.

The most popular language at present for writing ontologies is OWL[3], and in particular the description logic aligned OWL-DL, as this can be output to reasoners to test for consistency. As it is based upon description logic, OWL-DL is computationally complete and decidable due to the restrictions placed upon it. However, this is not the only language in use, and any formal logic-based language should be capable of producing an ontology, providing it is possible to define concepts and the relationships between concepts.

Several other attempts have been made at defining geographical ontologies, thus, examples of these are considered here. In particular, the handling of the definition of the features "river" and "lake" are compared, as these are basic features we may wish to represent in an inland water ontology. These features are traced through their hierarchy, to determine how they are derived and to understand the underlying definition, including considering the usage of spatial relations within the definitions and ontologies.

2.3.1 Ordnance Survey Ontology

The goal of the Ordnance Survey ontologies as indicated on their website[4] is:

> "To provide both an explicit representation of our organisation's knowledge and a set of increasingly automated operations that allow different datasets to be combined together, by representing them in a semantically meaningful way via ontologies."

The ontologies are written in two parts. First, a conceptual ontology that is readable by humans, which is written in 'Rabbit' (Dolbear et al., 2007), the Ordnance Survey's controlled English language. The intention of Rabbit is to allow domain experts to write up their knowledge in a controlled manner such that it can be translated into OWL-DL whilst still being readable by a person without particular technical understanding. On the basis of this, an OWL-DL based ontology is constructed that translates the conceptual ontology into a logical format readable by computers. The ontology examined here was the OWL-DL hydrology ontology[5], which considers features relating to inland water features found in the UK (and thus part of the Ordnance Survey dataset).

The ontology contains spatial relations as properties, including an encoding of the Region Connection Calculus (RCC) (Randell et al., 1992) under the property *spatially related*. Other spatial relations included elsewhere as properties include location, with

[3]Web Ontology Language (OWL) Reference: http://www.w3.org/TR/owl-ref/ (Visited, July 2006)

[4]Ordnance Survey Ontologies: http://www.ordnancesurvey.co.uk/oswebsite/ontology/ (Visited, July 2006)

[5]Ordnance Survey Hydrology ontology: http://www.ordnancesurvey.co.uk/oswebsite/ontology/Hydrology0.1.owl (Visited, July 2006)

relations for being located near, inside, behind etc. These properties are not grounded in actual definitions of these spatial relations, but represent the possible relations that can hold between features.

For example, as discussed in 2.5.1, spatial relations are defined logically in terms of the primitive relation of connection. From this, relations such as ‘part of’, ‘overlap’ and ‘proper part’ are defined. However, in the Ordnance Survey ontology of properties, this is not made explicit; because ‘spatially inside’ is a sub-property of ‘spatially connected’; both properties must hold if ‘spatially inside’ holds, but there is no indication logically of why this should be. This does still allow some reasoning of the relations that hold between features within the ontology. Thus, if a geographical feature has been identified within the ontology, then some of these properties will hold in relation to other features depending on the definitions within the ontology.

Within the OWL-DL ontology, many of the properties are not actually implemented in the logic, and are instead attached as comments to show the full definition for a particular concept. Thus whilst a concept may refer to such properties as ‘enables’ or ‘part of’ in its RDF comment, it may not necessarily have this property attached logically. This may be due to the limitations of description logic.

The first concept of interest within the ontology is *Hydrology Concepts*[6], from which other hydrographic concepts are derived. The concept *Body of Water* is a sub-concept of this, which is defined as being made of water and having a footprint, thus linking such objects to their geometry. Thus, spatial relations would act upon the footprint of a feature, allowing further inferences to be made.

Further to this, lake is a direct sub-concept, defined as being contained (typically) within some basin, as well as connected to other hydrographic features such as rivers, streams or other lakes. Rivers are also considered as direct sub-concepts, though there is nothing explicit in the logic that separates them from lakes. The comments define a river as flowing within a channel, containing at least one stretch, and having sources and mouths/confluences, which flows into another hydrographic feature such as the sea, a lake or another river.

By focussing on a smaller domain, the Ordnance Survey ontology contains great detail regarding hydrographic features such as rivers and lakes. This includes additional information such as potential usage e.g. a lake allows fishing to occur, and also the notions of connections between features. The ontology also mentions spatial relations, though these are not directly used within the definitions, for example using specific forms of spatial relation to connect parts together.

[6]This seems to be a category mistake, as this is a type of concept as opposed to a geographic feature.

The Ordnance Survey continues to develop and refine their ontologies, and an updated version of the hydrology ontology has been developed[7]. The *Hydrology Concepts* category has been replaced with a *Topographic Object* category, which seems more appropriate given the nature of the domain. *Body of Water* has also been removed, with 'river' and 'lake' now direct sub-categories of topographic objects. Instead, both are given the property of containing water.

Properties such as spatial relations have been re-organised into more refined categories as opposed to the more general RCC-based relations. Spatial relations are now defined in terms of location, such that an object can be located behind or inside another object for example. A new property of 'mereological relation' has also been added, which represents whether an object is part of another object. These new properties are incorporated logically into the definitions of objects, as opposed to merely being referenced in the description.

Thus although some issues are addressed as the Ordnance Survey refine their ontology (such as incorporating logic into the definitions), issues such as the handling of spatial relations remain.

2.3.2 SWEET

The Semantic Web for Earth and Environmental Technology (SWEET) is an upper-level ontology intended to represent information for Earth Science (Raskin and Pan, 2005). It takes a hierarchical approach, which OWL is able to represent effectively. However, it is also noted that OWL fails to represent numbers effectively at present, which is problematic for scientific purposes whereby numerical scales or variations of 'greater than' such as 'brighter' cannot be represented correctly.

SWEET consists of several different ontologies, covering both orthogonal concepts such as space and time, and scientific knowledge about phenomena and events. The *Earth Realm* ontology is the ontology interested in describing amongst other things geographical features, including inland water networks. The ontology contains some notion of spatial relations, including relations like 'overlap', 'inside' and 'surrounded by'. These are not explicitly linked to a spatial ontology however. There is a class of substances, allowing the differentiation between different forms such as land or water.

Within the *Earth Realm*, there is the class *Body of Water*, which is defined as being 'a body whose primary substance is water and is part of the hydrosphere'. This is divided into 'Ocean' and 'Land Water' classes, such that no feature can be both. A subclass of

[7]Ordnance Survey Hydrology ontology Version 2:
http://www.ordnancesurvey.co.uk/oswebsite/ontology/Hydrology/v2.0/Hydrology.owl (Visited, May 2008)

Land Water is ‘Surface Water’, which is defined as ‘a land water region that is surrounded by land’, thus inheriting the property of being a 2D region that can be mapped to a 3D layer of the Earth. Surface water is split into ‘Lake’, ‘Pond’ and ‘Water Channel’, where a Lake can be freshwater or saline, depending on the type of water that constitutes the primary substance. A ‘River’ is a sub-class of ‘Stream’, which is in turn a sub-class of Water Channel.

Because SWEET is an upper-level ontology, there are few relations used to ‘flesh out’ these definitions of rivers and lakes. Primarily, they are defined as being regions that exist on the Earth and contain water, but at the surface water level there is nothing that defines what makes something to be labelled a lake rather than a water channel for instance. The intention would instead seem to be that a lower level ontology would for instance make such distinctions, and then feed into the upper level for other reasoning purposes. Thus SWEET is able to cover a large range of features in general, rather than concentrating on a particular sub-domain in detail. This includes the spatial relations, which are included as relations but not explicitly defined to indicate their meaning.

2.3.3 GeoCrossWalk

GeoCrossWalk[8] is an attempt to provide a link between geographical features and the underlying data. As noted on its website, a simple overview of the aim of GeoCrossWalk is:

> “GeoCrossWalk is a database of geographical feature’s [sic] (like towns, rivers, woodlands and counties), their name and location. In other words, a gazetteer. GeoCrossWalk does not just store a feature’s location as a point, it stores the feature’s ‘footprint’.”

A gazetteer is similar to an ontology in its usage; although there is less emphasis on logical relations, the structure typically allows features to be broader or narrower versions of other features. Of particular interest with the project, is the link to a ‘footprint’ of geometric data; a lake may be represented by a polygon or a river by a line. This allows spatial reason to be implemented upon the data, thus asking what features are near each other or overlapping.

The project uses a feature type thesaurus to categorise the geographic places and features encountered. This is adapted from the Alexandria Digital Library Feature Type Thesaurus (ADLFTT)[9]. The ADLFTT is intended to allow mapping of features to any

[8]GeoCrossWalk: http://www.geoxwalk.ac.uk/ (Visited, November 2007)

[9]The Alexandria Digital Library Feature Thesaurus (ADLFTT) http://www.alexandria.ucsb.edu/gazetteer/FeatureTypes/ver070302/index.html (Visited, December 2007)

part of the world, and thus required some modification in order to be applicable to the more specific domain of the United Kingdom. This included the removal of features that would not occur within the UK, such as volcanoes, but more importantly, the re-ordering of some terms to more closely map the gazetteer to the UK. For example, in the ADLFTT, a stream is defined as[10]:

> "Linear bodies of water flowing on the Earth's surface. [USGS Feature Class Definitions]"

From this, rivers are defined as:

> "Natural freshwater surface streams of considerable volume and a permanent or seasonal flow, moving in a definite channel toward a sea, lake, or another river; any large streams, or ones larger than brooks or creeks, such as the trunk stream and larger branches of a drainage system. [Glossary of Geology, 4th ed.]"

Rivers are classified as being more specific forms of streams in the ADLFTT. However, within the GeoCrossWalk feature type thesaurus, the two are reversed, such that a stream is a more specific form of river. To complete the set of definitions, lakes are defined in the ADLFTT as:

> "Natural inland bodies of standing water, generally of appreciable size, occupying a depression in the Earth's surface. [Adapted from Glossary of Geology, 4th ed.]"

The feature type thesaurus does not place as great an emphasis upon logical relations as an ontology may, and the features are instead ordered in a simple structure, whereby one feature may be a narrower or broader version of another. Within the thesaurus, there is a category called *Hydrographic Features*, which contains two levels: the top level contains the features bays, channels, lakes and rivers; the second level contains streams, which are considered a derivative of rivers.

In comparison with other ontologies examined here, the feature type thesaurus of GeoCrossWalk contains very little detail in terms of logical definitions. Instead, the expressivity comes from the ontology being grounded upon geometric data, in the sense that a lake is not simply stored as a point but rather as a polygon representing the footprint of the feature. Similarly, cities are stored as a polygon encompassing the extent of the city,

[10]The definitions were taken directly from the ADLFTT website, and the tag in square brackets representing the original source of this definition listed on that website

instead of simply marking the city centre for instance. Thus GeoCrossWalk can handle spatial queries upon the data, for example determining if a particular river flows through a city or not.

A limitation of this approach is the definition of these polygons, as they are dependent upon a particular design or viewpoint. Since the definitions of features are not fully ‘fleshed out’, it may not be clear what choices were used in demarcating a particular feature, as well as being unable to modify this if the demarcation does not match the user’s requirements.

Finally, the decision to swap the ordering of river and stream is an interesting one, and shows the problems faced when attempting to construct ontologies of such domains. In the ADLFTT, rivers are a refinement of streams, such that any flowing channels of water are streams, and a refinement of this is the labelling of large streams as rivers. However, for GeoCrossWalk the two are swapped, changing the definition to rivers being the major feature, with streams becoming smaller channels of water. Thus the ordering of the two in this manner has a subtle impact upon the overall definition and meaning of these features. Depending upon the context, both could be considered correct, as the term ‘stream’ is used differently by different groups or countries for instance.

2.3.4 OpenCyc

Although it is not a specifically geographic ontology, OpenCyc[11] does contain definitions for the geographical domain by virtue of its aim of being a general knowledge base. This also gives a different perspective on the problem, since OpenCyc seeks to model knowledge from a variety of sources as opposed to a single viewpoint.

OpenCyc is not written in OWL-DL or similar, but instead is written in CycL[12] to fully handle the input of knowledge into the system. CycL is based upon first order predicate calculus, but extends this to allow even more flexibility and expressivity within the language, for example allowing higher order quantification. This contains mappings to OWL, thus interaction between the languages should be possible, allowing OpenCyc to be used as an upper level ontology.

An important aspect of OpenCyc is the use of *microtheories*, whereby each *microtheory* represents a particular context, by modelling a set of assertions about a particular aspect. Because they are context dependent, *microtheories* allow assertions to conflict with each other, thus definitions of objects can vary between *microtheories* and even be

[11]OpenCyc is the open source version of Cyc, a general knowledge base and commonsense reasoning engine: http://www.opencyc.org/ (Visited, April 2007)

[12]The syntax of CycL: http://www.cyc.com/cycdoc/ref/cycl-syntax.html (Visited, April 2007)

inconsistent with each other. For the analysis of the representation of “river” and “lake”, the *Universal Vocabulary Microtheory* is considered, as this provides descriptions of the intended interpretation of the assertions in the theory.

Although not explicitly contained in the definition of features, OpenCyc includes *microtheories* for spatial relations. Thus spatial relations can hold as predicates between terms, though these are defined at an upper level and hence do not describe how these relations are calculated. It should therefore be possible to create a *microtheory* for geographical features that uses spatial relations to define the features, as opposed to the existing definitions that are largely devoid of them.

OpenCyc is largely structured into sets, with assertions specialising further upon existing assertions. The most general level of interest is *Geographical Region*, from which the specialisation *Topographical Feature* is obtained. A topographical feature is defined as:

> “A three-dimensional feature of a planet’s surface, typically with boundaries defined by formations of rock, dirt, water, etc., or by significant changes in elevation.”

Thus OpenCyc is working with the physical notion of topographical features, as opposed to the projection to a map (therefore the three-dimensional aspect is crucial to the definition as opposed to assumed). This specialised further to obtain *Body of Water*, which is defined as:

> “A natural or artificial topographical feature consisting of a relatively large volume of water primarily in liquid form, contained by an instance of Basin-Topographical. Both the basin and the water are considered parts of the *BodyOfWater*”

OpenCyc acknowledges that it is not only the water that forms a water feature but also the land (or basin). It is from this feature that lakes and rivers are derived. A lake is a direct specialisation of Body of Water that is disjoint from Stream, Canal, Ocean and Harbour:

> “Each instance of Lake is a land-locked body of water, typically but not necessarily of fresh water. Two important specializations are *FreshWaterLake* (instances of which are fresh-water lakes) and *InlandSea* (instances of which are salt-water lakes).”

A stream is specialised also from Body of Water, but additionally with *Watercourse*, which is a channel of flowing water (either artificial or natural). Thus River is a specialisation of Stream distinguished by size:

> “A specialization of Stream. Each instance of River is a natural stream of water, normally of a large volume”

The distinctions between river and lake are more ‘fleshed out’ in OpenCyc than in some other upper-level ontologies, making clear the distinction between the two via the need for a flowing channel. OpenCyc also takes into consideration the fact that the land formation which forms a particular feature is also of importance, not just the water contained within. The allowance for *microtheories* means that further refinement of these ideas could be possible with OpenCyc, depending on the given context.

2.3.5 Conclusion

As previously noted by Agarwal (2005), a comprehensive geographical ontology does not exist at present. With each of the ontologies considered, deficiencies have been found, particularly in relating to actual data. Firstly, whilst some of the ontologies contained some spatial relations, they did not use these within the definitions of features. As spatial relations are crucial to the geographical domain, it would be preferable for an ontology of the domain to allow these to be incorporated.

One reason for the lack of spatial relations may be in part due to the choice of OWL-DL as the language used for the ontology, as restrictions in DL make implementing spatial reasoning difficult within this language. The restrictions in place upon DL also cause problems elsewhere, for example, inhibiting the definitions of features. However, a more descriptive language, such as that used for OpenCyc, can make verification difficult, as well as create consistency problems when attempting reasoning.

The ordering of river and stream highlights the subtle differences in viewpoints different people may have, and that terms may be used differently in different countries and communities, and thus highlighting the importance of context in definitions. Whilst a stream is considered a sub-concept of river within a UK context, other contexts reverse this ordering.

As some of the ontologies considered here are upper level ontologies, they are extremely detached from the data level, hence definitions are more general than may be desired. For example, in SWEET there is no logic that classifies the difference between lakes and rivers; this distinction would therefore need to occur at a lower level. Similarly, whilst GeoCrossWalk contains segmented data (and thus a river has a ‘footprint’ within the data), no logical definitions are given that explains why something is marked as a particular feature. Ideally, the definitions within the ontology should be logic based, such that it is clear what the distinctions between features are. Further, it would also be desirable if primitives defined within the ontology could also be related directly to geographic data.

Finally, none of the ontologies consider vagueness in any great detail. For example, whilst all seem to acknowledge that a river is a large version of a stream (irrespective of the ordering within the ontology itself), none offer a way of clarifying what it meant to be a large stream, and thus the boundary between the two features.

In conclusion, the key points that need to be addressed by geographical ontologies or systems implementing them (such as GeoCrossWalk) are:

- Spatial Reasoning: A geographical ontology should be capable of allowing reasoning about spatial relations that exist between features.
- Logic based definitions: The definitions of features should be fleshed out logically, so that it is clear what a particular definition means.
- Vagueness: Some method of handling vagueness should be included, due to the importance of this to the domain.
- Relation to Data: The primitives defined in an ontology should be linked directly to geographic data.

2.4 Ontology Grounding

The ontology level is usually seen as separate to the data level; reasoning on queries is performed within the ontology, and data is returned that matches these queries. Thus the ontology is devoid of the data context. This has a clear impact upon handling vagueness, where context is important. A proposed improvement to this is to ground the ontology upon the data (Jakulin and Mladenić, 2005).

By grounding the ontology, an explicit link between the ontology and the data is made, thus allowing reasoning to be made within the context of the particular data. GeoCrossWalk, discussed in Section 2.3.3, is an example of this within the geographic domain; the project aims to connect the data and the definitions by relating the meanings of features to a specific ‘footprint’ in the data, as opposed to simply a point.

The symbol grounding problem as proposed by Harnad (1990) suggests that computers do not actually understand knowledge they are provided with. There have been no adequate solutions to this problem as yet and it remains an open problem (Taddeo and Floridi, 2005), although some people question that symbol grounding is really a genuine problem. Ontology grounding does not solve the problem, although it could be argued that associating spatial extensions with named entities could be described as grounding.

Table 2.2: An example of the possible structure of layers allowing the same general layer to be connected to different forms of data layers. This is adapted from (Third et al., 2007)

General: spatial and temporal logic global structure high-level predicates river$[s,l](x)$, stream$[s,l](x)$		
Grounding 1: river$[s,l](x)$ $\leftrightarrow$ linear$[l](x) \wedge$ water(x) $\wedge\neg$small$[s](x)$ stream$[s,l](x)$ $\leftrightarrow$ linear$[l](x) \wedge$ water(x) $\wedge$small$[s](x)$		**Grounding 2:** river$[s,l](x)$ $\leftrightarrow$ 2D-linear$[l](x) \wedge$ $\exists y[$3D-bed(x,y) $\wedge$3D-channel$(y)]$
Data 1: 2-D topographic data Human-scale small$[s](x)$	**Data 2:** 2-D topographic data Boat-scale small$[s](x)$	**Data 3:** 3-D topographic/ bathymetric data 2D-linear$[l](x)$ 3D-bed(x,y) 3D-channel(y)
linear$[l](x)$		

In grounding definitions upon data, the symbols attached do have meaning with respect to the spatial extension they are related to.

Grounding the ontology upon the data allows reasoning with the data in particular contexts. Since context can be important to vagueness, grounding the ontology upon the data gives us a method of reasoning with that vagueness, as proposed by (Bennett, 2006), where it was noted that uses of words such as ‘large’ are dependent on the context they are used in. Thus if referring to whales rather than humans, grounding the definition on the data allows ‘large humans’ and ‘large whales’ to be referred to seperately.

To ground the ontology upon the data, work is required at both the data level and the ontology level. Previously, linearity was described as an example of an attribute which could be used to ground an ontology upon data, and it was shown that work was required at both levels to use such an attribute (Mallenby, 2007). An architecture for such a layering effect was discussed by Third et al. (2007), whereby the need for a grounding layer between the data and general ontology levels was discussed. An example of this layered structure is shown in Table 2.2.

2.4.1 Solving Queries through Model Construction

In order to ground the ontology upon the data, a method of handling logical queries that may be asked is required. One suitable approach given the data to be handled is *model building*, which has been applied in other areas using logic such as computational semantics (Bos, 2003; Blackburn and Bos, 2005). With model building, a model of the domain is constructed using theories that are input, to determine if they are consistent (or satisfiable). A *model checker* can then be used, whereby given a model and a first order query, the model checker determines whether the query is satisfied or not by the model. If there are no free variables then it is a simple test of consistency, whereas if there are free variables, the model checker can assign values from the model domain that will satisfy the query. An example of this was presented by Blackburn and Bos (2005).

The approach is suited to small, finite domains, but the approach does not scale well, as discussed by Kohlhase and Koller (2003). Whenever anything is added to the domain or new theories are added, the model will increase in size, meaning queries take longer to process. This becomes especially clear when dealing with existential or universal quantification, as these may require testing over the entire domain. Therefore the completeness of model building is also a problem computationally. Attempts to improve the efficiency of the approach have for example added salience values to the data to ensure that resources are handled efficiently (Kohlhase and Koller, 2003).

As noted by Claessen and Sörensson (2003), model builders tend to fall into two categories: MACE-*style* methods such as MACE (McCune, 2003) and PARADOX (Claessen and Sörensson, 2003), and SEM-*style* methods such as SEM (Zhang and Zhang, 1996) and FINDER[13]. These two styles approach the problem of model building differently, and thus are better suited to different types of problems: SEM-style methods perform well on equational problems whilst MACE-style methods allow a greater flexibility in the clauses and hence are more general purpose. MACE-style methods use the Davis-Putnam-Loveland-Logemann (DPLL) (Davis et al., 1962) algorithm to generate the models. A series of first order logical clauses are first taken and translated into a propositional logic clause set, by flattening and instantiating the clauses. These can then be proven using the DPLL-based algorithm. SEM-style methods on the other hand, attempt to formulate the problem as a constraint satisfaction problem, rather than converting the logic. Backtracking through interpretations is then used to determine a solution. This limits the sort of problems that can be handled to some extent, but also means that the process is optimised for particular problems.

[13]FINDER 3.0 website: http://users.rsise.anu.edu.au/ jks/finder.html (Visited, November 2007)

2.5 Qualitative Spatial Reasoning

A comprehensive overview of qualitative spatial reasoning is provided by Cohn and Hazarika (2001), where it is proposed that the aim of qualitative spatial reasoning is to:

> "provide calculi which allow a machine to represent and reason with spatial entities without resort to the traditional quantitative techniques prevalent in, for e.g. the computer graphics or computer vision communities." (Cohn and Hazarika, 2001, p.4).

Qualitative spatial reasoning aims to allow reasoning about the relations that hold between spatial regions, without requiring that quantitative features are explicitly measured such as length, distance and area, as well as being able to infer further relations that may hold. For example, if you were on a train and had a bag that had books in, you could infer that the books were also on the train due to the spatial relations between the bag, the books and the train. Geographically, such relations are common when describing the world around us, since we often talk of features being inside, part of or next to other features.

Of particular interest for this work is the reasoning of the topology of features, determining when regions are for example overlapping, disconnected or inside each other. The principal approaches considered here are the Region Connection Calculus (RCC) (Randell et al., 1992), and Egenhofer and Franzosa's 9-Intersection Calculus (Egenhofer, 1991; Egenhofer and Franzosa, 1991).

2.5.1 The Region Connection Calculus

The Region Connection Calculus (RCC) was introduced by Randell et al. (1992). RCC assumes an initial primitive relation $\mathsf{C}(x, y)$, which holds when the topological closures of regions x and y share a common point (Randell et al., 1992) and are thus considered to be "connected". From this initial connected relation, other relations that hold between two regions can be derived. A list of the basic key relations as listed by Randell et al. (1992) follows:

$$\mathsf{DC}(x,y) \equiv_{df} \neg\mathsf{C}(x,y) \tag{2.1}$$

$$\mathsf{P}(x,y) \equiv_{df} \forall z[\mathsf{C}(z,x) \rightarrow \mathsf{C}(z,y)] \tag{2.2}$$

$$\mathsf{PP}(x,y) \equiv_{df} [\mathsf{P}(x,y) \wedge \neg\mathsf{P}(y,x)] \tag{2.3}$$

$$\mathsf{EQ}(x,y) \equiv_{df} [\mathsf{P}(x,y) \wedge \mathsf{P}(y,x)] \tag{2.4}$$

$$\mathsf{O}(x,y) \equiv_{df} \exists z[\mathsf{P}(z,x) \wedge \mathsf{P}(z,y)] \tag{2.5}$$

$$\mathsf{DR}(x,y) \equiv_{df} \neg\mathsf{O}(x,y) \tag{2.6}$$

$$\mathsf{PO}(x,y) \equiv_{df} [\mathsf{O}(x,y) \wedge \neg\mathsf{P}(x,y) \wedge \neg\mathsf{P}(y,x)] \tag{2.7}$$

$$\mathsf{EC}(x,y) \equiv_{df} [\mathsf{C}(x,y) \wedge \neg\mathsf{O}(x,y)] \tag{2.8}$$

$$\mathsf{TPP}(x,y) \equiv_{df} \mathsf{PP}(x,y) \wedge \exists z[\mathsf{EC}(x,z) \wedge \mathsf{EC}(y,z)] \tag{2.9}$$

$$\mathsf{NTPP}(x,y) \equiv_{df} \mathsf{PP}(x,y) \wedge \neg\exists z[\mathsf{EC}(x,z) \wedge \mathsf{EC}(y,z)] \tag{2.10}$$

RCC-8 consists of eight of these relations: DC, EQ, PO, EC, TPP, TPPi, NTPP, NTPPi, where TPPi and NTPPi are the inverses of TPP and NTPP respectively. Figure 2.1 shows graphically the RCC-8 set. This set is both jointly exhaustive and a pairwise disjoint set of base relations, such that only one can ever hold between two given regions (Randell et al., 1992). Depending upon the requirements, this set can be restricted or expanded. For example, RCC-5 uses only DR, PO, PP, PPi, EQ, where PPi is the inverse of PP (Bennett, 1994). Thus, RCC allows different levels of refinement; as discussed in (Li and Nebel, 2007) where a generalised version of RCC (GRCC) is proposed. When dealing with rough sets it may not be possible to distinguish whether two regions are DC or EC; hence, only the RCC-5 relations could be used. In the same instance though, this would be sufficient for the given usage.

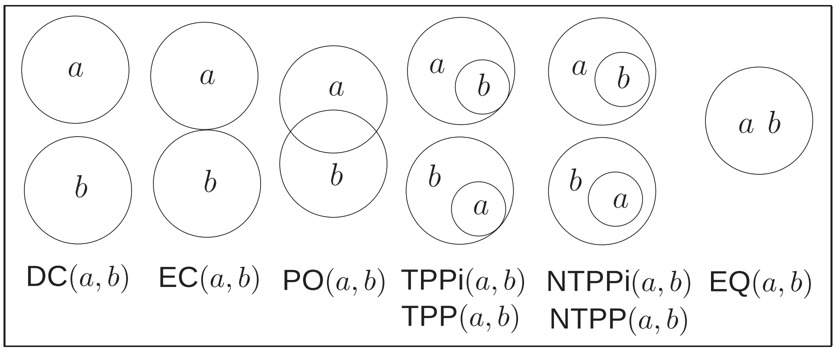
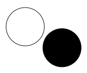
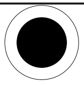
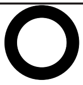
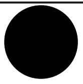
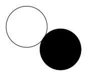
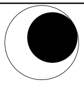
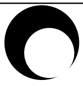
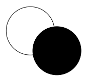


Figure 2.1: The RCC-8 relations, demonstrating the set of base relations that can hold between two regions

Figure 2.2: The eight topological relations between two spatial regions together with the corresponding 9-Intersection matrix (Egenhofer, 1991). For each, the label given to that relation is shown, together with the corresponding RCC relation in brackets.

 $\begin{pmatrix}001\\001\\111\end{pmatrix}$ disjoint (DC)	 $\begin{pmatrix}111\\001\\001\end{pmatrix}$ contains (NTPPi)	 $\begin{pmatrix}100\\100\\111\end{pmatrix}$ inside (NTPP)	 $\begin{pmatrix}100\\010\\001\end{pmatrix}$ equal (EQ)
 $\begin{pmatrix}001\\011\\111\end{pmatrix}$ meet (EC)	 $\begin{pmatrix}111\\011\\001\end{pmatrix}$ covers (TPPi)	 $\begin{pmatrix}100\\110\\111\end{pmatrix}$ coveredBy (TPP)	 $\begin{pmatrix}111\\111\\111\end{pmatrix}$ overlap (PO)

2.5.2 Egenhofer and Franzosa's 9-Intersection Calculus

The 9-Intersection Calculus (Egenhofer and Franzosa, 1991) has many similarities to RCC-8, and though developed independently also contains spatial relations significant to GIS, as noted by Bennett et al. (1998). However, whereas RCC has a logical basis, the 9-Intersection Calculus is derived from a more mathematical basis.

As detailed in Clementini et al. (1994), the binary topological relation between two objects A and B is based upon the intersection of the two objects' interiors (A° and B°), boundaries (∂A and ∂B) and exteriors (A^- and B^-). These are typically modelled in a 3×3 matrix as shown in Equation 2.11, whereby each is either empty or non-empty (0 or 1 respectively).

$$\mathfrak{I}_9(A,B) = \begin{pmatrix} A^\circ \cap B^\circ & A^\circ \cap \partial B & A^\circ \cap B^- \\ \partial A \cap B^\circ & \partial A \cap \partial B & \partial A \cap B^- \\ A^- \cap B^\circ & A^- \cap \partial B & A^- \cap B^- \end{pmatrix} \tag{2.11}$$

Thus, in theory, there are $2^9 = 512$ binary relations, but for connected homogeneously 2-D areas with connected boundaries there are 8 mutually exclusive relations providing complete coverage of the possible relations (Egenhofer and Al-Taha, 1992; Egenhofer and Franzosa, 1991). These are shown in Figure 2.2, and are identical to the RCC-8 relations noted previously.

The composition and transition between the relations have been considered, with a graph-like structure developed to represent this, as shown in Figure 2.3. In particular, Egenhofer and Al-Taha (1992) looked at the gradual transition from one relation to another, and proposed the graph be modified to include the dotted lines in Figure 2.3.

To determine possible relations between relations, a composition table could be derived. For example, if there are three regions and the spatial relation that holds between two pairs of the regions is already known, the possible relations that hold between the final pair can be derived via the composition table. These operations can be performed using set and matrix operations upon the intersections between the objects.

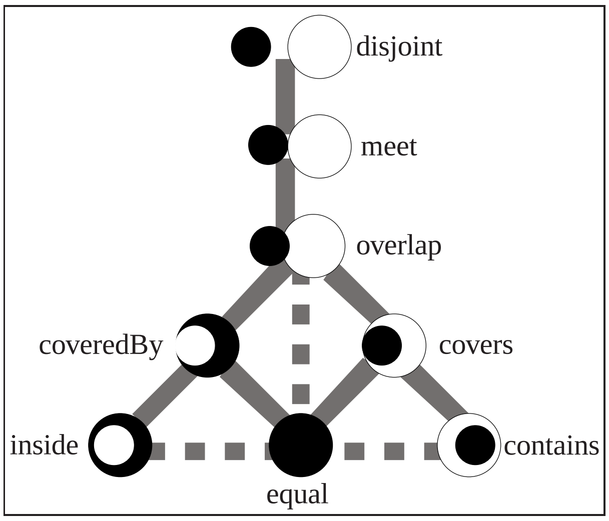


Figure 2.3: The 9-intersection calculus, demonstrating the spatial relations that hold between regions, arranged by conceptual neighbourhoods Egenhofer and Al-Taha (1992).

2.5.3 Allen's Interval Algebra

Allen's Interval Algebra (Allen, 1983) represents the relations that hold between two directed one-dimensional time intervals, and thus was initially proposed for temporal relations and reasoning. The Algebra consists of 13 relations, as shown in figure 2.4. The Algebra has however formed the basis of two-dimensional reasoning with the Rectangle Algebra (Güsgen, 1989; Balbiani et al., 1998, 1999). The Rectangle Algebra consists of applying Allen's Interval Algebra to both axes separately and comparing the results, thus there are $13 \times 13 = 169$ relations within the Rectangle Algebra. The Algebra is restrictive in the sense it is based upon rectangles, though as noted by Balbiani et al. (1999) it can express directional relations such as whether a region is to the left or right of another region in addition to the topological relations. Also, it is computationally simpler than other spatial reasoning approaches due to the restriction of using rectangular regions.

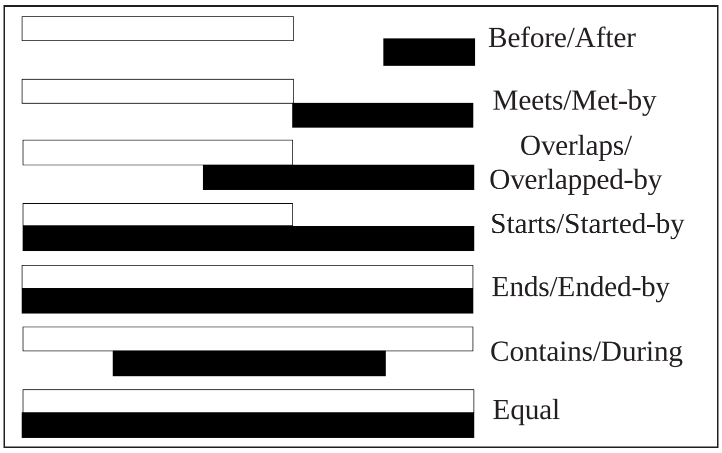


Figure 2.4: A graphical representation of the 13 different Allen relations. With the exception of the final relation Equals, the other 12 are in fact 6 pairs of duals. Thus, the first relation represents both white before black and black after white.

2.5.4 Applying Qualitative Spatial Reasoning to Data

The problem of combining qualitative and quantitative data was discussed by Abdelmoty et al. (1993). Here, the combination of different levels of information are discussed, such that the intention is to bridge the gap between the primitive level of points, lines and polygons, and the object level describing the spatial relations and definitions of features. Transitivity tables are formed representing the possible relations between different primitives. Thus, spatial relations can be calculated by deductive processes as opposed to computational geometric algorithms (or at least a reduced usage of such algorithms). Similarly, Rodríguez et al. (2003) combined the 9-Intersection Calculus with composition-based and neighbourhood-based approaches to reduce the number of topological constraints that needed to be satisfied, thus reducing the overall computation.

Pratt and Schoop (1998) discuss the problem of Euclidean space containing many regions which are of no interest or use, such as shapes with convoluted boundaries. They therefore propose a different calculus intended to deal with physical regions, represented as polygons, that would be of interest. However, this results in a more complex theorem due to the restrictions in place.

A hierarchical approach to determining RCC relations is discussed by Bennett et al. (1998). Moreover, the calculations were converted to boolean terms, such that the problem becomes one of the closure of half-planes.

Another approach to deducing the spatial relationships is to use constraint logic programming (Jaffar and Maher, 1994), as discussed particularly by Almendros-Jimenez (2005). Such an approach offers an interesting alternative, but is reliant on the efficiency of the constraint logic solver used, and as discussed by Almendros-Jimenez (2005), further work is required to improve such an approach for effective implementation.

2.6 Summary

This chapter has summarised the background required for this thesis. The problem of vagueness within the geographical domain was considered in general, identifying issues that can arise due to this problem. A selection of existing geographical ontologies were reviewed, and deficiencies were determined within these. The meaning of ontology grounding was discussed, which was shown to require consideration at multiple levels within the system in order to be implemented. Finally, qualitative spatial reasoning was considered, both the principal theories used and their application to actual data.

Chapter 3

Approaches to Vague Reasoning

3.1 Introduction

This chapter will compare the two principal approaches to vague reasoning, in order to determine the most effective approach for my case study. Section 3.2 will discuss the issue of vagueness, particularly in relation to the geographical domain. The two principal approaches to handling vagueness will then be considered; Section 3.3 will look at Fuzzy Logic and Section 3.4 will look at Supervaluation Theory. Section 3.5 will then compare the two approaches' suitability to handling vague aspects of the inland water domain, and discuss the approach I decided to use for my problem in Section 3.6. Finally the chapter is summarised in Section 3.7.

3.2 Vagueness

Vagueness is prevalent within our language and the manner in which we describe the world around us. Vagueness has been considered and discussed by philosophers for many years, with examples stretching back to ancient Greece; the *Sorites* paradox, which is often used to outline a problem of vagueness for reasoning, originating in a series of puzzles attributed to the Megarian logician Eubulides of Miletus. The *Sorites* paradox can be easily adapted to illustrate vagueness in geography, as shown by Varzi (2001b). The example given by Varzi (amongst others) is that of mountains, Everest in particular.

Recalling the previously given dictionary definition of a river, this can be rewritten in the form of a *Sorites* paradox. The 'large' part of the definition will be used, and for convenience large will be equated to mean the length of the river. In actuality, the term 'large' is itself vague, since there may be more than one factor determining if a river is large. The paradox starts with something that is clearly a river, the Nile, which has a length of approximately 6,690 kilometres:

> A 6,690,000 metre long channel of water is a river.
> For all k: if a (k+1) metre long channel is a river, so is a k metre long channel.
> *Ergo*, a 1 metre long channel is a river.

Clearly, the same argument would hold if the initial clause was a 1 metre long channel not being a river and the length was increased by 1 metre each time. The paradox would remain; whilst there are some things that are definitely rivers and some that are definitely not, there exist borderline cases that may or may not be considered rivers depending on people's perspectives.

Further to this, the borderline cases make it difficult to individuate features. Geographical features are often part of a larger feature (Smith and Mark, 1998; Fonseca et al., 2002). Whilst people have an understanding of what a river is as opposed to a lake, they may not be able to agree upon a specific border at which they would have left the river and entered the lake, if they were travelling along the water in a boat, for example. An example of this is shown in Figure 3.1, whereby a single feature is shown that may represent a river, but alternatively could be a series of lakes connected together. If it was the latter, some form of individuating would be needed to demarcate lakes from the original feature, but it is not clear where such boundaries would be placed.

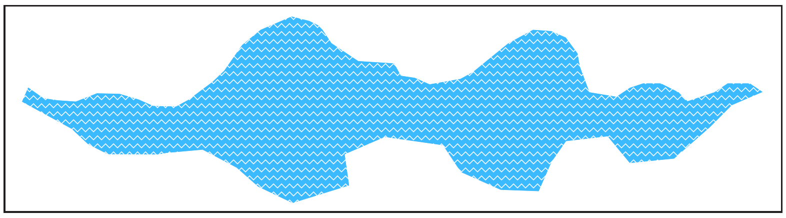


Figure 3.1: An example of the problem of individuation. It isn't necessarily clear whether this is a single large river, or a series of connected lakes.

The *Sorites* paradox is not the only form of vagueness, and Dubois et al. (2001) illustrate other forms of vagueness that may arise, whilst Bennett (2005) has also identified different varieties of vagueness and ambiguity. However, the Sorites paradox perhaps best highlights the problem of vagueness; if no precise boundary between two classifications exist, how can such classifications be applied? Clearly, this causes problems for reasoning about vague objects, as classical logic is insufficient.

This may lead some to view vagueness as a defect or deficiency of language that should be removed or avoided, but as argued by Bennett (2001b) and Dubois et al. (2001), vagueness is, in fact, an important part of our language. Thus someone can be described as ‘tall’ without knowing their exact height, or the River Tyne as a river despite not having a concrete definition of a river (or specific boundaries encompassing the Tyne). A far more suitable approach then, is to find a method for handling and reasoning about vagueness.

One of the principal considerations for ontologies of geographical information is the handling of vagueness, as noted by Tomai and Kavouras (2004). In order to be able to ground an ontology upon the data, it is important to evaluate the main approaches to handling vagueness, to allow their applicability to be determined with regard to the geographical domain and to the problem of ontology grounding in general.

A discussion of the different categories of vagueness is provided by Dubois et al. (2001), illustrating that vagueness can be subtly different depending on the circumstances in which it appears. Therefore, there may exist some circumstances where one particular form of reasoning is more suitable, whilst in other situations another approach is more applicable. Considering the geographical domain independently of others allows evaluation of the main approaches to handling vagueness and determination of which is most suited to the case study considered in this thesis.

As noted by Williamson (1999), vagueness can also be of a higher order than just first-order vagueness, in the sense that any statement made about a vague statement may itself also be vague. Kulik (2003) addresses the problem of second order vagueness with regards to spatial reasoning, and notes that vague geographic features such as ‘forests’ may have different ways of being characterised, thus second order vagueness could be seen as an overlay of these different characterisations.

The principal approaches for handling vagueness at present are Fuzzy Logic and Supervaluation Theory. Both approaches offer a method of reasoning over vague features. It is usually the case that the two are presented as opposing theories. However, this in part assumes that vagueness can only take one form, which as discussed by Bennett (2005) and Dubois et al. (2001) is not true. Rather, vagueness can take differing forms and hence may require different methods depending upon the context or problem. Recently, it has even been proposed that the two theories are not as incompatible as previously thought (Fermuller and Kosik, 2006), although it may still be preferable to use one approach for a given system and not mix their usage.

Instead of dismissing a particular approach, it seems more appropriate to consider which approach is most suited to the given situation. The data that is to be input needs to be considered as well as the intended output, and from this, it may be determined which approach is most suitable; in some instances this will be Fuzzy Logic and in others, Supervaluation Theory.

3.2.1 Type of Vagueness Addressed

Before evaluating the different approaches to vague reasoning, it is important to clarify the type of vagueness that is to be addressed by the system. Some of the different types of vagueness that may relate to spatial reasoning were discussed in Section 2.2, such as boundary, classification, location and *sorites*. The intention of the system is to demarcate regions within the dataset which correspond to geographic features. Whilst this may seem to include all of the previously mentioned types of vagueness, the final classification of the regions as particular features(such as rivers or lakes) would be performed using an ontology, thus classification vagueness is not address by the system at this stage. However, the other types of vagueness previously discussed would be addressed by the system.

With features such as a river, the location of the feature is generally agreed upon, but the boundary of the feature is not, hence boundary vagueness would need to be addressed. The way such features are identified from the dataset is to identify smaller regions corresponding to some attribute within the dataset, then combining these logically to form the feature of interest. With these parts of the feature, location vagueness needs to be addressed, as the location of these features is not known. Finally, *sorites* vagueness may need to be addressed, as the thresholds used to classify the parts of the feature may themselves be vague.

Thus, the type of vagueness to be addressed in this thesis is to identify the regions that correspond to a particular feature, as opposed to classifying a region as a particular feature. The identification of this region may itself be dependent upon other types of vagueness, such as determining parts of the feature (location vagueness) and combining these to determine the boundary of the feature (boundary vagueness).

3.3 Fuzzy Logic in Geography

Fuzzy Logic is the most popular approach to handling vagueness, and has been used in a variety of applications since its conception by Zadeh (1965, 1968, 1976). The underlying concept is to allow data to be processed using partial set membership rather than strict set membership or non-membership. For example, a person is not considered to be strictly ‘tall’ or ‘short’, but rather they are considered to belong partially to some *degree* to the set of things that are tall/short. A common misconception made is that the degrees of truth represent a form of probability, and thus that Fuzzy Logic is a probabilistic approach to vagueness. This is inaccurate, as although both pertain to handling uncertainty, they do not correspond to the same thing.

Issues related to Fuzzy Logic were discussed by Williamson (1994, p. 120 - 130), in particular as to whether Fuzzy Logic addresses the Law of Excluded Middle, as claimed. Rather, it is based upon classical logic and thus is still susceptible to the Sorites paradox. The meaning of the degrees of truth may also be unclear, particularly if the predicate depends on more than one variable. For example, to determine whether a person was ‘tall’ to some degree of truth a suitable choice of measurement would be to use the person’s height. However, to determine whether a person was ‘large’ to some degree of truth may require consider multiple variables other than height (such as weight or waist size), for which it is less clear how to combine them into a single degree of truth.

Fuzzy Logic is best suited to situations where it is not felt that a sharp boundary exists between vague interpretations, but instead there exists a gradual transition between interpretations. An example of this is given by Kulik (2003); it may be desirable to state that there is a sharp boundary between ‘forest’ and ‘snow’ regions, but this would lose some of the notion of the transition between the two due to the density of the trees increasing or decreasing. A Fuzzy Logic approach would instead use the density as a degree of truth, allowing the transition between the two to be modelled effectively.

With geographic features, there may be different conceptions of how to represent them. For example, the notion of ‘objects’ and ‘fields’ in geography was discussed by Galton (2001). Objects are typically static with a definite location, and can be regarded as belonging to a *type*; for example, rivers are natural objects where each individual river belongs to the generic type *river*. Fields represent a function over the set of values characteristic of a feature. Examples could include elevation, population or the distinction between land and water. Further to the previously noted observation of gradual transitionary boundaries, Fuzzy Logic would seem to be most suited to modelling objects in terms of fields, e.g., specifying the boundary of a mountain (i.e., an object) using elevation values (i.e., a field). Thus the display of such results may take the form of a colour gradually transitioning to another colour, or more discrete separation of such boundaries e.g. grouping transitions into larger categories such as 0-10,10-20 etc.

It is not entirely clear how spatial regions can be compared using Fuzzy Logic, although the Region Connection Calculus (RCC) as first proposed by Randell et al. (1992) is adapted to fuzzy spatial relationships by Palshikar (2004) and Liu and Chen (2006).

This expansion would allow the use of Fuzzy Logic when reasoning spatially about geographical objects, which increases the applicability of Fuzzy Logic to GIS.

The use of Fuzzy Logic appears to be strongly suited to areas where numerical data will allow the construction of membership functions (Ergin et al., 2004; Porter et al., 2006). In these examples, numerical data allowed membership functions to be constructed that allowed fuzzy reasoning to be successfully used in analysing and classifying particular features. Fuzzy Logic was proposed as a possible method of handling sorites vagueness within geography by Fisher (2000).

3.4 Supervaluation Theory in Geography

Initially proposed by Fine (1975), Supervaluation Theory proposes that there exist many interpretations of the language. If a statement is true in all interpretations it is considered '*supertrue*' and, similarly, is considered '*superfalse*' if it is false in all interpretations. For all other interpretations though, the truth value will depend upon the interpretation; in some interpretations they may be true and others false. Thus Supervaluation theory has similarities with modal logics in its approach to handling statements (Dever et al., 2008).

Returning to the example of determining whether someone is tall or short, it is clear that some heights of people, such as anything over 7 feet, would always be considered tall. Thus "person *x* is tall" is supertrue if *x* was taller than 7 foot in height. Similarly, the statement would be superfalse if *x* was shorter than, for example, 3 foot in height. However, for statements such as like "A person of height 5 feet 9 inches is tall" may be true in some interpretations but false in others. In Supervaluation semantics, '*precisifications*' are used to determine the boundary points at which statements are considered true or false in a given interpretation. Supervaluation semantics by itself does not logically add anything to handle vagueness. Instead, it acts as a framework for developing an approach to handling vaginess (Bennett, 2001a,b).

Williamson (1994), questioned the validity of Supervaluation Theory as an approach to handling vagueness. In particular, the notion of *supertruth* is questioned; if statement 'S' is true if and only if S (i.e. truth is disquotational), then either 'S' is supertrue only or 'not-S' is supertrue only. This is similarly argued by Fodor and Lepore (1996), who claim that if something is indeterminate in one interpretation, it must be indeterminate in all. Thus Supervaluation theory ignores the essence of such vague sentences, and does not, in fact, handle the vagueness effectively. These criticisms are addressed to some extent by Dever et al. (2008), where Supervaluation Theory is argued to take a modal perspective upon vagueness. A modal Supervaluation Theory is also discussed by Liu and Chen (2006), where it is used to enhance description logic to allow vague reasoning.

Higher-order vagueness can be problematic for Supervaluation Theory. As noted above, with first-order Supervaluation Theory, a statement is either supertrue, superfalse or true or false depending upon the *precisification* used. However, at what point does something go from being true in some instances and false in others, to being supertrue? For example, at what height does "person *x* is tall" go from being true in some instances to supertrue? This is higher-order vagueness, and returns to the criticisms noted above. However, Varzi (2001a,b) has questioned the importance of the problem of higher-order vagueness to Supervaluation Theory, as he sees it as a problem of semantics.

Supervaluation Theory is suited to situations where, although the location of the boundary is vague, it is agreed to exist. This is different from uncertainty, where it would not be possible to mark a boundary despite having all available information. For example, most people would be able to agree upon the general position of a given river on a map, but if they were asked to mark the boundary of the river they would most likely not reach a consensus. The *existence* of the boundary would be agreed upon, but the *location* would most likely not.

If the object-based and field-based conceptions of geography as discussed by Galton (2001) are considered, Supervaluation Theory is more suited to the representation of object-based conceptions. Thus the display of such features would consist of variable sharp boundaries, such that the boundary would represent a particular *precisification* as opposed to representing a permanent or fixed boundary.

Supervaluation Semantics have previously been applied to the problems of forests (Bennett, 2001b) and inland water networks (Bennett et al., 2005). Here, the *precisifications* are generated from user preferences, to allow control over how classification occurs. Varzi (2001b) suggested that the vagueness is conceptual; there are many precise things that can be conceived to be a mountain, but Varzi argues the vagueness merely arises because it is not clear which thing is being referred to specifically, when Mount Everest is referred to, for example. This clearly fits with a Supervaluationist approach. Kulik (2000) uses Supervaluation Semantics to classify spatial regions and to answer statements such as whether an animal settles within a given boundary, as well as comparing the results with those that would have been obtained had Fuzzy Logic been used instead.

3.5 Handling Vagueness in Inland Water Networks

The use of vague reasoning approaches has thus far been considered in a general sense. Both Fuzzy Logic and Supervaluation Theory have been applied successfully to different situations that required some form of vague reasoning. Extending upon the notion of vagueness having many different forms, it is often desirable to handle vagueness differently, depending on the intended outcome.

As a general observation, Fuzzy Logic, when applied to the geographical domain, is suited to when gradual boundaries are required; instead of stating where a feature starts or ends, a range where it may start or finish would instead be given. Supervaluation Theory, on the other hand, takes a more definitive approach in a given context; it is instead stated that within a specific interpretation a feature has a definite boundary. It is thus important to consider which vague reasoning approach is most applicable to a given problem, opposed to simply dismissing a particular approach outright. The intended domain must be analysed, to determine what kind of vagueness is most relevant to that domain, and what the intended usage of the domain is.

The domain of inland water networks is intended to cover significant inland water features, such as rivers and lakes. In addition to the previous example given of size, other aspects of the domain are also vague. Geographical objects may not be a clearly demarcated entity but part of another object (Smith and Mark, 1998; Fonseca et al., 2002). The individuation of entities is therefore important to geographical domains. With inland water networks, this could apply to individuating features such as rivers and lakes from a larger, connected water network. This could mean, for example, deciding if something is a large river with a series of bulges or a series of lakes connected by smaller rivers, as shown previously in Figure 3.1.

In order to make a decision on what vague reasoning approach to use, the following points need to be considered:

- *Input Data*: The format of the data may help determine the most effective form of reasoning. For example, if the data consists of continuous valued observables, Fuzzy Logic may be more appropriate as it is possible to generate membership functions. On the other hand, geometrical data may be more suited to Supervaluation Theory, since making sharper judgements regarding particular boundaries may be required.

- *Intended Framework*: The framework in which the reasoning will be used is an important consideration. The meanings of some logical definitions will vary depending upon the approach used, which will in turn impact upon the results obtained. Careful consideration of this is therefore required.

- *Intended Output*: Finally, the intended output for the data and segmentation must be considered. This includes both the graphical output and the usage of such data.

Much like the input, the two approaches are best suited to different types of output; with Fuzzy Logic a gradient approach to boundaries is more suited whereas Supervaluation Theory can be represented with sharp but moveable boundaries.

Each of these points will now be considered in more detail, to show the different situations that each approach is suited for. This will in turn help determine which approach is most suited to the case study considered in this thesis.

3.5.1 Input Data

Geographical data can be represented in a variety of formats, with two of the principal forms being topographic data and observations of some given attribute over a particular area. Although both can be modelled using either of the vague reasoning approaches, it would seem that in each case one approach is more suitable than the other.

Referring back to the previously mentioned geographical types of objects and fields, topographic data is more suited to object-based conceptions. This is because geometric features tend to be thought of as having a particular location and boundary, even if the perceived boundary is not fixed. Thus Supervaluation Theory would appear to be more suited to topographic data, as this is more suited to situations where the presence of a boundary is perceived but which do not have a specific location fixed for it.

With continuous observables, a field-based conception is more suitable. This is because the change of a particular variable over an area is of more interest for such data, as opposed to individual, crisp values. For example, an estuary is something that could be defined geometrically from topographical data, but also could be defined based upon the relative salinity of the water (Cameron and Pritchard, 1963; Pritchard, 1967a), as discussed in Section 2.2. The definition there related the dilution of sea water and fresh water, thus the use of salinity data for water would help with such a definition.

3.5.2 Intended Framework

One of the potential benefits of Supervaluation Theory over Fuzzy Logic is its relation to classical logic. Within a given *precisification*, predicates will evaluate to true or false, and hence classical logic can be applied directly to this. Fuzzy Logic however, is not intended to use the same equivalences as classical logic. For example, the basic definitions of

Fuzzy Logic are:

$$t(\mathsf{A} \wedge \mathsf{B}) = \mathsf{min}\{t(\mathsf{A}), t(\mathsf{B})\} \tag{3.1}$$

$$t(\mathsf{A} \vee \mathsf{B}) = \mathsf{max}\{t(\mathsf{A}), t(\mathsf{B})\} \tag{3.2}$$

$$t(\neg \mathsf{A}) = 1 - t(\mathsf{A}) \tag{3.3}$$

Where A and B are assertions and $t(\mathsf{A})$ represents the degree of truth of A, such that $0 \leq t(\mathsf{A}) \leq 1$. As noted by Dubois et al. (1994), this means some boolean equivalences of classic logical do not hold, and instead represent 'partial truth'. Whilst this does not mean Fuzzy Logic cannot be used to reason with, it does raise problems.

For example, returning to salinity data, suppose a measurement in percentage terms of the ratio of sea water and fresh water at observed points was recorded. Thus any given point would have $x\%$ sea water and $100 - x\%$ fresh water. Suppose, for example, a classical definition of water was:

$$\mathsf{water}(x) \equiv \mathsf{fresh\_water}(x) \vee \mathsf{sea\_water}(x) \tag{3.4}$$

With Fuzzy Logic, this would become:

$$t(\mathsf{water}(x)) \equiv \mathsf{max}\{t(\mathsf{fresh\_water}(x)), t(\mathsf{sea\_water}(x))\} \tag{3.5}$$

Thus, the only time it is ever certain that something is water, is when it is completely sea water or freshwater. Further, any observation in between cannot state with certainty that x is water, merely give the maximum degree of truth. This is especially true when the observable in question is half of both; the resultant assertion of the degree of truth is also only 50%. Whilst this could be overcome with a rule stating x is water if $t(\mathsf{water}(x)) \geq 50\%$, this potentially loses some of the information present. Similarly, logical conjunction in Fuzzy Logic also potentially loses some of the information, due to the loss of the largest degree of truth.

Fuzzy Logic has been proposed as a framework for geographical data by Stefanakis et al. (1996), where different attributes are combined using mathematical functions to determine results. Another example is discussed by Hatzichristos and Giaoutzi (2005), where membership functions were generated for key attributes in landfill siting, then combined to determine overall suitable areas.

The intended framework therefore, will impact upon the choice; Supervaluation Theory will allow for more classical logic approaches, whereas Fuzzy Logic will be more suited to mathematical approaches.

3.5.3 Intended Output

The intended output is, to some extent, influenced by the input format used, since the results will graphically overlaid onto that input data. The key difference in the two approaches, therefore, is the representation of boundaries between features. With Fuzzy Logic some form of graded boundary would be preferable, perhaps the blending of two colours from one to the other, whereas with Supervaluation Theory crisp boundaries are more suited.

As with the input, geometrical output is more suited to Supervaluation Theory than Fuzzy Logic. For example, in Figure 3.2 there are two rivers R_1 and R_2, with R_2 flowing into R_1. Although there is not a specific boundary between the two, the existence of one could still be perceived, as opposed to there being a gradual transition from one to the other. The dotted lines, therefore, represent some of the possible boundaries that could be perceived, although many more would exist. With Fuzzy Logic, these potential boundaries would form part of a graded transition between the two, which seems less appropriate to the intended usage of individuating the features.

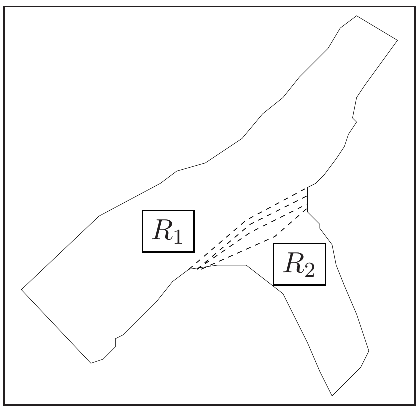


Figure 3.2: An example of determining possible boundary locations. Here, there are two rivers labelled R_1 and R_2 respectively, with the intention being to determine the boundary between the two. Some possible boundary locations are shown by the dotted lines. Fuzzy Logic on the other hand would have a graded region of possible boundaries.

With Fuzzy Logic, it is instead preferable to project the variations in observables, as opposed to generating sharp boundaries. This was shown by Hatzichristos and Giaoutzi (2005), where the different attributes under consideration are combined to generate an overall ‘suitability’ measurement, which is a number between 0 and 100. This scale is represented graphically by varying shades of grey, with 0 being white and 100 being black. Thus, rather than demarcating individual regions, areas that are most suitable are highlighted, as well as how abrupt or smoothly these transition to unsuitable areas.

The intended usage will impact upon the output and thus the approach used. As was noted previously in Section 2.2, an estuary can be defined both by its topology and by its salinity; thus, depending upon which output is required either Supervaluation Theory or Fuzzy Logic could be used. If an estuary is to be defined as a purely topographic feature, then Supervaluation Theory would be suitable. However, if instead the extent of the salinity of the water was to be considered, Fuzzy Logic could be to model the relative salinity as well as mark out candidates for the extent of the estuary.

3.6 Implementing Vague Reasoning

The key considerations when determining which of the two approaches mentioned is most appropriate for a given problem have been looked at. From these observations, a decision about which approach is to be used to work with the vague aspects of my case-study can be made.

The intended system takes topographical data as an input, and allows regions to be demarcated within this data representing inland water features such as rivers and lakes, as has been noted previously (Mallenby, 2007). Some of the key attributes for inland water features were identified by Ganter and Wille (1999), which included size and linearity, and hence these are the types of attributes that are intended to be extracted in order to determine features.

As noted above, Supervaluation Semantics lends itself better to these requirements than Fuzzy Logic. The use of Supervaluation semantics will allow crisp boundaries to be generated, as opposed to the fuzzy regions of Fuzzy Logic. Such crisp regions are easier to work with spatially, as although spatial reasoning with Fuzzy Logic is possible, it is not without complications.

In order to use Supervaluation Semantics, a method that uses the semantics as a framework needs to be implemented, such that a *precisification* can be determined from defined predicates. By itself, Supervaluation Semantics models vagueness in terms of possible interpretations, as opposed to giving any method of analysing semantic variability that can occur within natural language. Thus, this framework needs to be expanded to allow the parameterisation of such interpretations, to ensure that they can be modelled within the system.

The intention is that the observations that determine the classification of a feature are dependent upon the *standpoint* of a particular user, hence the approach will be referred to as *Standpoint Semantics* (Bennett et al., 2005; Third et al., 2007; Bennett et al., 2008). Standpoint Semantics is intended as a refinement of Supervaluation semantics, whereby the range of possible *precisifications* of a vague language is described using a (finite) number of relevant parameters relating to observable properties. The applicability of a particular predicate is thus defined by a set of thresholds assigned to these parameters.

For example, a predicate such as rivers may be based on several parameters, which may include size or flow for instance. With standpoint semantics, each of these parameters could be modelled by assigning them a threshold, which represents the distinction between the different instantiations for that parameter. Thus, flow could be a measure of the flow rate along the watercourse, and the threshold determines the distinction between ‘flowing’ and ‘stagnant’. The value of this threshold is dependant upon the user’s standpoint, thus if the user feels that rivers should be fast flowing they can set the threshold accordingly, and similarly if they view rivers as anything with a flow they can reduce the threshold.

This is represented throughout this thesis using the following notation (Mallenby and Bennett, 2007; Third et al., 2007): an *n*-ary predicate p has an interpretation that is dependent on *m* thresholds $t_1, \ldots, t_m$, represented as:

$$\mathsf{p}[t_1, \ldots, t_m](x_1, \ldots, x_n) \tag{3.6}$$

Thus the set of thresholds $t_1, \ldots, t_m$ represents a particular *standpoint* and are used to determine whether a particular predicate p is true or not for a given instance. To implement a predicate ‘tall’, where it is intended to show that above a certain height someone is considered tall, a predicate could be constructed to represent this as:

$$\mathsf{tall}[h](x) \tag{3.7}$$

where h is the height which a person is required to be greater than, or equal to, in order to be considered tall. This idea was expanded further by Bennett (2006), where there would also be included in the notation the ability to determine the context in which tall is being used, since clearly “tall” as applied to giraffes represents a different range of absolute heights from “tall” as applied to humans. For the purposes of this thesis, a standpoint is assumed to be used for a particular instance, hence the tallness of a giraffe would be a different standpoint to that of the tallness of a person, as opposed to a single standpoint for tallness.

For each threshold, it is important to define the values which will make the predicate true. With tall, this would be a single value, hence the predicate could be rewritten as:

$$\mathsf{tall}[h \geq z](x) \tag{3.8}$$

to signify that in order for x to be evaluated as being tall, the height h must be greater than or equal to z. On the other hand, a threshold may be dependant upon multiple values, defining ranges of values which will make a predicate true. For example, suppose a predicate normal was introduced in relation to height, to signify a person who is neither tall nor short (where short has previously been defined as being a person whose height is less than, or equal to, some threshold). This could be defined as:

$$\mathsf{normal}[y < h < z](x) \tag{3.9}$$

where h is the height of person x. Thus, normal depends on h being between both thresholds, y and z.

When a standpoint has multiple thresholds, the interaction between the thresholds may be complex. For example, suppose the following predicate was defined to represent a river:

$$\mathsf{river}[f > f_1, l > l_1](x) \tag{3.10}$$

where f is the rate of flow of the feature, l is the length of the feature, and f_1 and l_1 are the respective thresholds which these need to be greater than for x to be labelled a river. In this example, a river is therefore required to be both ‘flowing’ and ‘long’, and neither threshold has precedence over the other. However, there may be instances where one parameter being satisfied is sufficient; for example, if the length of the object was significantly larger than the specified threshold. Therefore, the computation of the thresholds may require multiple dimensions; for example, two thresholds could be plotted on a two-dimensional graph to determine what pairs of thresholds would mean the predicate evaluates to true. This would require further work, and is beyond the scope of this thesis.

Finally, if the definition of a feature contains predicates that use standpoint semantics, then that feature is, by inheritance, dependant upon the same standpoints. For example, suppose a predicate was introduced to represent large males, where large means someone who is tall and wide:

$$\mathsf{large\_man}[h, w](x) \leftrightarrow \mathsf{male}(x) \wedge \mathsf{tall}[h](x) \wedge \mathsf{wide}[w](x) \tag{3.11}$$

where h and w are the thresholds which a person’s height and width need to be greater than, or equal to, for these predicates to be evaluated as true. Because large_man is dependent on both of these parameters, the standpoint of the predicate contains both parameters. The predicate could have been written using the same form as river was previously, where the thresholds are built directly into the predicate. However, defining the predicate in terms of other predicates allows more control logically over the impact of the parameters. Thus, it is more suitable to define river in terms of a series of vague predicates rather than a single predicate with a standpoint containing a set of parameters.

3.7 Summary

This chapter has analysed the principal approaches to handling vague reasoning, looking at their strengths and weaknesses when applied to the geographical domain. Fuzzy Logic was shown to be best suited to situations where transitional boundaries are required, whereas Supervaluation Theory was shown to be suited to more crisp definitions. Therefore, it is better to consider what the desired outcome of reasoning about the data is, to ensure that the most suitable approach can be chosen.

It was also shown that Supervaluation Theory at least could be implemented within a system to handle vague features. By building upon the framework of Supervaluation Semantics, user preferences or standpoints can be used to determine *precisifications*, allowing reasoning to be undertaken in a given context, and thus allowing vagueness to be handled to some extent. How this may have been handled had Fuzzy Logic been used instead was also discussed.

Much of the decision on which approach to use still comes down to personal preference. There is much literature on the deficiencies (both philosophical and logical) of the respective approaches, and many people have strong views on the viability of these approaches, taking the viewpoint that only one is ever applicable. This need not be the case; rather it is better to consider the problem and intended output carefully. Since vagueness can arise in many different forms, it follows that handling vagueness can also be performed using different approaches, some more applicable than others for given instances. Thus, choosing a vague reasoning approach need not involve taking one stance and one stance only, but rather choosing the approach that suits the needs of the problem and thus using the most effective approach in a given situation.

Chapter 4

Data and Attribute Representation

4.1 Introduction

This chapter will show how to represent the data and attributes in such a way that reasoning about vague features within the domain is possible. Section 4.2 will give an overview of the problem domain, discussing the input data to be used. The representation of the data using a 'skeleton' will be considered in Section 4.3. The storage of data will then be considered in Section 4.4. Finally the results of this chapter are summarised in Section 4.5.

4.2 Overview of Problem

The problem domain considered in this thesis is that of inland water networks and the classification of inland water network features. These include features such as rivers, streams and lakes, which have previously been shown to be vague. The intended system, therefore, needs a method of representing this vagueness that will allow the user to segment, individuate and label such features, as proposed previously (Mallenby (2007)). This system expands upon earlier initial work by Bennett et al. (2005).

The system's initial input data was topographical data representing the Humber Estuary on the East coast of England. The initial input was obtained from the Global Landcover Facility (GLCF)[1], which were vectorised for input into the system. Further examples were obtained from the Ordnance Survey Digimap Collections[2], which were again vectorised for the system.

Providing the original network is a fully connected one (no part is disconnected from the rest), the initial input for the system would be a single polygon representing the inland water network. In Section 3.6, it was determined that for topographic data, Supervaluation Theory was the most suitable approach if the intended output is individuated features. Using Standpoint Semantics (that is based upon Supervaluation Theory), the user would enter values for thresholds to determine *standpoints*; thus when the system is queried to return these features it will return regions that have crisp boundaries that correspond to instances of the feature, as demarcated according to the chosen standpoints and associated thresholds.

The intended language to develop this part of the system is Prolog, which offers logical reasoning via Horn clauses. By developing this section within Prolog, interaction with the reasoning stage should be simpler, irrespective of the logical language used to develop the ontology and reasoning processes. Further, some imperative programming elements can be implemented, allowing the handling of some of the mathematical formulae that may be required to solve problems within this stage. A discussion of the different forms of Prolog will be given in Chapter 6.

4.3 Finding the Skeleton of a Polygon

In order to be able to reason effectively with the input polygons, an approach is required that represents information related to the polygons more efficiently. This includes being able to represent variation within the shape, as well as simple measurements such as width, area or other such size and distance measurements. When we think of features such as rivers, we often imagine an approximation of a line, whereby the variation in the width and the edges remains low. Thus some method of determining a skeleton of the input polygons is desirable, from which important characteristics can be derived with which to define features.

[1]Landsat ETM+ imagery: http://glcfapp.umiacs.umd.ed:8080/esdi/index.jsp (Visited, July 2006)
[2]Ordnance Survey Digimap Collections: http://edina.ac.uk/digimap/ (Visited, August 2006)

4.3.1 The Medial Axis

The Medial Axis of a polygon as first proposed by Blum (1973), is defined as the locus of the centre of all the maximal inscribed circles of the polygon. Here, a maximal inscribed circle is an inscribed circle that cannot be completely contained within any other inscribed circle in the polygon (Ge and Fitzpatrick, 1996). A simple example of this is shown in Figure 4.1, including examples of maximal inscribed circles within this context. The Medial Axis was proposed as a method for determining linear stretches of a river by Bennett et al. (2005), whilst McAllister and Snoeyink (2000) also discuss the benefits of using the Medial Axis in relation to inland water networks in more detail.

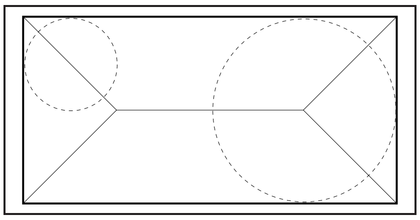


Figure 4.1: The Medial Axis of a simple polygon. The input polygon is represented by the thick black lines, with the Medial Axis by the thinner lines. The circles represented with dotted lines are examples of maximal inscribed circles, thus, the centre points of these circles are also points on the Medial Axis.

First, the centreline of rivers can easily be derived from the Medial Axis, as this will be a line that is equidistant from both banks. Further, this centreline generation can also be used to determine “opposite” banks of a river, since this approximates to finding the left and right sides of a line. Finally, depending on the construction used to generate the Medial Axis, it is possible to approximate the area of a given river network. For McAllister and Snoeyink (2000) this was possible as they use a Voronoi diagram based approach, and thus can use the Delaunay triangulation, the dual of a Voronoi diagram. A comparison of approaches to Medial Axis calculation will be discussed later in this section.

In a similar problem, the Medial Axis was used to approximate road junctions by Itonaga et al. (2003). This was used as a method of converting an image of a road network into a graph representation, which allowed a more effective representation of the overall hierarchy of the network, especially when combined with rules to solve ambiguities arising from the Medial Axis process. This has clear parallels with river networks, as it would be clearly beneficial to generate a simplified representation of a river as a graph or similar.

Whilst the Medial Axis is relatively simple to describe, it is computationally complex to calculate. Ideally, an approach would be developed to produce the Medial Axis of a given polygon, as well as to work in reverse and generate a polygon from a Medial Axis. Theoretically, the Medial Axis can be computed in linear time via histograms (Chin et al., 1999). However, as noted by Bose et al. (2006), the algorithm is quite complex due to the triangulation algorithm used, meaning the algorithm may only be of theoretical interest. An alternative approach, therefore, may be preferrable.

The three main approaches to calculating the Medial Axis can be grouped loosely as follows: The first is by finding the Voronoi diagram of the polygon, of which the Medial Axis is a subset. The second is to use distance transform based measures, which derives from the fact that a point on the Medial Axis is equidistant to two or more edges. Finally, fast marching or gradient/flux based approaches may be used, which derive from Blum's 'grass fire' description of the Medial Axis. Specifically, if you were to set fire to the edge of a polygon which then burned inwards and you recorded the location of the corners of the polygon at small time intervals, the end result would be the Medial Axis of the polygon, as the fires would converge to these lines. Thus, such approaches could be considered to be "thinning" the input polygon to find the Medial Axis. The second and third approaches are similar to each other, since both may involve calculating some form of distance transform across the polygon.

Computing the Medial Axis from an input polygon is required. Although this should be as efficient as possible, speed is not imperative as the Medial Axis for an input file can be pre-computed as opposed to "on-the-fly". A further requirement is to be able to store information related to the Medial Axis, such as the radius of the maximal inscribed circle at a given point on the axis, which represents the distance to the closest edge at that point. Finally, any approach used should allow some method of reversing the process such that a polygon can be generated from a section of the Medial Axis (or the original polygon to be generated from the full Medial Axis), either via the algorithm or by providing the required information to allow an approach to be developed.

4.3.1.1 Medial Axis Calculation - Voronoi Diagram Approach

The Voronoi diagram of geometric objects is a partition of space into cells, such that each cell represents the region closer to a particular object than any other object (Voronoï, 1907). In the simplest case, the Voronoi diagram of a set of points S is the partition of the plane which associates a region $V(p)$ with each point p from S in such a way that all points in $V(p)$ are closer to p than to any other point from S. An example of this is shown in Figure 4.2a. Given the distance based aspects of the definition, the connection between the Medial Axis and Voronoi diagram is evident; since the edges of the Voronoi cells

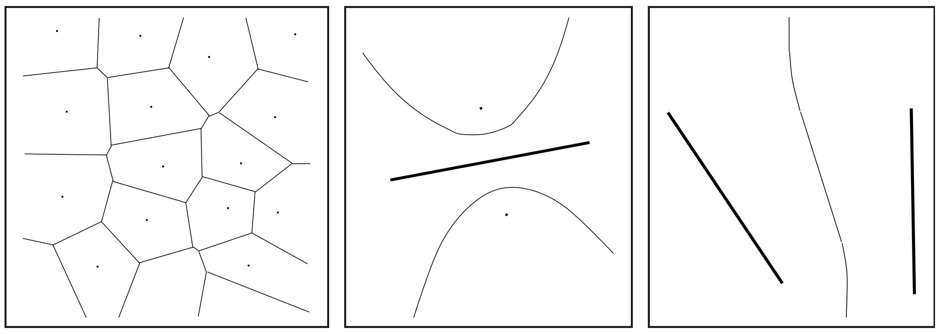


Figure 4.2: The Voronoi diagrams for mixtures of lines and points. For Figure 4.2a, the dots are the input and the lines are the Voronoi edges. In Figures 4.2b and 4.2c, thick lines represent the input and thinner lines represent the resultant Voronoi edges. Figure 4.2b has a mixture of lines and points, whereas Figure 4.2c has just lines.

represent points equidistant from two or more objects, the Medial Axis must be a subset of these edges. This was discussed in more detail by Lee (1982), where it was shown that for a convex polygon, the Medial Axis and Voronoi diagram of the set of edges of the polygon would be the same, whereas a concave polygon would require the removal of edges occurring at concave corners of the shape.

By deriving the Medial Axis from the Voronoi diagram, the exact Medial Axis can be obtained, as opposed to some approaches which need to approximate the distance calculations and thus the Medial Axis. Information such as the distance to the edge can also easily be extracted.

For a set of points, the Voronoi diagram is reasonably straightforward to implement. For example, an inefficient brute force method would be to first find the bisectors of all pairs of points, split these into lines wherever they intersect and finally remove all lines that do not satisfy the criteria for Voronoi diagrams. More efficient algorithms have been developed using a sweepline approach (Fortune, 1987), which have a complexity of $O(n \log n)$, though these can be difficult to implement effectively.

The Voronoi diagram of the set of edges of a polygon, however, is more complicated. The Voronoi diagram can be computed inefficiently from the set of bisectors of all pairs of points, thus the resulting diagram will also only consist of straight lines. However, when there is a mixture of lines and points or just lines, the calculations become more complex. The resultant edges are also more complex, with parabolas required to define the Voronoi edges generated between points and lines. Examples of these are shown in Figures 4.2b and 4.2c, where the introduction of lines increases the complexity of the resultant Voronoi diagram.

The relation between the Medial Axis and Voronoi diagram was shown by Lee (1982), as well as how this can be implemented to calculate the Medial Axis quickly for a polygon. This was further expanded by Chin et al. (1999) showing that the process could be performed in linear time, although in practise this is difficult to implement. Held (2001) has developed an approach called VRONI, which can handle a variety of inputs to generate the Voronoi diagram and derivatives such as the Medial Axis. The algorithm has also been implemented in C, and thus can be used without the need to generate code.

In conclusion, a Voronoi diagram based approach allows us to calculate the Medial Axis exactly and with useful information. However, the approach is dependent upon the algorithm used to generate the Voronoi diagram, which has been shown to be complicated for polygons.

4.3.1.2 Medial Axis Calculation - Distance Transform Approach

The Medial Axis of a polygon is the locus of all maximal inscribed circles of a polygon. A point on the Medial Axis therefore is equidistant from two or more points, as each maximal inscribed circle must touch at least two different edges. This follows from the definition of a maximal circle, requiring that a the circle cannot be completely contained by any other inscribed circle. The radius of a maximal inscribed circles represents the distance to the edge from a point on the Medial Axis.

The distance transform for a polygon can therefore be computed, whereby for each point in the polygon the distance to the nearest edge is calculated, and from this the Medial Axis can be derived. The advantage of this approach is the simplicity of the algorithm; if the points are represented as a grid the distance transform can be computed easily, and points can be compared to determine which satisfy the maximality criteria.

Because the inside of the input polygon is represented as a grid, the accuracy of the resultant Medial Axis is dependent upon the granularity of the grid used. This is discussed further by Meijster et al. (2000). A further problem of the algorithm is generating a polygon from the resultant Medial Axis. A simple approach would be to find the union of all the maximal inscribed circles being considered, but this may not be straightforward to implement. Also, it is not clear what edges a Medial Axis point relate to, although this may be possible to track with modification.

A distance transform approach was used by Ge and Fitzpatrick (1996), where it is noted that merely comparing the neighbourhood of a point may not be sufficient. Hesselink et al. (2005) present an approach which uses a simplified form of the Medial Axis called the Integer Medial Axis, which bases the calculations on a grid as mentioned above. Remy and Thiel (2002, 2003, 2005) have shown how the process can be simplified by the use of look-up tables, as well as alternatives to the Euclidean distance such as Chamfer distances, whereby the Chamfer distance between two points is the path between with the lowest cost (thus the path along a grid of points).

In conclusion, a distance transform approach represents a simple method of determining the Medial Axis, but may not be the most accurate due to the granularity of the grid used.

4.3.1.3 Medial Axis Calculation - Thinning Approach

The final set of approaches derive from the “grass-fire” description of the Medial Axis, where if we imagine burning the edges of our input polygon such that they move inwards at a constant rate, the Medial Axis will be the result of the location of the corners over time. Thus if the polygon is thinned progressively, the Medial Axis could be derived. The thinning progressively could therefore be computed over time, or alternatively gradient based approaches could be used to represent how the thinning would occur and derive the Medial Axis from the results.

Similar to distance transform approaches, thinning approaches can be simple to implement. For example, if marching methods were used we could track the distance over time easily and thus find the Medial Axis and the distance to the edge at each point. It is also possible to reverse the process to generate the input polygon, by expanding out until the distance is now zero (and thus have reached the edge of the polygon). However, like the distance transform approaches, the problem of accuracy can occur, as a grid is used instead of exact points.

Telea (2002) discusses an approach that uses the fast marching method to determine the Medial Axis. The aim is not only to track the distance to the edge, but also the edge which generated that point. A comparison of similar methods was also conducted by Reniers and Telea (2006), where the speed and accuracy of such approaches were compared. A flux based approach was proposed by Dimitrov et al. (2000), whereby the inside of the polygon is treated as a vector field and thus the gradient at each point represents the direction in which that point would thin using the “grass-fire” approach. The gradient field essentially represents how thinning would occur. From this, the flux is calculated, by finding the points where the gradients converge, which will represent the Medial Axis of the shape.

In conclusion, thinning methods may suffer from the same problems as other distance transform approaches, since they may not be the exact Medial Axis.

4.3.1.4 Medial Axis Calculation - Conclusion

Depending on the method chosen, considerations include the complexity of the algorithm as well as the accuracy of the final Medial Axis. In terms of simplicity, distance transform approaches are the most straightforward algorithms, but suffer from potential loss in accuracy depending on the granularity of the grid used. Voronoi diagram approaches are exact, but computation of the Voronoi diagram for a polygon can be complex.

The approach decided upon for this project was the Voronoi diagram based approach VRONI proposed by Held (2001). This was due to a number of factors: Firstly, attempts with other approaches did not produce accurate enough results, or the resulting Medial Axis had gaps due to the complexity of the input polygons. Although it may be desirable to simplify the Medial Axis to a simpler skeleton, the initial axis would ideally be as accurate as possible to reduce accumulative errors. Therefore, the Voronoi diagram approach of VRONI is preferable.

Secondly, the code has already been implemented efficiently in C. This saved time that would have otherwise been spent refining Medial Axis code. The code is designed to interface with other programs, although for this study it was sufficient for the calculation of the Medial Axis to be pre-computed and an input file generated.

VRONI stores the information related to the Voronoi diagram and Medial Axis in an efficient manner, which allows data such as the distance to the edge (as well as which edges are nearest) to be easily extracted, for example. This means it is possible to implement code to segment the image into smaller sections as required, as well as generate the original polygon from the Medial Axis.

In addition to Voronoi diagrams and the Medial Axis, VRONI can also compute other derivatives, such as the Delaunay triangulation (Delaunay, 1934), which is the dual of the Voronoi diagram. The Delaunay triangulation represents an effective method of triangulating a series of points, and may be of use in other sections of the geographical domain. For example, in forests each tree could be treated as a point and the Voronoi diagram and Delaunay triangulations calculated. From this, features such as density and shortest distance between trees could be calculated, which may be used in the segmentation of the data.

4.3.2 Refining the Medial Axis to a Skeleton

A method of calculating the Medial Axis for a given input polygon that is both efficient and accurate has now been determined. However, the Medial Axis is extremely sensitive to noise and variation along the edge of the input polygon, as shown in Figure 4.3. This is due to the additional corners of the polygon, whereby corners represent the point at which two edges join. For every corner whose inside angle is convex, there will be an additional arc in the Medial Axis that is connected to that corner. This follows from the “grass fire” definition.

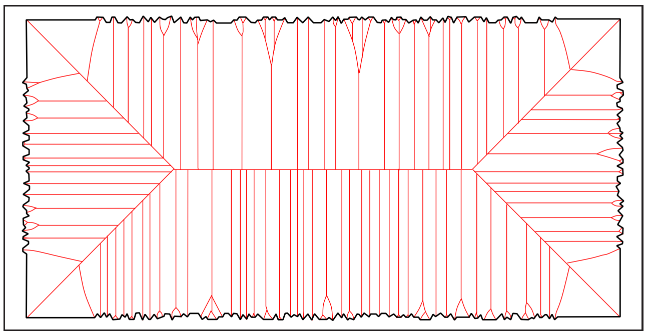


Figure 4.3: The Medial Axis of a rectangle with noisy edges. The Medial Axis is extremely sensitive to variation in the edge, resulting in a complicated skeleton. This may be acceptable if the exact axis was required, but if the general topology of the rectangle is of interest only, a method of pruning the axis is required, to generate the correct skeleton.

An approach is therefore required that allows the Medial Axis to be pruned, such that the resultant simplified skeleton retains the overall topology of the input polygon, whilst not removing any arcs that are required or the removal of which would result in disconnected parts of the skeleton. An overview of approaches is provided by Bai et al. (2007), which include previously discussed methods for Medial Axis extraction such as thinning and gradient based methods. However, it is also shown that these approaches do not guarantee a connected skeleton representing the topology; some arcs may be shortened too much or removed entirely, whilst more spurious ones may remain.

The approach used in this work was that of Contour Partitioning (Bai et al., 2006, 2007). Recalling the previously noted Medial Axis requirement that arcs are made up of the locus of the centres of maximal inscribed circles, it follows that such circles must touch at least two different edges. Contour Partitioning builds on this approach by instead using a series of connected edges referred to as a ‘contour’ (a ‘contour’ here defined as a successive sequence of connected edges), where skeleton branches in the final simplified skeleton consist of points whose associated maximal inscribed circles touch at least two ‘contours’.

An example of the results is shown in Figure 4.4, which shows the effect of using contour partitioning on the rectangle in Figure 4.3. The simplified skeleton is intended to represent the overall rectangle represented by the corners marked a, b, c, d. These form the basis of the contours used; $a - b$ represents the set of edges between a and b, $b - c$

all edges between b and c, $c - d$ all edges between c and d and finally $d - a$ all edges between d and a. Thus any Medial Axis point whose associated maximal inscribed circle only touches one of the contours is now removed, leaving the resultant skeleton.

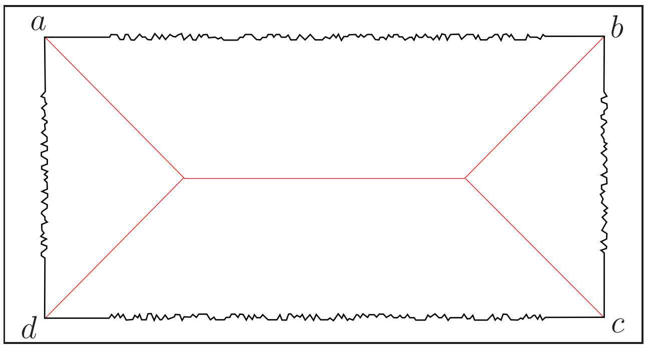


Figure 4.4: The simplified skeleton of the rectangle in Figure 4.3. Here, four contours were used, constructed as $a - b$, $b - c$, $c - d$, $d - a$.

A drawback to contour partitioning is determining the contours to use, particularly in determining an automatic approach. With Figure 4.4 for example, it is apparent visually that the four corners labelled a, b, c, d, that would resemble a rectangle, are most likely of interest, but implementing an approach of automatically deriving this could prove difficult. An initial attempt may be to use the convex hull of the shape, but this would not have generated the same results as Figure 4.4, as shown in Figure 4.5, where extra edges remain.

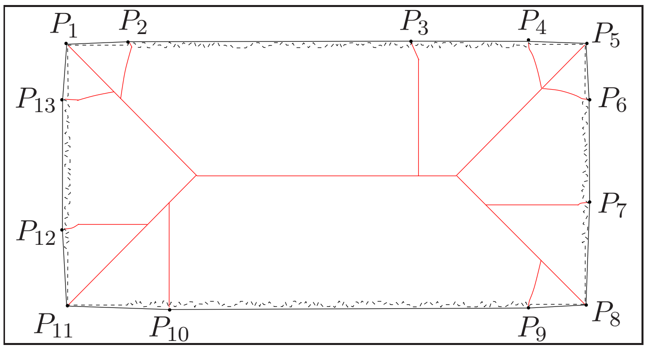


Figure 4.5: The simplified skeleton of Figure 4.3 when the convex hull is used to determine the contours. Here, the contours were generated from the points $P_1 - P_{13}$, which form the convex hull as represented by the continuous black line (the original input polygon is represented by the dotted line). These points generate the contours $P_1 - P_2, \ldots, P_{13} - P_1$.

The problem of automating the decision is discussed by Bai et al. (2007). Here, the aim is to determine which points have the highest importance, and use this ranking to determine what order to remove points. After each iteration of removing a point, the outline of the contour is compared with the input polygon using a similarity measure, where points are continually removed until the contour shape would no longer be similar enough to the input polygon (and thus not produce a skeleton that retains the input polygon's topology).

Although this approach sounds promising, it was found to be unsuitable for the inland water network data input here. For example, often there may be small offshoots (such as small harbours) that would be added as separate contours when it would, in fact, be more desirable to simply include them on contours either side of them (creating a single , larger contour). The approach could be adapted to provide an automatic solution, but it is beyond the scope of this work. For the input data, it is thus sufficient to determine the contours to be used manually.

4.4 Skeleton and Data Storage

4.4.1 Medial Axis Generation using VRONI

Having decided upon what approaches to use to collect and refine the skeleton, these need to be implemented within the code. As discussed in Section 4.2, the intended input data is vectorised from a black and white image. Thus the land is a series of polygons. This is converted into a format that VRONI can accept, to allow the Medial Axis to be calculated. The first refinement required is to select the correct side of the polygons on which to calculate the Medial Axis; depending on the direction of the input polygons (clockwise or anticlockwise), the left side of a polygon edge may represent the inside or the outside of that edge. This is easy to determine, and can quickly be rectified if the polygons are ordered in a different direction than expected.

The next decision required is that of how to handle the region that represents the sea, and the connection of this region to the inland water network. The problem domain examined in this thesis is that of inland water networks; thus whilst the region at the mouth of the river that connects to the sea may be of interest, the Medial Axis beyond that point is not of interest. However, VRONI will calculate a Medial Axis that extends around the input polygons, and thus will have many lines which are not of interest. This can be rectified easily by defining a bounding box surrounding the polygon that is slightly larger than the given input polygon in all directions, and not storing any Medial Axis points that are outside of the box.

A final change that is required of the code is to ensure storage of the boundary points that generate each Medial Axis node, allowing polygonal segments of the water to be

Table 4.1: The functions stored for the initial data. These are stored as asserted facts in Prolog. 'Function name' here represents a particular function , whereas internal storage would be shortened and concatenated.

Function Name	Arity	Attributes stored
Boundary Node	2	Identifier, co-ordinates of point
Boundary Line	3	Identifier, the two nodes making up the ends of the line
Boundary Polygon	3	Identifier, list of co-ordinates, list of points/lines
Medial Axis Node	4	Identifier, point co-ordinates and radius of maximal inscribed circle, degree of point, lines that node is end of
Medial Axis Edge	3	Identifier, the two nodes that form the ends of the line
Medial Axis Tangent	2	Identifier to link to Node, list of boundary points/lines the point is generated from

derived using the Medial Axis. These can be collected from the code, with calculation of the related edges occurring at the next stage. An initial input file can now be generated from VRONI that stores the polygons and Medial Axis with necessary information in a Prolog file, enabling the next stage of initialisation.

4.4.2 Initial Data Storage

The initial input from VRONI simply stores a series of points labelled as either being part of a polygon or the Medial Axis. This needs to be stored in a more logical fashion, allowing us to derive further information. For the boundary polygons, the ability to identify individual polygons is required, as well as to determine both what lines and nodes make up a polygon as well as what particular polygon a line or node is part of. A summary of how this information is stored is shown in Table 4.1.

First, each boundary node is stored in the form boundarynode/2, where that particular node is numbered with a unique identifier and the co-ordinates of the point stored. From this, all boundary lines can be found and stored as boundaryline/3, where each line has a unique identifier and the number of the two boundary nodes that make up the ends. Finally, this can be used to determine the points that are part of a particular polygon; as the polygons that are stored are simple polygons, the points can be easily searched and the identifiers of these points returned.

The storage of the Medial Axis must now be considered. As a simplified skeleton through contour partitioning is required, the information must be stored in a manner that will allow this process to be implemented. This stage could have been implemented within VRONI as suggested by Bai et al. (2007), but it can also easily be implemented in Prolog. The Medial Axis can be translated into a graph structure, which is a logical candidate for representation of the Medial Axis, given the sort of queries that may be performed on the data, such as determining loops or paths between nodes. The simplified skeleton will therefore form a subgraph of the Medial Axis graph.

The nodes of the Medial Axis are stored as medialaxisnode/4, which, as stated previously, begins with an identifier. Next, the co-ordinates and the radius of maximal inscribed circle at that point are stored as a triple, point/3. As the Medial Axis is to be stored as a graph, the degree of the node is also stored, as well as the lines that node is part of. As with the boundary, the edges can be stored as medialaxisedge/3, where the nodes that make up the ends of the line are stored. This structure inherently stores a graph, and thus queries can be written to perform graph functions, such as finding paths, using this stored structure. However, Prolog has built in graph functions with optimised functions already incorporated, thus the information is also stored as a Prolog-style graph, to take advantage of these optimised functions. The boundary polygons as graphs can also be stored as graphs, which will aid in other calculations.

To perform contour partitioning, the contours which a Medial Axis node is related to need to be known. The points on the boundary that a Medial Axis node is generated from are already calculated in VRONI, thus these co-ordinates must be converted into the associating points on the boundary. These may already be existing boundary points, but in the case that they are not, the line which that point is part of needs to be calculated. These are stored as tangent/2, where for each node a list of the co-ordinates of the points on the boundary that are tangential to that node are stored, as well as the associated boundary nodes or lines for each tangential point.

All the required information to perform contour partitioning is now available. Here, contours are defined as paths in the boundary graph, such that nodes are part of only one path (with the exception of the end nodes which are part of two). Further, all nodes are part of at least one path, thus the contours form a path that visits all nodes of the boundary graph once in a loop. It is beyond the scope of this project to develop an automatic approach to determine the contours, though a potential set of end nodes can be determined, from which the final set will be chosen. As the contours that form banks of features are of interest, suitable end nodes of contours will be points that split two banks, and hence will usually be points where the inside angle is very small in the middle of a series of points with inside angles close to $180°$. An example of this is shown in Figure 4.6. This approach is similar to the one suggested by Bai et al. (2007), but requires some manual 'fine tuning' since some points may need to be removed to obtain the final set of end points to be used.

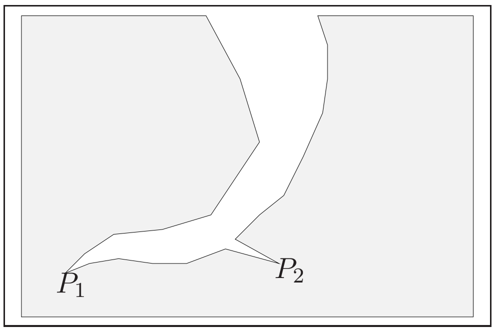


Figure 4.6: An example of suitable end points for contours. The shaded region represents the land, thus the inside angle measured is the angle generated at corners with respect to the water. P_1 is a suitable candidate for a counter end point, as the inside angle is extremely small, whilst the inside angles of the points preceding and following it are very close to $180°$. P_2, on the other hand, has a very small inside angle, but the points directly preceding and following are much greater than $180°$, and thus it may not be suitable

The contours for the main land polygon are now known. For islands, the island polygon can be treated as a single contour (unless a particular inlet on the island is of interest, and thus the process above would need to be repeated). It is now easy to determine which contours a point lies upon, as the boundary lines or points a Medial Axis point is generated from have previously been stored. Thus for each point, the contours that point is associated with are determined, with any points pruned off (and the edges connecting them) where the associated maximal inscribed circle for that point only touches one contour. The simplified skeleton has now been generated, which can be stored in a similar method as used for the Medial Axis, using a graph structure.

4.4.3 Polygon Storage

An effective approach to storing information about the polygons is now required; both the initial polygons and any future ones that may be generated. One option is to simply store them the same as the initial boundary polygons; a list of the consecutive corners of the polygon. However, whilst this may be the simplest approach initially, there are more efficient ways to store the data to improve efficiency of operations that may be performed upon the data at a later stage. The Winged Edge structure (Baumgart, 1972) and variations such as the Half-Edge Winged structure (Mantyla, 1988) offer a more effective representation of polygons, as opposed to simply storing the corner points. An example of the Half-Edge structure is shown in Figure 4.7.

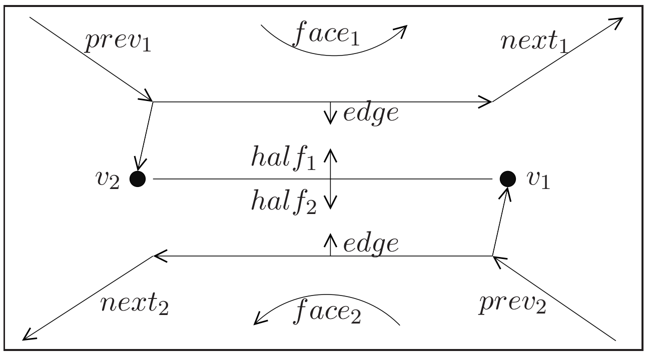


Figure 4.7: An example of the Half-Edge Winged Boundary representation, adapted from the example given by Mantyla (1988). Each edge in fact consists of two half-edges; $half_1$ is ordered as $v_2 - v_1$ and $half_2$ is ordered $v_1 - v_2$. The half-edges are associated with $face_1$ and $face_2$ respectively, $prev_1$ and $prev_2$ are the edges preceding each half respectively, and $next_1$ and $next_2$ are the edges that follow.

In the Half-Edge structure, an edge consists of two half-edges ordered in opposing directions. In Figure 4.7, the edges around each face are ordered anticlockwise, thus for each half-edge the inside is considered to be on the left of the edge. For each half edge, the edges that precede and follow them along that face are also stored, which can be determined by the ordering of the half edges. The structure can easily be modified to allow for multiple preceding and following edges.

With the half-edge structure, simple queries can already be performed, such as determining if two polygons share an edge. If they do, there will exist an edge whose half-edges are the duals of each other. The union of such polygons can also be calculated by following the path around a polygon and switching to the other polygon when a half edge shared by both is reached. However, this relies upon assumptions such as that there are no overlapping edges (though it can easily be adapted to accommodate this). Initial polygon data can be easily translated into such a structure: First, all polygons and vertices are labelled individually, then all edges are found and stored as two half edges in opposing directions. For each edge, the edges that follow and precede (in the same direction) are then determined, as well as the inside face for that half edge. Whenever a new polygon is added, only the edges of that new polygon need updating, therefore the structure can be easily maintained and updated.

4.5 Summary

This chapter has discussed how topographical data can be prepared for input into the proposed vague reasoning system. It has been shown how the problem domain needs to be considered to determine the most effective representation of the data. The Medial Axis has been shown to be a powerful method of representing the initial data, as well as how contour partitioning can help overcome the sensitivity of the Medial Axis to produce a simplified skeleton that retains the topology of the data. Finally, it was shown how this data can be effectively stored in the system to allow queries to be performed upon the data.

Chapter 5

Attribute Collection and Data Segmentation

5.1 Introduction

This chapter builds upon the initialised data generated in chapter 4 to implement approaches to extract attributes from the data and use it to segment the input data into polygons corresponding to possible interpretations of vague feature terms. Section 5.2 will show how this data can be segmented into polygons. Section 5.3 discusses how attributes can be collected from the data to segment the data into polygons. These attributes can then be used in the grounding section that will be discussed in the next chapter. Section 5.4 will show how the spatial relations between the polygons can be calculated, including how to calculate line intersections and point locations. Finally, the chapter is summarised in Section 5.5.

5.2 Data Segmentation

From chapter 4, an initial data set consisting of the original land polygons has been derived, and a skeleton formed from pruning the Medial Axis of the inland water region. This skeleton is stored as a graph, where the nodes store the smallest distance to the edge (the radius of the maximal inscribed circle at that point), as well as the points that the maximal inscribed circle touches the edges at. The original inland water polygon from which the Medial Axis was generated can also be stored. However, as was previously noted, the intention is to individuate features that are contained within a given inland water network; thus, a method of segmenting this original polygon into segments corresponding to geometric measurements is required, which themselves may correspond to vague terms.

This was the motivation behind deriving the Medial Axis, as it provided a method of extracting information from the original polygon that may be of use for segmenting into smaller regions. An example attribute that may be measured is that of linearity (as will be discussed in Section 5.3.1), whereby the intention is to mark linear regions depending upon the variation in the widths along the skeleton. To do this, stretches of the skeleton that correspond to a given measurement of linearity are determined, then the polygon associated with that stretch is generated, in effect reversing the Medial Axis process. Thus, segmenting the data into smaller polygons is an important part of the intended system.

Generating polygons from segments of the Medial Axis has been attempted previously (Mallenby, 2007). As already noted, the Medial Axis generated by VRONI can be represented as a graph with a series of vertices connected by edges. For each vertex on the skeleton, the radius of the maximal inscribed circle at that point is also stored, which is also the shortest distance to the boundary. The boundary edges that touch this circle will be tangential, and hence the line from the centre of the circle to the tangent point will be perpendicular to the boundary line (or will be a boundary corner).

Thus, when using VRONI, these facts can be used to determine, for each vertex on the Medial Axis, points on the boundary that the maximal inscribed circle touches. From this set, for each skeleton edge the vertex is part of, two boundary points are determined to associate with that edge, such that for each edge there will be four boundary points associated with that edge[1].

This is straightforward when there are only two possible points associated with each vertex, as a boundary point each side of the skeleton edge need only be determined. However, if there are multiple potential boundary points, some method of determining which boundary points to choose is required. An example of instances where there may be more than two boundary points associated with a vertex is given in Figure 5.1. If a skeleton edge consisted of two points like P_2, there are only four boundary points that can be associated with that edge, as P_2 only has two associated boundary points (T_5 and T_6). However, if one or both of the points was similar to P_1 or P_3, there would be more than four boundary

[1]Exceptions to this are when the vertex is also a boundary node, or two separate vertices share the same boundary point. In these cases, there may be less than four points associated.

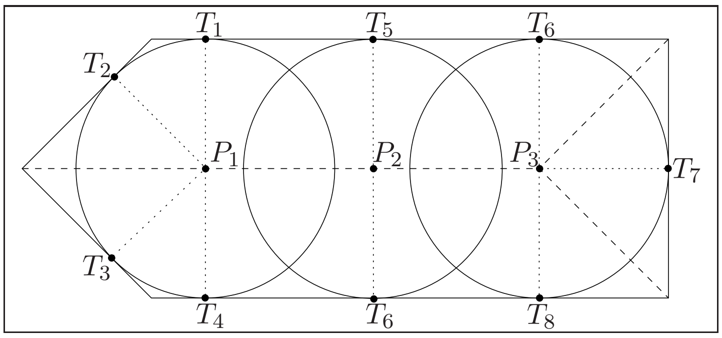


Figure 5.1: An example of the number of points on the boundary which maximal inscribed circles touch varies along the Medial Axis. At P_2, there are only two different points on the boundary, which is the minimum number of points a maximal inscribed circle must touch. At P_1, the convergence of the boundary edges to a single point means there are now four different points on the boundary touching the maximal inscribed circle ($T_1 - T_4$. Finally at P_3, there are three different points on the boundary touching the maximal inscribed circle ($T_7 - T_9$) due to the two boundary corners near to that point.

points in total associated with that edge. Hence, some method of choosing a subset of these points would be required if only four were required.

To do this, the properties of tangents of circles are again used; in this case, the properties of tangents to two circles. For the two vertices for a given skeleton edge, the associated circles will have two tangent lines that each touch the circles at one point only (as shown in Figure 5.2). The boundary points that are closest to these tangent points are then chosen, and the polygon for that particular edge is constructed accordingly. For point P_1 in the figure, a boundary point is chosen from the set of possible points that is closest to T_1, then repeated for T_4. Similarly, for P_2 a point that is closest to T_2 on the boundary is chosen, and a point closest to T_3.

For each skeleton edge generated by VRONI an associated polygon can be generated, with examples of the possible combinations shown in Figure 5.3. In addition, there are variations of the polygon in Figure 5.3a, whereby one (or both) of the boundary edges is in fact a single point. Finding a polygon associated with a series of connected skeleton edges therefore, can be achieved by performing a spatial union operation on the set of polygons associated with the set of edges in question.

This approach had deficiencies however. Firstly, finding the union of polygons was not a simple task, as they were not stored in an efficient manner such as the half-edge approach implemented in Section 4.4. Another problem was that of ‘inlets’ along the boundary (Mallenby, 2007), as shown in Figure 5.4.

Figure 5.4a shows how this can occur from a single skeleton edge. For each skeleton vertex there are two associated boundary points, which are labelled as being to the left

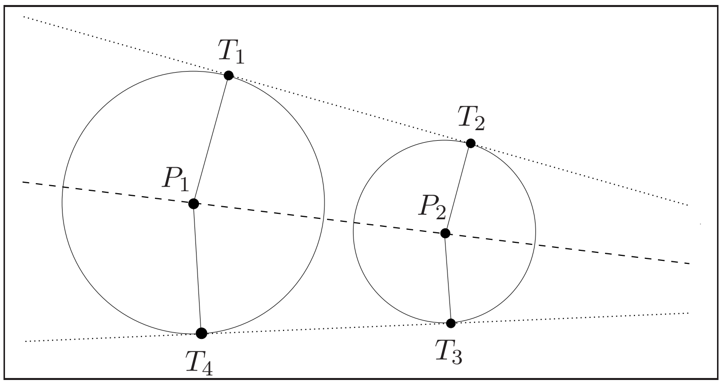


Figure 5.2: An example of the two tangent lines that two circles share. The thick dotted line represents the skeleton, with points P_1 and P_2 representing points along this skeleton and the circles for each representing the maximal inscribed circle at each point respectively. The thin dotted lines represent the two tangents which touch each circle at one point only, at points $T_1 - T_4$. These points are used to determine which boundary point to associate with each of the skeleton vertices for the skeleton edge $P_1 - P_2$.

or right of the skeleton edge, using an orientation determined from ordering the skeleton vertices numerically[2]. From this, the 'left' boundary edge is made from nodes $N1$ and $N2$, by directly joining the two nodes using the shortest distance between the two. But the path between the two along the boundary actually deviates from this straight edge, as shown in Figure 5.4a. Because only the shortest distance between the boundary nodes is used, the polygon edge may deviate from the boundary edge and generate inlets.

Figure 5.4b can occur when the union of connected polygons is performed. For each skeleton vertex there is a set of associated possible boundary points, with the points chosen dependent upon the skeleton edge in question. The number of points however, will impact upon the union of two polygons. In Figure 5.4b, vertex V is shared by two polygons P_1 and P_2, and has associated nodes N_1, N_2, N_3. For the bottom portion of the polygons, the union is straightforward as both use N_1. However, along the top portion there are nodes N_2 and N_3, thus a path between the two nodes needs to be determined. If a normal spatial union was performed on the two polygons, the edges N_2V, VN_3 would be used as contained above, which leaves the gap shown in the figure. Alternatively, the two nodes could be joined using the shortest line between the two (thus replacing N_2V, VN_3 with N_2N_3) (Mallenby, 2007). However, this once again generates an 'inlet' at the edge of the polygon, where the marked polygon does not extend all the way to the edge. A close-up of an example of the inlet problem is shown in Figure 5.5.

One suggestion for handling these inlets was to use Supervaluation Theory (Mallenby,

[2]As noted in the previous chapter each vertex is numbered uniquely, hence a skeleton edge is ordered using these numbers, ordering an edge from the lower numbered vertex to the higher number.

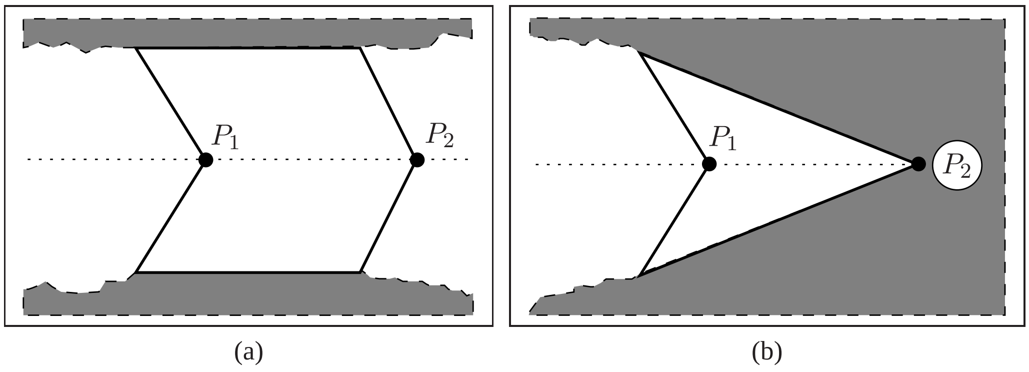


Figure 5.3: An example of the polygons original stored by the system. In both cases, the grey polygon with dashed lines represents the land polygon (the boundary), and the white polygon demarcated by the continuous black line is the polygon associated with skeleton edge $_P1 - P_2$. The thin dotted line is to represent an approximation of the skeleton.

2007) , since sometimes you may wish to skip over the gap and other times you may wish to fill it. However, an improved representation of the data as shown in Section 4.4 allows for an improved segmentation approach. In particular, the storage of the land polygons as graphs also allows us to use graph functions to rectify problems.

Once again, each skeleton vertex generated by VRONI has a set of boundary points associated with it. However, rather than precomputing each polygon in advance for each skeleton, the boundary points which are required for a skeleton edge (or set of skeleton edges) are determined, and the required associated polygon is generated from these.

An example of the process is shown in Figure 5.6. For each skeleton edge, a left and right side is designated (depending on the order of the vertices): First, a boundary point to associate with each of the vertices on each side is chosen, to use as the key polygon construction points. This set will consist of six points; the two vertices plus two boundary points on the left side of the edge and two points on the right (except in the case where one of the vertices is also a boundary point, where again that point can be used).

If both vertices are of degree two, this process is straightforward, since one boundary point will fall on each side of the edge for each vertex. In Figure 5.6 for example, nodes N_1 and N_3 are both of degree two, thus it is clear which boundary points to use for them. However, at N_2 there are three possible boundary points to choose from. The solution used here is to use the boundary point that is closest to the opposing vertex on the edge. This improves upon the previous tangent approach, as it is simpler and quicker to implement.

To generate the polygon associated with $N_1 - N_2$, the ‘left’ and ‘right’ sides of the skeleton edge are first determined. This is performed as before by ordering the edge numerically (thus N_1 to N_2), and thus using this as the orientation of the edge. In Figure

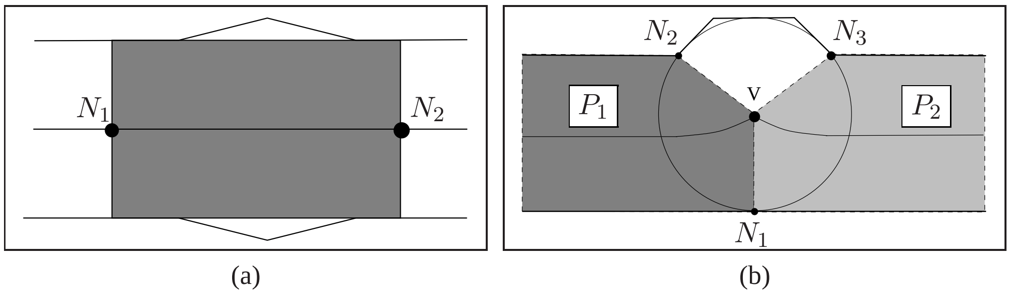


Figure 5.4: Examples of errors that could occur in the polygon generation approach implemented previously (Mallenby, 2007). In Figure 5.4a, errors occur due to the limited number of boundary points used in the polygon generation. In Figure 5.4b, Polygons P_1 (the darker polyogn) and P_2 (the lighter polygon)represent two polygons stored within the system. If the union of these was to be formed, the result is a region similar to the one marked by the thick border. One option would be to add edge $N_2 - N_3$, but there would still be a gap that is not part of a region.

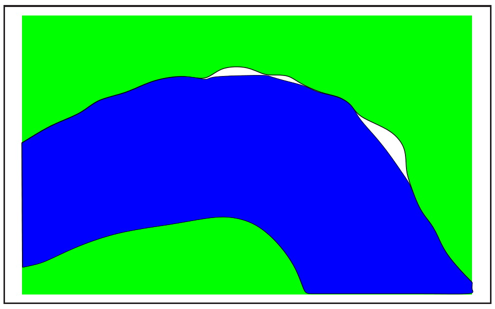


Figure 5.5: A close-up of inlet errors generated in previous implementations (Mallenby, 2007). The green represents the land, and the blue represents a segmented water polygon. The white regions are gaps where the water does not stretch to meet the boundary, and thus inlets form.

5.6, the left side is the top side, and the right side is the bottom side. For the left side, there is only a single boundary point for each vertex to consider, hence the choice of boundary nodes is complete. For the right side, N_2 has two possible candidates, B_1 and B_2, from which B_1 would be chosen, as it is closer to N_1 than B_2.

Now that the key points have been determined, the next phase is to generate the two sides. Previously, these points would have formed the complete polygon, but gaps appeared as highlighted in Figure 5.4. Instead, the graph of the boundary can be used to generate sides to ensure that the polygon fills right to the edge. The boundary graph is a weighted graph, where the length of an edge is also the weight of that particular edge. To generate an edge between two boundary points, the minimum path in the weighted graph between the two is found, which is the shortest path of connected boundary edges

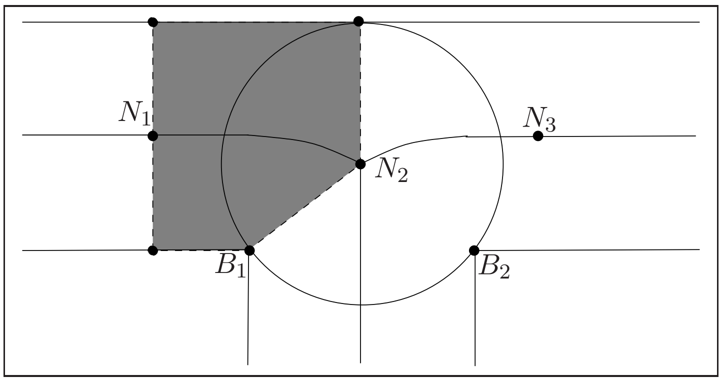


Figure 5.6: An example of how a line is translated into a corresponding polygon. For this shape, the line being considered is $N_1 - N_2$. For N_1, both boundary points associated with that node are used. For N_2, B_1 is chosen as it is closer to N_1. The shortest path along the boundary between successive boundary points is then traced, with the grey region demarcated by the dotted line representing the resultant polygon.

between the two. Once these are generated, each of the vertexes are joined to the two sides, creating a single loop which is the resultant polygon. For Figure 5.6 therefore, the polygon generated from $N_1 - N_2$ would be the area outlined by the dotted line.

This approach easily adapts to generating the polygon for sets of lines, as a left and right can be determined for all edges, the boundary points found on each side and the minimum path between each consecutive set of points found. If no path exists between two nodes since one is on a different land polygon to the other, a direct edge between the two points is added instead. For example, in Figure 5.6 if B_1 and B_2 were on different land polygons and the polygon in question was for $N_1 - N_2 - N_3$, the edge $B_1 - B_2$ would be added to the polygon at that point.

Another potential problem arises if the minimum path between two points is extremely large. The solution used is to compare the length of the path with the distance between the two nodes, and if the path is significantly longer than this distance the direct line between the two nodes is used. Examples of this are shown in Figure 5.7. In Figure 5.7a a small path along the boundary connects B_1 and B_2, thus this path is used to connect the two in the resultant polygon. In Figure 5.7b however, the path stretches off for a significant distance, thus the edge $B_1 - B_2$ would be added instead. One option would be to treat this as a type of vague feature (Mallenby, 2007), but the distinction between the two is more clear cut than with other features; if the distance between the two points is extremely small in comparison to the length of the minimum path then it is better to skip over rather than use the minimum path along the boundary graph.

By using this approach, various properties of the generated polygons can be guaranteed. First, the generated polygon will be simple and have no self intersections. This

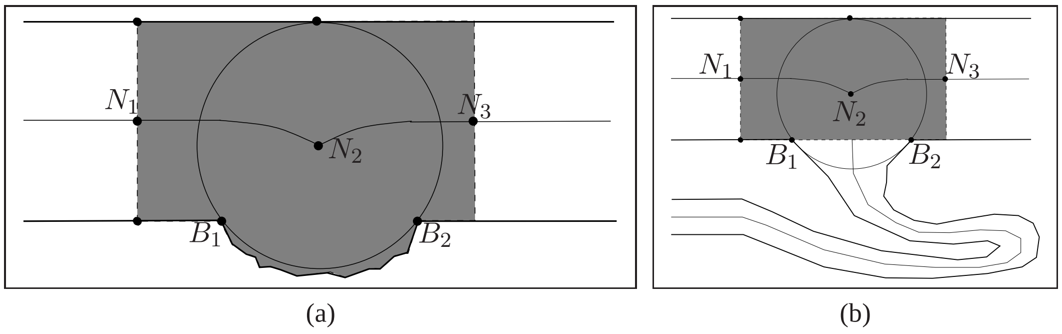


Figure 5.7: Examples of two different approaches to handling the path between nodes B_1 and B_2. In Figure 5.7a, the boundary path between the two points is quite small, hence the path between the two is used, thus filling in to the edge. In Figure 5.7b, the boundary path is significantly longer as there is another channel stretching from the two points (the skeleton line is an approximation), thus a direct line between the two points is used.

follows from the fact that minimum paths between points are used to create edges in the polygon, as well as tracking the points that have been used. The boundary polygons themselves are simple, thus the graphs of each boundary polygon will be a single loop. Using the minimum path each time effectively marches along this graph towards the final goal, thus never generating an edge that intersects another edge within the final polygon. By tracking which points have already been visited, it can also be ensured that paths back to points that have already been included are not added; thus the approach will always march towards a final point.

The next property that arises from this is there is a one to one mapping between polygons and skeleton edge sets that generate them. If all possible polygons were generated, there would be a unique set of edges that generated each polygon, and each polygon would be generated from only one set of edges: First, if the polygon generated by a single skeleton edge is considered, it is unique to that edge as even if another edge generated the same left and right sides, the end skeleton vertices differ and thus the polygons will differ. Thus, if two sets of edges differ by at least one edge, there will be at least one part different in each polygon. Thus, there is a unique mapping between polygons and skeleton edge sets.

A method of taking a subset of the skeleton graph and generating an associated polygon has now been determined.

5.3 Attribute Collection

The data is now stored in an efficient manner and a method of generating polygons from the skeleton has also been determined. The next step is to determine ways of measuring attributes of the skeleton and use these to generate polygons to be used to reason about the features. The motivation for this is the relation between the geomotery of an object and its classification, which allows a person to identify geographic features on a map based on the shape of the feature.

This is because the way a particular feature interacts with the surrounding terrain will impact upon its shape, hence the shape may allow a person to infer properties of a feature despite a lack of data on other properties. For example, flow (or lack of) is often identified as a key attribute of rivers or lakes (Ganter and Wille, 1999), but data on flow rates is not always readily available. However, because of how water interacts with terrain, it may be possible to infer whether a particular region of water is ‘flowing’ or ‘stagnant’ from its shape, and hence identify rivers and lakes from their geometry. For example, rivers flow downhill from a source to either a lake or the sea. The path that the river course takes will depend on the underlying terrain, as the water will forge a channel through softer terrain, whilst flowing around harder terrain. The expected shape of a river, therefore, would be a long, narrow channel winding through the terrain, since the wider a river, the stronger the flow required to forge the channel. Similarly, because lakes are basins that rivers flow into, the shape would be expected to be more bulbous, as the water fills outward rather than continuing to flow along a channel. The geometry of these features, therefore, could be used to classify them, as a proxy for flow.

In addition, the shape can be used to determine if a feature might be artificial or natural. For example, as already noted rivers will flow around hard, rocky areas, following the path of least resistance. However, features such as canals are usually designed to follow the shortest path possible between two locations, and thus could be dug through rocky areas in attempt to keep the channel as straight as possible. This lack of curvature could be used to identify the difference between canals and rivers.

The attributes related to the shape properties of a feature should be measurable irrespective of the size of the shape, as, for example, the definitions for ‘stream’ and ‘river’ may be dependent upon the notion of a channel (with streams and rivers being small and large channels respectively). Thus, the definitions for channel used should allow for channels of all sizes to be identified, with an additional size measurement applied afterwards to perform the actual distinction between streams and rivers.

5.3.1 Linearity

As mentioned above, the shape of a water feature could be used as a proxy for the overall flow rate across the feature, with rivers tending to be long and narrow and lakes more bulbous. The attribute that will attempt to identify this is referred to here as 'linearity', due to the correlation between linearity and flow (Bennett et al., 2005); because flow gives rise to linearity within geometric data, it means linearity is an indication of flow within data.

With linearity, there is an intuitive idea of it corresponding to something like 'the extent to which something can be described as "long and thin" '. However, this is not necessarily something that can be directly measured from data, meaning linearity may be defined as whatever formally defined property that is used to capture or approximate this intuitive notion.

To expand this, the use of 'linear' here does not mean straight or unchanging as some may define it, but rather that the variation in the feature relative to primarily some measure of width is small or non-existent. Such features can be curved for instance, providing the curvature is not too extreme. As noted previously, artificial features such as canals would be expected to be as straight as possible, whereas rivers would be expected to wind across the terrain. The linearity measurement, therefore, should be able to capture this fact, allowing rivers to have curvature. Thus, linearity is intended as a measure of near uniformity of width here.

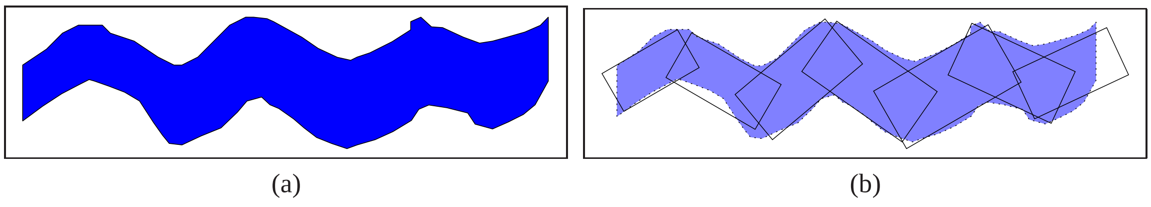


Figure 5.8: An example of a 'linear' feature. Figure 5.8a shows an example of a water feature, which may be considered linear due to the low variation in width along its course. This is highlighted in Figure 5.8b, where a series of rectangles have been overlaid.

An example of this is shown in Figure 5.8, with a potentially linear feature given in Figure 5.8a. This intuitively looks linear, as it fits the description of being "long and thin". Figure 5.8b shows the feature with a series of rectangles overlaid, which shows the low variation in width along the feature. Therefore, any formally defined property used to capture linearity should be able to capture parts of a feature which may approximate to rectangles (low variation in width). In addition, as the feature may curve (as in Figure 5.8a), the property should be able to connect these approximately rectangular parts into a single connected feature as appropriate (for example, where rectangles overlap).

This is a natural way of considering a river, as mentioned by Third et al. (2007), where the notion of a river being considered as an approximation of a line is discussed. If we were to look at a topological map, we would expect the rivers and streams to be long, narrow features where the sides follow roughly the same path, whereas the lakes and ponds would be more bulbous, short features with a large degree of variation in the sides (expanding and retracting to and from each other at a sharp rate).

Such an attribute is clearly vague, since it not only is uncertain what the boundary between linear and non-linear is, but it is also uncertain how you may measure linearity. The intended usage here of linearity is that it a linear feature is one that has some degree of uniformity of width about it; although parts may widen or shrink the sides of the feature stay reasonably uniform relative to each other and the width along the channel remains reasonably uniform.

The measurement for linearity used here was performed in relation to the variation in the width of parts of the input polygon along the polygon's skeleton (Mallenby, 2007). This is not the only method of measuring linearity, and further work would be needed to compare different possible approaches to the calculation. This measurement should be designed to ensure that large and small rivers or streams can be classified using the same attribute. An example of what information is extracted from the data to be used in the calculation is shown in Figure 5.9.

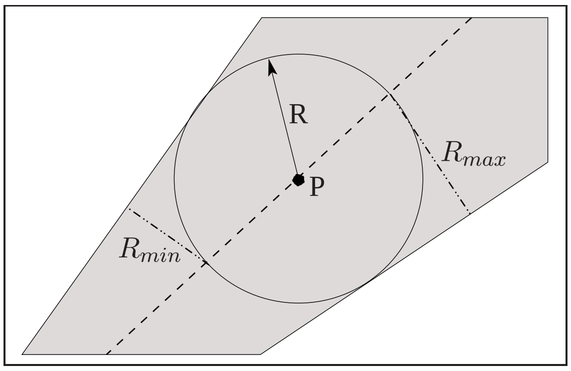


Figure 5.9: An example of how linearity can be determined with respect to the skeleton. Here, point P has an associated maximal inscribed circle of radius R. Of all the skeleton points within R, the points with largest and smallest radii are at the edges of the circle, and hence the minimum and maximum radii to use are R_{min} and R_{max} respectively.

This approach will now be described in detail. As shown in Figure 5.9, for each node P on the simplified skeleton, there is an associated maximal inscribed circle of radius R. To determine the linearity of that point, all the skeleton points that fall inside that maximal inscribed circle are found (and hence are of distance less than or equal to R

from P). The radii at these points is noted, finding the minimum and maximum radii. These can easily be found using the graph structure, by determining the possible paths from P and including all points whilst the length of the path is less than the radius R. In Figure 5.9, these are marked R_{min} and R_{max} respectively.

Next, the ratios $\frac{R_{max}}{R}$ and $\frac{R}{R_{min}}$ are determined and compared against the value of the threshold for linearity. If both these ratios are lower than the threshold, the point is considered linear, else it is non-linear. Thus, for a given threshold L, point P is marked as linear iff the following equation holds:

$$\mathsf{MAX}(\frac{R_{max}}{R}, \frac{R}{R_{min}}) < L \tag{5.1}$$

where MAX is a function returning the maximum value of the two inputs, and R,R_{min} and R_{max} are defined as described previously. By relating the measurement to the set of points within R and using R,R_{min} and R_{max} in the calculations, the approach can be used for all points on the skeleton, irrespective of width at that point. Larger values of R will result in more values being considered, whilst smaller values will consider fewer accordingly. This is more suitable than specifying a set distance to search around a point, since a set value may not be suitable for all points along the skeleton.

As noted, there are other possibilities for measuring linearity, including how the widths at points along the skeleton are used. For example, an alterative would be to compare the ratio between the minimum and maximum directly rather than against R, such as $\frac{R_{min}}{R_{max}}$. Another option is to consider the different paths from P; in Figure 5.9, there are two paths from P, which could me marked as to the left and right of P respectively[3]. These paths could therefore be tested separately, e.g. $\frac{R_{min}L}{R_{max}L}$ and $\frac{R_{min}R}{R_{max}R}$ where R_L represents the radii of points to the left of P and R_R the radii of points to the right of P. The linearity of P could then be dependent upon being linear in both directions, or just in one direction.

In theory, all points on the skeleton would need to be tested for linearity, as was originally performed by Bennett et al. (2005). Using the information from VRONI, it is possible to calculate, for any point on the Medial Axis, the radius of the maximal inscribed circle at that point, as shown in Figure 5.10. In the example P_1 and P_3 are points stored in VRONI, with P_2 not stored. The Medial Axis is a straight line between the two points, with the radii of the maximal inscribed circles along the Medial Axis increasing at a constant rate (where this not the case, VRONI would store extra points

[3]The determining of the 'left' and 'right' of P is not important, and could also be labelled 'before' or 'after' P for instance. The main point is that the two sets represent two different directions away from P along the skeleton graph

to indicate this). Thus, for any edge P_1P_3 in VRONI, the radii of the maximal inscribed circle along the edge will increase at a constant rate from the minimum radii of the two points to the maximum. As shown in Figure 5.10, this means that the radius at any point along the edge can be calculated using similar triangles. Thus, the radius at all points could be calculated from previously stored information. A downside of this, however, is the additional overhead in increased calculations and points to consider.

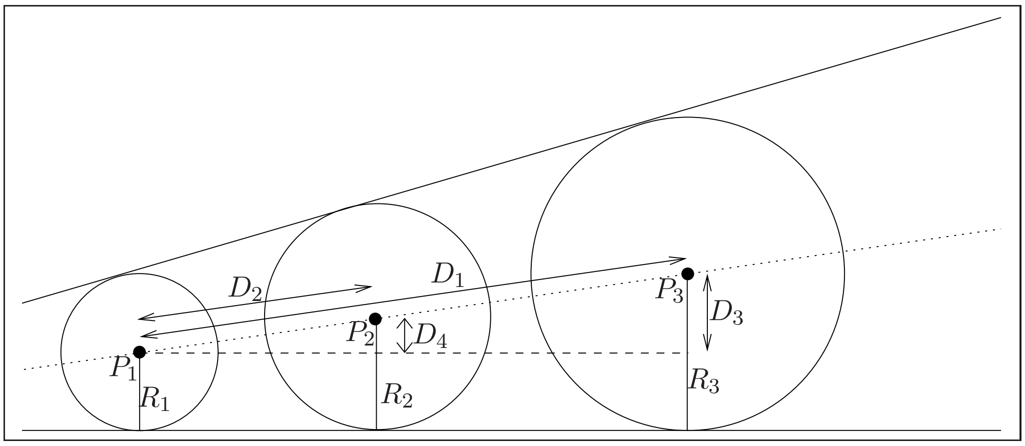


Figure 5.10: An example of how the radii of maximal inscribed circles can be calculated through similar triangles. In the example, P_1 and P_3 are the points marked in VRONI, with radii R_1 and R_3 respectively. The distance between the points along the Medial Axis is marked as D_1, whereas the difference between the two is D_3 ($R_3 - R_1$). Using similar triangles and D_2 (the distance between P_1 and P_2 along the Medial Axis), the length D_4 can be calculated, hence R_2 (radius of maximal inscribed circle centred at P_2) can be calculated since $R_1 + D_4 = R_2$.

Instead, only the vertices of the skeleton generated by VRONI are used, and a requirement is added that for an edge to be considered linear, both end vertices of the edge must also be marked as linear. This is a reasonable approximation to use, due to the nature of the Medial Axis and its generation in VRONI. In VRONI, a threshold value is used to split edges into smaller parts to maintain accuracy when representing the Medial Axis graphically; for example, when needing to approximate curved parts by using a series of small straight lines. This threshold is set within the program and is small, to ensure that the length of any given edge is very small. This includes straight edges being broken down further, since these are treated as a form of curve also.

In addition, if there is any variation in a polygon edge, the Medial Axis will have further edges and vertices due to the sensitivity of the approach. Thus, if both vertices of a skeleton edge are linear, there is no major variation in the width of the channel between the two vertices, whereas if one vertex is non-linear, at some point along the edge there is significant variation in the width.

Such an approximation is not without problems. In the case of only one vertex being non-linear, it means two edges are marked as non-linear, when in fact part of these edges could still be linear had each point on the edges been measured for linearity. In general though, because the edges are very short this is not a concern.

Another consideration is what happens with vertices that are not of degree two. For example, in Figure 5.9, all the vertices within the radius will be of degree two. However, in some parts of the skeleton, this will not be the case.

A first case that must be dealt with is that of the end points of the skeleton. These can occur inland (where the skeleton meets the land boundary), and where the skeleton meets the sea. In the case of the inland ends (where the skeleton meets the boundary), the radius at the end vertices will be zero, and hence would generate errors in the calculation. Edges that are touching the boundary are therefore marked as non-linear, rather than cause an error when calculating.

The other possible end of the skeleton would be where the inland water network joins the sea, and thus where the Medial Axis was cut off as it projects outwards. In Section 4.4, it was noted that the Medial Axis would in fact extend beyond the inland water into the sea, hence a cutoff point was selected where from which the Medial Axis was removed. However, as shown in Figure 5.11, this means some points that are near to this cutoff will have maximal inscribed circles that extend beyond this cutoff, for example point $P2$ in Figure 5.11. Because all required radii are not available, it is not possible to accurately determine if the point is linear, hence all points whose associated maximal inscribed circle extends beyond the cutoff point of the skeleton are marked as non-linear.

The case of points that are of degree three or more requires more thought. An example of this is shown in Figure 5.12, where there are three possible paths from N_2 to vertices within the radius of the maximal inscribed circle at N_2. One approach is to treat such junction nodes the same as nodes of degree one; thus the linearity definition would now require that all points within the radius are of degree two. Another would be to allow the calculation to cross over such junctions, and thus returning to the finding of maximum and minimum radii contained within the radius at that point and determining linearity using the ratios between these.

There are also variations that fall between these, for example requiring that a certain number of the paths are linear, and thus testing each path from the point independently for linearity. Whilst this would not affect the results if there are only two possible paths (the node is of degree two), this may affect the results at junctions. For example, in Figure 5.12, the linearity of edge $N_1 - N_2$ needs to be determined. The maximum radius is always R; thus only the minimum value may change. If all points that fall inside the radius were

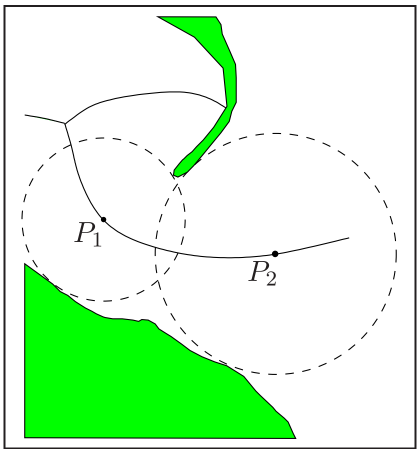


Figure 5.11: An example of problems that can occur where the skeleton meets the sea. P_1 and P_2 are points on the skeleton, with the associated maximal inscribed circles represented by the dotted lines. P_1 does not extend beyond the cutoff point of the skeleton and thus can be measured for linearity, whereas P_2 does extend beyond the cutoff; hence P_2 will be marked as non-linear.

used, then the minimum radius will be R_3, which may be too small in comparison to R for N_2 to be labelled linear. On the other hand, if each path was considered individually, then the set of points on the paths P_1 and P_2 from N_2 may be marked as linear when P_3 is not, due to the lower variation in the radii. If linearity was only required along some of the paths from the vertex instead of all, more points may be marked as linear due to a more relaxed definition.

For this thesis, the linearity function has been written to allow changes to this to be made. This is performed by storing the information of paths from a point in a tree like structure, allowing decisions to be made easily depending on the definition used. For this work, the requirement that all points to be considered were of degree two was implemented (as other methods of handling junctions were implemented as will be discussed later in this chapter), but it would be simple to change for other approaches.

This is not the only way of marking linearity, and may not generate the exact results that would be preferred: First, because of the presence of end nodes with radii of zero, there will always be non-linear sections at the sources of rivers. This can be rectified at the grounding level, when appropriate predicates can be introduced to identify such features (Third et al., 2007). Second, there may be sections that are marked as non-linear that correspond to small gaps or bends. A method of handling these is discussed in Section 5.3.3. Another problem may be that parts that would not be expected to be marked as linear are marked as such. An example of this is shown in Figure 5.13.

The example shown does not initially appear like it should be marked linear, as aside

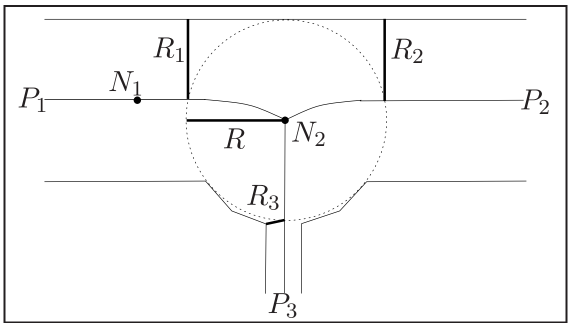


Figure 5.12: An example of how junctions can affect the linearity measurement. Here, there are three possible paths from N_2, labelled $P_1 - P_3$, and we want to determine the linearity of edge $N1 - N2$. For all directions, the maximum is R, but for each path there is a different minimum; on path P_1 this is R_1, on P_2 it is R_2 and on P_3 it is R_3. Thus the measurement is affected by which paths are included in the calculation.

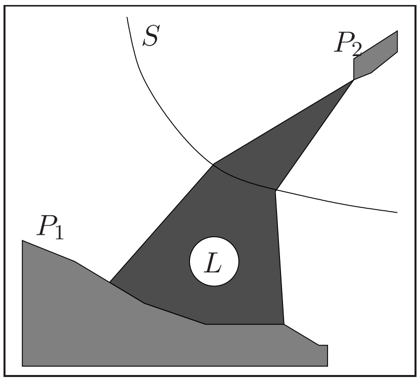


Figure 5.13: An example of deficiencies in the linearity measurement. The figure is an approximation of results generated from the topographic data sets used in testing the system. Here, S represents the simplified skeleton, P_1 and P_2 represent the input polygons and L is the region marked linear to some given standpoint.

from P_2, there are no edges along the top, and P_2 contrasts greatly with *P1*. However, if the simplified skeleton marked S is investigated, it is noticed that it curves around P_2. Thus, if we were in a boat and wished to travel down the centreline of the river (keeping opposing banks equidistant from us), we would follow S and curve around P_2, whilst staying the same distance from P_1. Thus under some thresholds, a region such as L will be marked as linear.

Figure 5.13 highlights the differences when measuring linearity from the centre outward as opposed to considering linearity with respect to the edges. Thus, instead of measuring the variation in the Medial Axis looking *outward*, an alternative is to measure this variation by traversing the edges looking *inward*. A refinement therefore is to require that the regions are linear with respect to the edges as well as the centre, for example by considering the curvature of the edges (Mallenby, 2007). The approach that was implemented here was to determine the lengths of the sides generated by the previous linearity stage, then to compare with the associated centre-line. If the variation in these lengths was significant (as would be the case in Figure 5.13), then the region was marked as non-linear with respect to the edges, despite being linear with respect to the centre. This approach could be refined further to ensure that it could be implemented independently of the other linearity marking approach discussed previously.

Modification of the linearity calculations may seem ‘ad hoc’, as the linearity with respect to edges approach arose from problems with the initial approach to determining linear sections. However, the notion of linearity in relation to river networks is in itself vague; if there existed a precise measurement then it would not be considered as such. Therefore, there will be differences that arise due to different measurements, and it is instead preferable to provide the user with options so that they decide for themselves what definition they wish to use.

5.3.2 Expansive

The linearity segmentation performed in Section 5.3.1 results in every skeleton edge being marked as either ‘linear’ or ‘not linear’, with connected sets of ‘linear’ skeleton edges used to generate ‘linear’ regions. This process can be repeated with connected sets of ‘not linear’ skeleton edges, to generate the regions that occur between linear regions. The term ‘expansive’ was used to reflect that the regions in question are not long, narrow stretches, but instead are more bulbous.

Whilst linearity is a measure of low variation in width, ‘expansiveness’ is a measure of high variation in width, and expansive regions correspond to the regions where the variation in widths is above a given threshold. Thus, when performing the linearity segmentation, the data will be seperated into two sets of regions; the set of polygons that correspond to linear regions and the set of polygons that correspond to expansive regions. The spatial sum of the two sets will always equal the input data set, since all skeleton edges are marked as either ‘linear’ or ‘expansive’ (not linear). In addition, changing the threshold used to determine linearity will impact upon the regions marked as expansive, since more skeleton edges being marked as linear will result in fewer skeleton edges being marked as expansive, and similarly if fewer edges are marked as linear then more are marked as expansive.

Identifying these ‘expansive’ regions allows them to be used within other definitions. For example, if the intention is that linearity will identify rivers or streams, then expansive regions may correspond to more bulbous features such as ponds or lakes. Another possible usage may be to connect linear stretches together, since the expansive regions can be referred to in the definitions of other features.

5.3.3 Interstretch

As already noted in Section 5.3.1, attempts at marking linear sections does not always mark all the regions as linear that would be preferred. In particular, ‘gaps’ may occur due to minor fluctuations in the data, where a gap is a small expansive region (as discussed in Section 5.3.2) separating two linear stretches, and upon examination it would seem that all three parts should in fact be a single linear stretch. These gaps can occur due minor fluctuations within the data, for instance bulges at the edges, sharp corners and small ‘spikes’ of land protruding from the edge. An attempt to rectify this would be to use different linearity measurements or to relax the threshold value used, but relaxing the threshold too much may mean it is too general. Similarly, modification to the linearity measurement would not remove gaps either; thus, despite experimentation to find such a compromise, it was realised there will always be ‘gaps’ in the results of linearity segmentation.

An example of such ‘gaps’ that can occur is shown in Figure 5.14, whereby due to a slight bulging in the channel, there are two separate linear stretches marked L_1 and L_2 separated by the expansive region P, as opposed to it being a single linear stretch. Such a gap could be rectified by a more relaxed linearity threshold, but this in turn would affect all other parts of the data, and thus other parts could be marked as linear (or larger regions), when the preferred marking of such regions would be non-linear. Thus, local deviations within the measurement need to be tolerated, since a global change would prove too general.

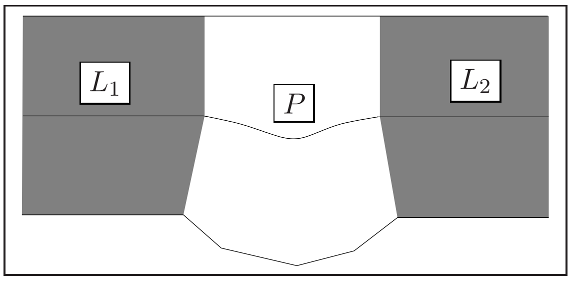


Figure 5.14: An example of the sort of ‘gaps’ that can occur during linearity measurement. Regions L_1 and L_2 are marked as linear, since the width along the skeleton does not vary. However, the region marked P is marked as expansive, as the width expands due to the bulging at one of the boundaries. A more relaxed linearity definition may rectify this, but this may result in too much being marked linear elsewhere.

What this highlights is that with a vague concept such as linearity, there are almost certainly always going to exist discrepancies in the segmentation. This is because the notion of linearity here stems from the abstract notion of a river being an approximation of a line, and thus variation is expected to occur in a linear fashion. However, when working with actual data there will always be more variation and additional features that occur that go against this abstract notion.

Therefore, I feel that artificial features to fill in such gaps will always be required to some extent, and that this is something humans do within our minds when we look at such data (Third et al., 2007). If a user was asked to perform the task manually, it is likely they would ignore some local extremes and fill in gaps without realising. Similar examples exist in many other areas where localised errors may be tolerated; for example, in image recognition there may be small errors and imperfections within the dataset, thus mathematical morphology operations such as dilation may be used to 'fill in' such errors. My approach does not remove vagueness entirely, but does offer more control over how the vagueness is handled, and approaches a system more akin to how a user would segment the data manually if asked, using a global value for linearity but with the addition of a second threshold to allow for localised deviations from this.

Identifying and filling in such 'gaps' through the identification of artificial features has been discussed previously (Mallenby, 2007). To avoid confusion, the term *interstretch* was introduced (Mallenby and Bennett, 2007; Third et al., 2007), to show this is an artificial feature as opposed to an already existing natural feature. *Interstretch* was defined as being a maximal self-connected non-linear region that is connected to two linear stretches, such that all parts of that region are close to both linear stretches. It follows from Section 5.3.2 that *interstretches* must be either tangential proper parts of or equal to an expansive region between two linear stretches. To identify *interstretches*, another vague predicate named 'close-to' was introduced, which would again be calculated within from the data (much as linearity was previously). Once this segmentation was completed, the predicate was fed into the *interstretch* definition.

To determine 'close-to' regions, the following steps are performed:

1. Find shared nodes: Obtain the skeleton of the expansive polygon, and find the two nodes that are shared with the two linear stretches being considered. These will be two of the endpoints of the expansive polygon[4].

2. Find all reachable nodes: Using the graph structure of the skeleton, find all nodes that can be reached from a shared node within a specified distance (designated via

[4]If an expansive polygon occurs at a junction of three or more channels, there may be several such points, one for each connected linear stretch. These are considered in pairs, since an *interstretch* is a 'gap' between two stretches.

a threshold). Repeat for the other shared node, resulting in two subgraphs. Store each of these graphs as a set of nodes.

3. Find all nodes close to both: Find the set intersection of the two sets of nodes, hence the set of nodes that are within a specified distance of both shared nodes. Return the subgraph of this resultant set, where for every pair of nodes N_1 and N_2 in the set, the edge $N_1 - N_2$ is added iff it is in the original skeleton of the expansive polygon.

4. Generate polygon: Using the previously described approaches, generate a polygon from this final skeleton, attaching the label 'close-to'. These polygons can then be used in definitions such as *interstretch*.

The distance used to determine 'close-to' is thus a threshold used to determine the boundary for a particular standpoint, and could either be fixed globally or defined to be related to the width (similar to the linear measurements discussed in Section 5.3.1. Previously, the width at the mid-point was compared against the length, and if this was below a threshold the region marked as an *interstretch* (Mallenby, 2007). Here, the threshold is instead related to an actual distance, whereby if a region is 'close-to' another region, it means the distance between the two is below a specified threshold.

An example of this is shown in Figure 5.15. The example is of three polygons; two linear polygons P_1 and P_2, with P_3 representing the expansive polygon connected to both. The underlying skeleton is also shown, with the shared nodes marked as N_1 and N_2. When given a threshold value, the graph contained within P_3 is considered, and a path from N_1 is found, such that all points on the path can be reached by traversing the graph in a distance less than or equal to the threshold. This is repeated with N_2, generating two sets of nodes. By finding the intersection of these two sets, the set of nodes that are reachable from N_1 and N_2 with a distance less than or equal to the closeness threshold can also be found.

In Figure 5.15a, the threshold used is quite small, hence the paths generated (C_1 and C_2) only extend a small distance from the shared nodes. The intersection of the nodes of these paths would be empty, thus no new polygon would be generated. With Figure 5.15b however, the two paths do converge, as in this example the two paths are equal. This may not always be the case, and the intersection may remove some nodes from the results. Because the intersection of the two paths leaves a connected graph that includes N_1 and N_2, a polygon can be generated to represent the region 'close-to' both linear polygons, represented by the lightly shaded region within P_3. If the threshold had been sufficiently large, then all of P_3 would have been marked as close to both.

By using this approach larger non-linear regions can be segmented into smaller *interstretches*, as not all of a region may be suitable to be used to join two linear stretches.

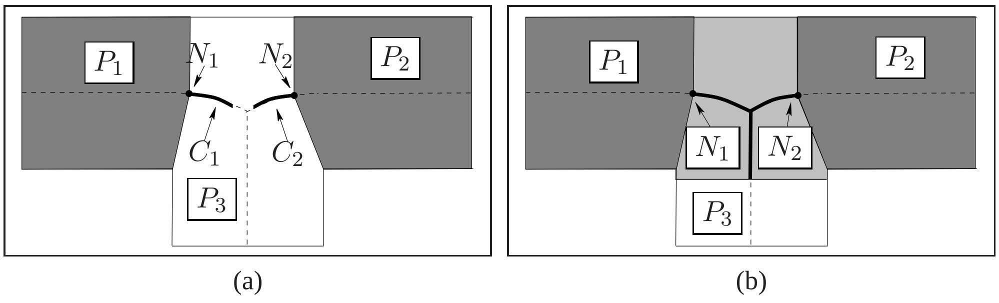


Figure 5.15: An example of how varying the threshold for closeness will impact the results. P_1 and P_2 are linear polygons, with P_3 the expansive polygon connected to both. The dotted line is the skeleton of the dataset, with nodes N_1 and N_2 representing shared nodes. The thick line represents the points that are within the threshold distance of the shared nodes.

An example of this is shown in Figure 5.16. Here, the linearity segmentation has been interrupted due to the bulbous feature underneath the stretch, thus forming a gap. To join the two stretches together, therefore, a way is needed of finding a region to fill in the gap that does not necessarily include all of the bulbous region, since the intention may be to mark this region as something different. The proposed ‘close-to’ attribute would therefore achieve this, allowing us to determine how much of this bulbous feature to include within the measurement.

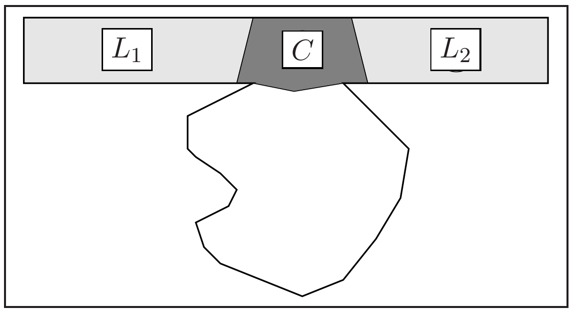


Figure 5.16: An example of a situation where it may be desirable to mark the gap between linear segments. Here, L_1 and L_2 are linear regions, separated due to the bulbous feature. The region C is a region that is ‘close-to’ both regions, thus bridging the gap.

This approach could be adapted to determine ‘closeness’ in other parts of the domain. For example, a potential definition of estuary would be a region that is close to the sea; thus, the measure used above could be used to determine a region that is close to and connected to the sea.

5.3.4 Island and Island-Water

An island is generally agreed to be a region of land entirely surrounded by water, and thus is easy to identify in the data. However, the presence of islands has an impact upon the classification, namely determining the number of channels. This is apparent from the simplified skeleton produced from the Medial Axis; when an island is present in the inland water network, the skeleton will produce a loop around the island. This in turn produces problems for the linearity marking phase, as the island will produce a dramatic difference in the widths.

An example of this is shown in Figure 5.17, where the difference in the generated skeleton with islands (Figure 5.17a) and without islands (Figure 5.17b) is clear. Thus, instead of having a single linear section, the result may instead be two separate linear stretches on each side of the island. The problem, therefore, is determining whether to classify the stretches as separate or as a single channel that would have been marked as linear had an island not been present in the dataset.

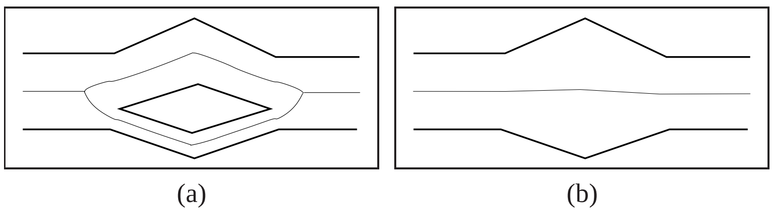


Figure 5.17: An example of the impact the presence of islands can have to the Medial Axis and associated skeleton. Figure 5.17a left shows the simplified skeleton that results in a stretch with an island in, whereas Figure 5.17b shows how this stretch would look if the island was removed.

A method of comparing the island against the surrounding area is therefore needed, to decide if the island will be regarded as being within a single channel or surrounded by multiple channels. The island can be determined using logical queries, as can the water regions directly touching it. From these, an additional measurement is therefore needed to determine whether to consider the surrounding water as separate stretches, or as a single ‘island-water’ region.

Again, this is an artificial term[5] introduced to signify ‘island-water’ is something added to rectify errors that may be clear to humans but not to a computer; even small land artefacts in a water channel will have an impact upon the skeleton generated, whereas a

[5]This may not be the most appropriate name for this feature, as ‘island-water’ could also be taken to mean water contained within an island. However, determining appropriate labels for artificial features can be difficult, since ideally the name should still be meaningful to some degree.

human would be able to identify that these should be ignored.

To determine the threshold between ‘island-water’ and separate channels, the sizes of the respective polygons are compared. In GEOLOG, regions with holes are stored as a set of polygons; the first is the ‘outer’ polygon (which for polygons with no holes would be the only polygon in the set), with any remaining polygons in the set representing holes in the polygon. Thus, ‘island-water’ regions will consist of an outer polygon and a series of holes, where each hole represents an island.

The area of the outer polygon is found, as is the sum of the areas of the holes. These are then compared as a ratio $\frac{outer}{holes}$, where $outer$ is the area of the outer polygon and $holes$ is the sum of the areas of the holes in the polygon. If this ratio is greater than a specified threshold value, the island is designated as small enough to be considered ‘insignificant’ within the channel, whereas if the ratio is lower than the threshold, the island is too large in comparison to the ‘island-water’ region for it to be considered a single region and hence the region is treated as separate channels.

Implementing such a ratio may not be straightforward. For example, where there are a series of islands close together, in the implementation described above they would be treated as a single set. However, in some instances it would be preferable to view some of the islands in this set separate to the others, such as when one is significantly larger. This may be problematic to implement as it requires nested definitions relating the sizes of the islands as well as the ‘island-water’ region.

For the purposes of this thesis, the vague threshold part of the ‘island-water’ predicate will not be considered, due to the implication issues detailed previously. Further work would be required to determine how to implement the predicate with a vague threshold attached; hence, in this thesis only the initial marking of ‘island-water’ regions based upon their connection to islands will be used.

5.3.5 Conclusion

A series of predicates with vague parameters have now been derived that can be used to generate standpoints, in which the data is segmented into smaller regions, representing particular features. These predicates are listed in Table 5.1.

The basic initial measurement is one of linearity, from which further vague predicates were developed to classify further regions. These are not the only thresholds that could be used however, and more may be required to handle different water features. For example, these thresholds are based upon two-dimensional topographical data, but to extend to three-dimensional data it is likely that different considerations of how to mark linear parts

Table 5.1: The list of predicates that have been identified to segment the data into regions representing features.

Predicates
linear
expansive
interstretch
island
island-water

or other such basic attributes would be required. Another example is the classification of an estuary; such a feature may be dependent upon both topographical and salinity data, thus a standpoint may be dependent upon thresholds on both sets of data to make decisions.

Finally, there will be additional thresholds in a particular standpoint applied to classify the features further, which may correspond to different forms of vagueness as discussed by Bennett (2005). For example, the linearity predicate is a form of *sorites* vagueness, which in turn identifies a region (location vagueness). However, it is still vague whether this region corresponds to a river or a stream, which is an example of classification vagueness. For this form of vagueness, measurements such as length, width or area may be used, to determine the threshold between the two types of feature.

The aim of this section therefore, is to illustrate how a formalisation of the conceptualisation of vague predicates can be developed and implemented in order to identify features to use in segmentation and classification. These features can then form the basis of the logical queries and definitions discussed in the next chapter.

5.4 Calculating Spatial Relations

The final required stage is to determine the spatial relations between the polygons. For spatial relations, the Region Connection Calculus (RCC) (Randell et al., 1992) will be used, specifically the RCC-8 relations previously discussed in Section 2.5.1. Reasoning with RCC is generally undecidable unless restrictions are made, for example RCC was found to be decidable for certain binary relations interpreted over arbitrary topological spaces by Bennett (1994), whereas reasoning about the disjunction of RCC-8 relations was been shown to be NP-complete by Renz and Nebel (1999).

In order to reason specifically about the relations between polygons an additional step from quantitative to qualitative is required. Logically, this makes things simpler as a single concrete model is established to base the RCC-8 relations upon. However, geometrically this adds complexity to the problem, as all possible intersections and shared edges now need to be calculated, as well as all points that are either on the boundary or inside the opposing polygon. Therefore the calculations required at each stage should be reduced in order to speed up the reasoning process.

Previous work on an abstract level was done by Bennett et al. (1998), which illustrated the process could be broken down into a hierarchical tree, reducing the calculations required. The intention here is to use information that can be retrieved quickly to break down the possible relations that may hold between two polygons, to ensure that the need for more intensive computations such as line intersections and point locations is reduced. A similar approach using the 9-intersection calculus was discussed by Rodríguez et al. (2003), where again the aim was to use reduce the computations.

5.4.1 Reducing Down the Set of Possible RCC-8 Relations

An initial premise is that any of the eight RCC-8 relations may hold between the two polygons in question. One option therefore would be to simply test for all intersections and whether any points of each polygon are inside the opposing polygon, in order to determine the RCC-8 relation between the two polygons. However, this is clearly inefficient, since in the case of polygons whose relation is DC, a large amount of calculations will be performed that will return no results. A first stage therefore, is to reduce the set of potential relations that can occur between two polygons.

A first approximation is to compare the bounding boxes of each polygon, here defined as the smallest axis-aligned rectangle that can entirely contain its polygon. Thus, if the bounding boxes of two polygons do not touch at all, the RCC-8 relation for the polygons is DC. This significantly reduces the initial calculations, and allows us to eliminate relations that are not possible.

Previously, this was performed using the relations derived from Allen's interval Algebra (Allen, 1983) (hereby referred to as Allen relations), where, for the X-Axis and Y-Axis respectively, the Allen relation that holds between the two bounding boxes along that axis was calculated and used to aid the calculation of the RCC-8 relation between the two polygons (Mallenby and Bennett, 2007). However, the Allen relations were originally intended to be used for time intervals, and hence take account of direction from past to future, which is irrelevant to the spatial calculations performed here. An improvement, therefore, is to use RCC-8 with each axis, which does not take account of direction, and thus is simpler to work with.

If the X-Axis and Y-Axis are treated as separate dimensions, the RCC-8 relations between two bounding boxes can be determined in each axis. The resulting pair of RCC-8 relations can then be compared to determine what possible RCC-8 relations between the two polygons these allow. Determining the RCC-8 relations is straightforward; for a given axis the minimum and maximum values of the two polygons are found and represented as two lines. These can then be sorted as numbers to determine what RCC-8 relation they correspond to. This is repeated for the other axis, hence each operation is only working in a single dimension.

This results in a pair of RCC-8 relations, which in turn represent a set of possible RCC-8 relations that can hold between the two polygons, as shown in Figure 5.18. In these examples, it can quickly be determined that the first example shows two disconnected polygons, thus no further computation is required. With the other two examples a set of possible relations remains. However, these reduced sets can be used to determine the most effective approaches to take next, thus tailoring deductions to each pair of polygons.

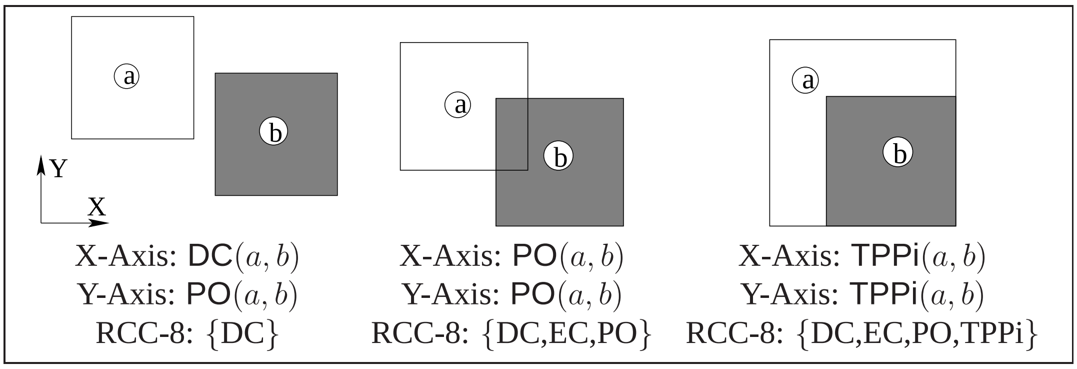

a
b
Y
X
X-Axis: $\mathsf{DC}(a,b)$
Y-Axis: $\mathsf{PO}(a,b)$
RCC-8: $\{$DC$\}$
a
b
X-Axis: $\mathsf{PO}(a,b)$
Y-Axis: $\mathsf{PO}(a,b)$
RCC-8: $\{$DC,EC,PO$\}$
a
b
X-Axis: $\mathsf{TPPi}(a,b)$
Y-Axis: $\mathsf{TPPi}(a,b)$
RCC-8: $\{$DC,EC,PO,TPPi$\}$

Figure 5.18: Examples of how the bounding boxes of two polygons a and b may be related spatially, and what possible RCC relations these represent. The RCC-8 relations were obtained for the X- and Y-Axes, then compared to see what the set of possible RCC-8 relations are for the polygons.

Since there are 8 RCC-8 relations, it follows that in total there are $8 \times 8 = 64$ possible combinations when comparing in two dimensions. However, most of these result in the same possibilities of RCC-8 relations, and in fact there are only 14 different possible combinations, listed in Table 5.2. A method of reducing the possible RCC-8 relations quickly has now been determined; for example it can be quickly determined in some cases that two polygons are definitely disconnected.

Such an approach is similar to the Rectangle Algebra (RA) (Güsgen, 1989; Balbiani et al., 1998, 1999), where the Allen relations were used in two dimensions to generate a series of possible relations. However, whilst in RA the 169 possible relations are considered separate so as to provide other possible information such as directional relations,

Table 5.2: The possible relations as a result of comparing the Allen relations between the X- and Y-axis. Starred relations also have versions replacing TPP/NTPP with TPPi/NTTPi respectively.

Possible RCC combinations from previous stage
DC
DC, EC
DC, EC, PO
DC, EC, PO, TPP *
EC, PO
EC, PO, TPP, NTPP *
EC, PO, TPP, TPPi, EQ
PO
PO, TPP *
PO, TPP, NTPP *

the combinations are used here to derive topological relations only. The importance of bounding boxes to reducing the set of possible spatial relations has also been discussed by Clementini et al. (1994), where similar results to those in Table 5.2 were determined for Egenhofer's 9-intersection calculus.

The effectiveness of this in implementation is clearly related to the spatial relations between the polygons in the input set. For example, a set of polygons that were spread out enough such that no two bounding boxes touched would be easily processed as the approach would mark all polygons as DC quickly. Similarly, a set of polygons where all bounding boxes touched each other would render this stage useless, since all pairs of polygons would require further calculation to determine the RCC-8 relation.

To test the approach in a manner that it may be implemented in the intended system, a series of test files were developed. These test files were based upon the topographical datasets used in later chapters. For each dataset, an initial set of inland water polygons were segmented (in addition to the land and sea polygons). This initial set was then split into smaller parts, each time generating a larger set of polygons with which to test the approach. The results of these tests are shown in Figure 5.19, where the number of polygons represents the number in the test set and the number of pairs of polygons whose RCC-8 relation could be determined using the bounding box alone is given as a percentage of the total number of pairs of polygons needed to be tested for a given input set.

Although the results in Figure 5.19 are encouraging, it is worth noting that the manner in which the test files are generated will in part help this. For example, consider the set of polygons p_1, p_2, p_3, where each polygon is a rectangle 2 units wide by 1 unit tall, and

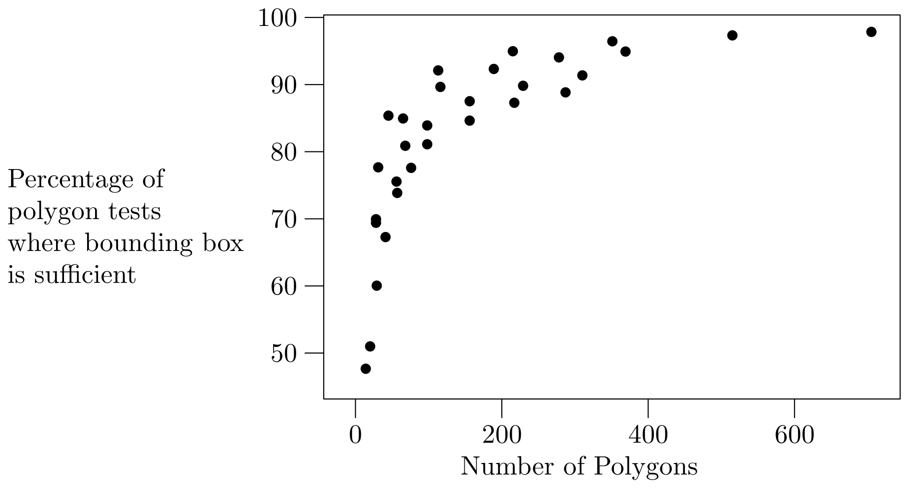


Figure 5.19: Chart of the percentage of polygon pairs for which the bounding box approach is sufficient to determine the RCC-8 relation. The X-axis represents the number of polygons in the test set; thus, if there are n polygons then there are $\frac{n!}{2!(n-2)!}$ or ${}_nC_2$ pairs of polygons to compare. The Y-axis shows the percentage of these pairs where the bounding box is sufficient to determine the RCC-8 relation that hold between the two polygons.

polygons are connected by the tall edge, as shown in Figure 5.20. In the initial state, there are 3 relations to calculate, 1 of which can be determined from the bounding boxes alone, thus $\frac{1}{3} \approx 33\%$ can be determined using bounding boxes alone.

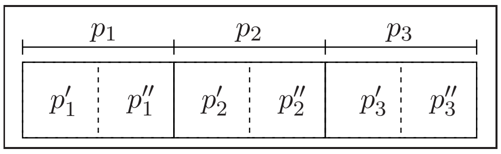


Figure 5.20: An example of a drawback of the approach used to generate the test set of polygons. The thick lines represent the initial polygons, with the dotted line showing how these would be split to generate a further test set.

For the second set, each polygon is split into two squares, e.g. p_1 is split into p_1', p_1''. These are combined with the original 3 to produce a set of 9 polygons, thus 36 possible pairs. If two polygons can be determined to be disconnected using the bounding boxes, then it follows the same holds for combinations involving them and their parts. Thus, the bounding box will be sufficient to determine p_1 is disconnected from p_3 and the two new polygons generated from splitting p_3 in half. In addition though, it is clear from Figure 5.20 that several other pairs will be found to be disconnected from the bounding box, essentially wherever there is now at least one polygon between the pair. Thus there are now $\frac{19}{36} \approx 53\%$ of relations that can be determined using bounding boxes alone. Thus, using this approach to generate test sets would be expected have a larger percentage of

Table 5.3: The definitions of the RCC-8 relations used in the system. Only 6 are shown here, as TPPi and NTPPi are merely the inverses of TPP and NTPP respectively.

RCC	Definition in system
DC(X, Y)	There is no intersection between X and Y, and no point of X is inside or on the boundary of Y (and vice versa)
EC(X, Y)	There exists a point that is on the boundary of both X and Y, but there are no points of X inside Y (or vice versa)
PO(X, Y)	There exists either at least one intersection between X and Y, and there are points that are inside one polygon but outside the other
TPP(X, Y)	All points of X are either inside or on the boundary of Y
NTPP(X, Y)	All points of X are inside Y
EQ(X, Y)	All points of X and Y lie on the boundary of each other

polygons each time whose relations can be determined using only the bounding boxes, as the number of polygons is increased by splitting existing polygons. However, the test sets are still suitable for testing the effectiveness of the approach for the intended usage, as in the intended system there will often be a need to segment polygons into smaller parts; thus such an approach would need to prove effective in eliminating pairs of polygons from further testing. The test set is representative of the kind of dataset that the system will need to handle, hence it is a reasonable method of developing the test set.

Calculations for RCC-8 relations are based upon the locations of the corners of the polygons to be compared, and whether they are inside, outside or on the boundary of the other polygon in question. In addition, all points of intersection between the two polygons that are not already a corner point of a polygon are added to the test set. Table 5.3 shows the RCC-8 relations defined in terms of the tests that are required in order to make decisions as to the RCC relation between two regions.

This is similar to approaches proposed by Haarslev et al. (1998) and Grütter and Bauer-Messmer (2007), where the spatial domain was also restricted to polygons as opposed to arbitrary points due to the existence of efficient algorithms to handle polygons. This is important as it must be clear what a region means in order to determine the domain of quantification when working with logic queries at a later stage.

5.4.2 Calculating All Intersections Between Polygon Edges

The intersections of the edges of two polygons has been studied extensively, in an attempt to improve upon the brute force approach of comparing all lines against all others. More efficient methods are based upon the sweep line approach (Bentley and Ottmann, 1979); whereas the brute force approach has a complexity of $O(n^2)$ the sweep line approach has a complexity of $O((n+k)\log n)$ where k is the number of intersecting pairs of segments in the results. Such an algorithm is therefore sensitive with regards to the output, with greater numbers of intersections increasing the total time required. This is of course dependent upon an efficient implementation of such an algorithm, as naive implementations may not achieve this. Also, Žalik (2000) suggested that alternative algorithms may perform just as well in actual implementation even if they are not theoretically as efficient (such as the binary tree based algorithm suggested by Žalik (2000)).

The aim of such algorithms is to reduce the comparisons between lines. For this system, three approaches to the problem are considered: The first is to use a sweep line approach for the two polygons together. The second is to use the Allen relations approach (Mallenby and Bennett, 2007), as used in the previous section to determine which lines to test for intersection. Finally, a hybrid of the two is considered, whereby the bounding boxes are used to determine a smaller set of lines to test, then a sweep line approach is used upon this set.

5.4.2.1 Sweepline Approach

The Bentley-Ottmann algorithm (Bentley and Ottmann, 1979), commonly referred to as the sweep line algorithm, is the standard algorithm used to determine line intersections. It works by sweeping a line over the line segments, and storing intersection events as they occur. To implement in this system, the lines from both polygons could be combined into a single set, passing the sweepline over them. The result would be a series of intersections, some of which may occur on only one polygon; as shared points are of interest also, two line segments sharing an end point would be considered an intersection. Thus, 'self intersections' would need to be filtered out, which could easily be done given the line segments that make up a polygon are already stored.

5.4.2.2 Bounding Box Approach

This is an improvement on the approach proposed previously (Mallenby and Bennett, 2007), where it was noted that in order for a line to intersect with a polygon, that line must touch, intersect or be inside the bounding box of that polygon. The Allen relation approach is again used to determine if a line falls inside a bounding box and thus eliminate quickly lines which could not cause an intersection. The use of bounding boxes in polygon intersection algorithms has been proposed previously, for example by Barton and Buchanan (1980), as it ensures that only regions where intersections could occur are considered.

The effectiveness of the approach is directly linked to the total number of intersections between two given polygons, since a high proportion of intersections reduces the effectiveness of the bounding box removal stage. However, heuristics to measure the effectiveness of such approaches were proposed by Suri et al. (1999) and Zhou and Suri (1999) . In particular, it is suggested that the degradation in performance is smooth, and that results do suggest that breaking complex objects into smaller, simpler parts is an effective approach.

Previously, the approach was to choose the polygon which had the fewer edges, then compare each edge of that polygon against the bounding box of the opposing polygon (Mallenby and Bennett, 2007). If a particular edge touched, overlapped or was inside the opposing polygon's bounding box, then it was tested against all the edges of the opposing polygon, to determine if any intersections occurred. This test can be performed in a variety of ways, though the approach here was to again compare the Allen relations of both axes for both lines, as this was found to be both simple and effective.

An improvement upon this is to build upon the usage of the bounding boxes, as shown in Figure 5.21. The only intersections that can occur between two polygons must occur in the intersection of the two bounding boxes. If the RCC-8 relation between the bounding boxes was EC, then any shared points would have to touch the shared bounding box edge or point. Thus given two polygons A and B, the only edges from A that can intersect with B are those that either touch, intersect or are inside the bounding box of B, and similarly for edges of B that intersect A.

To implement this, first the set of edges from polygon A are found that are touching, overlapping or are inside the bounding box of polygon B. This is then repeated with polygon B, finding all edges that are touching, overlapping or inside the bounding box of A. Finally, a pair-wise comparison of the two sets is performed, using the Allen relations based test (Mallenby and Bennett, 2007).

The intention is that this additional stage is computationally simpler than the intersection stage, and thus the additional computation of the sets will be offset by a larger reduction in the time required to calculate the intersections for the reduced set of lines. For example, in Figure 5.21 the set of test edges is significantly smaller than the total combined number of edges contained within polygons A and B; thus, providing the calculation of the set of edges to test is sufficiently quick, a reduction in the time taken to determine intersections would be expected.

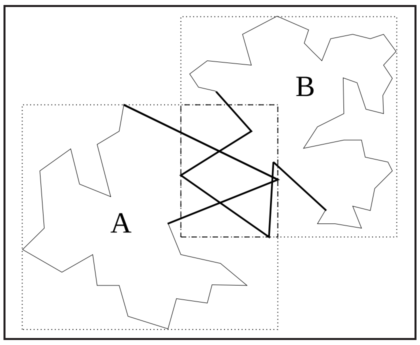


Figure 5.21: The possible intersections between two polygons. If the bounding boxes of the two polygons A and B are compared, the only edges that need to be tested are those that are inside, touch or overlap the heavy dotted area, specifically the region formed by finding the intersection of the two bounding boxes. Thus the only lines from A to test are those that touch, intersect or are inside the bounding box of polygon B, and vice versa.

5.4.2.3 Hybrid Approach

This combines the sweep line approach with the bounding box approach discussed above. As before, the two sets of lines are determined that touch, overlap or fall within the opposing bounding boxes, thus reducing the total set of lines that are required to be tested. Instead of using the Allen relations approach above to determine intersections, however, a sweep line approach is now implemented across the two sets, registering intersections only when they occur between lines from two different sets. In addition, by tracking the lines that have been visited, the algorithm can be ended once all the lines from one polygon have been visited, thus potentially saving further computation time. The intention of this hybrid approach is that combining the two approaches will provide the most efficient overall algorithm.

5.4.2.4 Comparison of Approaches

The first consideration is the time complexity of the algorithms. As already noted, the time complexity of the sweep line approach is $O((n+k)\log n)$, where k is the number of intersections that are found. For the bounding box approaches, the overhead associated with determining the reduced set of lines needs to be determined. A first test therefore, is to determine what lines may intersect based upon the bounding box of the opposing polygon. Determining the bounding box can be performed by finding the minimum and maximum X and Y values for the polygon, which requires $4 \times n$ where n is the number of lines in the set. Thus it requires $O(4n) = O(n)$ time.

To test if a line is touching, inside or intersecting the bounding box, the minimum and maximum X and Y values for the line are calculated, which requires $O(1)$ time, since it is a simple comparison of the two values each time. Next, these maximum line X and Y values are tested to see if they are smaller than the respective minimum bounding box values, and similar for the minimum values being larger than the respective maximum bounding box values, eliminating any lines that are. This may mean that some lines that are not actually touching, intersecting or inside the bounding box are included in the final test set, but performing the test in this manner keeps the computation of this stage down. Each of the four comparisons can be performed in constant time; therefore, to test n lines again requires $O(1) \times O(n) = O(n)$ time.

The overall complexity is $2(O(n) + O(n)) = O(n)$ for the bounding box phase, as this needs to be performed twice. As the intersections phase would be expected to require more than linear time, it follows that the bounding box phase does not add to the overall complexity of the algorithms. The complexity of the intersections stage must therefore be calculated, to determine the overall complexity for the algorithms.

For the original bounding box approach, the Allen relations between two lines were used to decide whether to compute the intersection or not. This means if for two sets of lines n and m, the worse case would be to compare all lines against each other, thus requiring $O(n \times m) = O(n^2)$ time. Thus the overall complexity is the same as the brute force approach, which is unsurprising given the worst case means comparing all lines against each other.

For the hybrid approach, the complexity is the same as the sweep line approach, since as stated above in the worst case the bounding box approach has had no impact and the test set contains all lines remaining in the test sets. Therefore, the overall complexity would again be $O((n + k) \log n)$.

Theoretically therefore, the sweep line approach remains the most efficient of the three algorithms, as in the worst case it would not require the additional $O(n)$ time that the other two algorithms would. However, as discussed by Žalik (2000), the actual run-time is often of greater interest than the time complexity, since the theoretical time complexities are based upon the worst case. In addition, as noted by Suri et al. (1999) and Zhou and Suri (1999), the two polygons in question will usually have few intersections, thus the overhead of using the bounding boxes to eliminate lines from the test set may be negligible in comparison to the gains from testing fewer lines for intersections. The effectiveness of the algorithms therefore needs to be tested; first testing how effective the bounding box is at removing lines from the test set and then the actual run-times of the three algorithms.

The test set once again is the set used in Section 5.4.1, where the results are considered in terms of the number of lines in the pairs of polygons that are tested. First, the effectiveness of the bounding box approach at removing lines from the test set is tested, as shown in Figure 5.22. Here, the reduction is represented as a percentage of the total number of lines for a given pair of polygons. Again, the bounding box approach is effective at reducing the set of lines that need to be tested, though the issues noted in Section 5.4.1 should be considered here also.

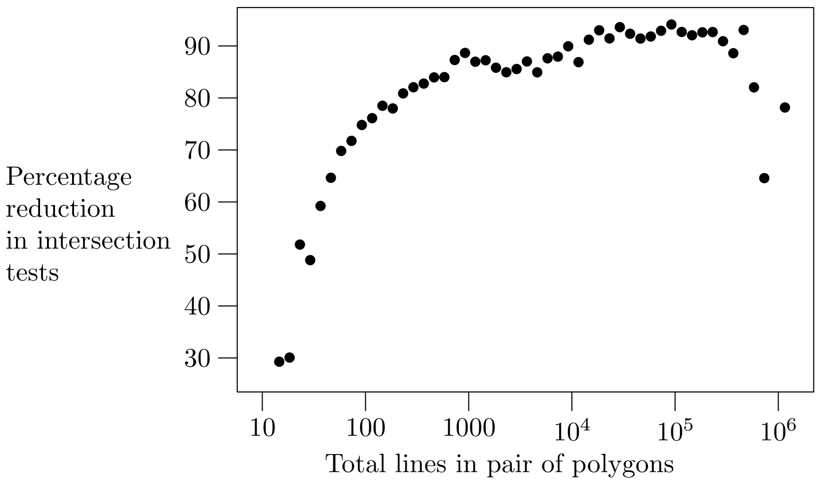


Figure 5.22: Chart of the percentage of line intersection tests that are removed due to bounding box stage. The X-Axis represents the total number of lines in the pairs of polygons, and has been scaled logarithmically. The Y-Axis shows the average percentage of lines that can be removed due to being outside the intersection of the bounding boxes.

Next, which of the approaches proves fastest in implementation needs to be determined; the sweep line approach, the original bounding box based approach or the hybrid of the two. In terms of actual intersection tests, the bounding box approaches would be expected to be faster simply because the set of lines to test is significantly reduced as shown in Figure 5.22. However, in order to reach this stage the initial bounding box stage must also be considered, which if performed inefficiently could mean the overall time for the algorithm is slower than simply using the sweep line approach independently.

A comparison of the run-times of the algorithms is shown in Figure 5.23. The results are the average of 5 runs of the test sets previously used. First, both the bounding box based approaches prove to be more efficient than simply using a sweep line approach on the full set. Thus, there is little additional run-time required in order to determine which lines can be ignored by the intersection tests.

Of greater interest though, is the comparison between the two bounding box approaches; the original approach proves faster initially but the time to compute intersections increases at a faster rate than under the hybrid approach, which proves more efficient for the larger sets. This is most likely due to the initial overheads of preparing the sets of lines for a sweep line approach, which are more noticeable for smaller values. This over-

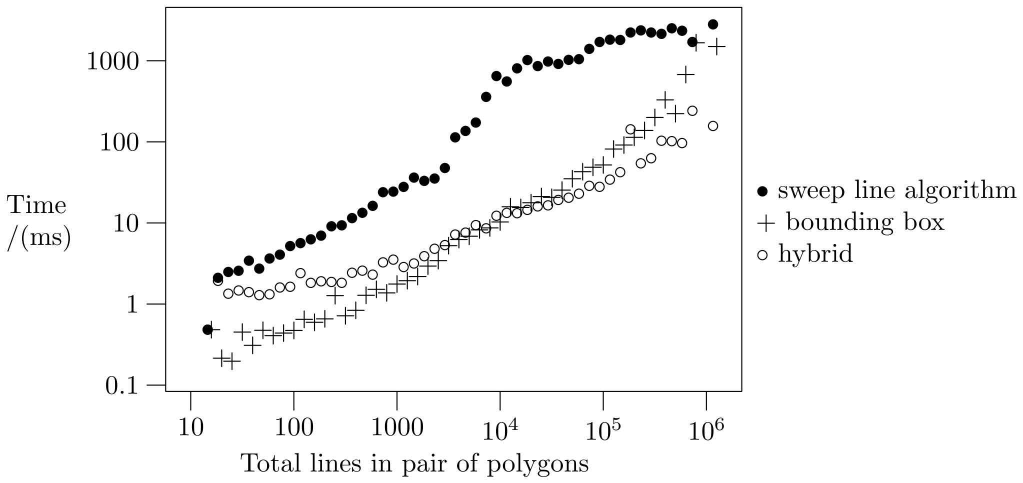


Figure 5.23: Chart of the run-times of the algorithms. The X-Axis represents the total number of lines in the pairs of polygons, while the Y-axis is the run-time in milliseconds. The chart is logarithmically scaled in both axes.

head however proves reasonable for larger line sets, where the hybrid approach is more efficient.

A final comparison to make between the algorithms is the run-time in terms of the number of actual intersections, as shown in Figure 5.24. Whilst Figure 5.23 shows the results in terms of the size of the input set, the time taken to find the intersections is also of interest, since large sets may only have a small number of intersections. The bounding box approaches are again quicker than the sweep line algorithm on its own, though interestingly the performance of the original bounding box approach degrades steeply such that for large numbers of intersections it takes as long as the sweep line algorithm does. The hybrid approach once again is marginally slower for smaller numbers, but the performance degradation is much lower than the original approach, thus proving to be quicker for large numbers of intersections.

In conclusion, the hybrid algorithm seems the most appropriate choice of the three algorithms presented, using information previously calculated to speed up the calculation of all intersects whilst remaining simple to understand. The algorithm could most likely be optimised further, but was fast enough for the purposes of the project, providing simple to understand algorithms that could be applied to a variety of problems. The result of this stage is a set of points of intersection, which may include existing corner points of a polygon. These can be separated into existing and new points using set operations.

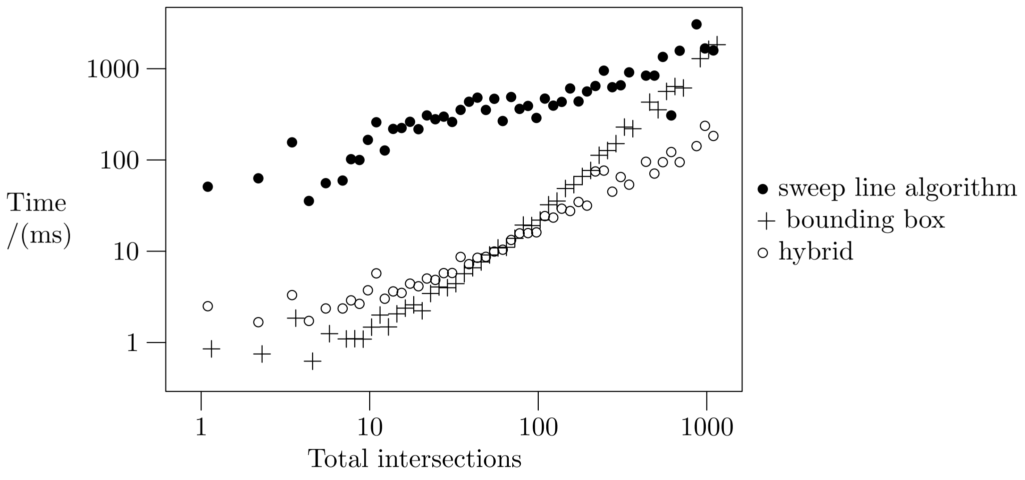


Figure 5.24: Chart of the run-times of the algorithms in terms of the number of intersections found. The X-Axis represents the total number of intersections, while the Y-axis is the run-time in milliseconds to find the intersections. The chart is logarithmically scaled in both axes.

5.4.3 Points Inside Polygons

As with the intersection tests, the aim is to reduce the number of points that are tested to keep computation time down. The bounding boxes are once again used to reduce the set of points to test, since clearly if a point falls outside the bounding box of a polygon then it must be outside the polygon also. This subset is then tested using a ray-crossing based algorithm, that has been discussed extensively in various computational geometry books (Preparata and Shamos, 1985; Foley et al., 1995; Mortenson, 1997; de Berg et al., 2000).

The basic principle of the algorithm is to extend a ray from the point horizontally, and count the number of times the ray intersects polygon edges. If the number of intersections is odd, the point is inside the polygon, else if the number is even it is outside. The number of intersection tests can be reduced by using Allen relations to eliminate lines that could not intersect the projected line. How to handle points that lie on the boundary is often an issue for such algorithms. However, this set of boundary points was previously found in the intersection tests; therefore this set can be used to remove points on the boundary, leaving only points explicitly inside.

5.4.4 Using the Results with RCC

The results of the previous stages is a series of sets of points. Therefore, the RCC relations can be tested for using set operations on these points, as the previously given definitions easily translate into set operations:

First, all the potential RCC relations are found using the Allen relations based approach mentioned earlier. From this stage, decisions can be made based on which tests to do; for example if the results of this stage is the set {DC,EC}, there is only the need to test for at least one intersection to determine if the answer is DC or EC. Implied relations can also be found; if DC is not a member of the list then the two regions are connected, whereas if DC and EC have both been removed then there is at least some part of one region is part of the other.

5.4.5 Conclusion

A system capable of calculating the spatial relations that hold between two polygons has now been defined, bringing capabilities found in GIS to the input data. The reason that existing GIS was not used was because new polygons may be generated when the dataset is segmented, thus spatial relations needed to be determined as they arose. In addition, the development of line and point intersection tests may help with other aspects of the system.

5.5 Summary

This chapter has shown how the data can be segmented to form the basic blocks from which hydrographic features will be generated. It was shown how sub-regions of the data can be generated from the skeleton structure, by determining edges from the stored boundary graphs to combine into polygons. The thresholds that could be used to determine standpoints were discussed. These can be applied to collect attributes that can be used to segment the data into polygons that can then be passed to the next stage for reasoning. Finally, it was shown how the spatial relations between these can be analysed and determined in a manner that is efficient enough for the purposes of the system.

Chapter 6

Grounding an Ontology upon the Data Level

6.1 Introduction

This chapter will discuss how to ground an ontology upon the data level, thus bridging the gap between the two. Section 6.2 will examine what grounding an ontology means for this system, and hence what is required in order to effectively ground an ontology upon the data. Section 6.3 will determine the most effective logic language in which to develop the ontology, by considering the requirements of the intended system. In addition, this section will consider how to handle the queries whilst managing the domain. Section 6.4 will consider how to store the queries in a structured manner. Section 6.5 will discuss the output from this stage and how it can be represented to allow integration with other systems. I have named the system described here, GEOLOG, because it handles geographic data in a logical fashion. Section 6.6 will define predicates and definitions that can be used in GEOLOG to define features of interest. Finally, the chapter is summarised in Section 6.7.

6.2 Ontology Grounding

Jakulin and Mladenić (2005), described the fundamental notion of grounding as follows:

> "The higher-level abstract concepts are grounded in lower-level concrete concepts, which, in turn, are grounded in perceptions."

Grounding an ontology upon data is an attempt to add context to the ontology in terms of the data, by providing a link between the two levels. This process works in both directions, again as noted by Jakulin and Mladenić (2005):

> "A *grounded* concept is one that is both conceptualisable (there is data exemplifying the concept) and groundable (the concept can be recognised from the data)."

However, it could be argued that these two notions should be reversed; a concept is 'conceptualisable' if the concept can be recognised from the data, and is 'groundable' if there is data exemplifying the concept.

Ontologies are often viewed as separate from data, the intention being that an ontology could be used with different data sources, as opposed to being dependent upon a particular data source. However, as has been previously discussed, context is important to making judgements in relation to vagueness, for example, when using the vague term 'large'. By grounding an ontology upon data, the definitions can be 'fleshed out' in terms of the data, thus allowing 'large' to be measured within a particular context. Grounding upon data thus provides a link between symbols and data that may represent those symbols.

The need for a layered approach to grounding ontologies upon data has been discussed previously (Third et al., 2007), with an example of the structure shown previously in Table 2.2. This layered architecture allows the ontology level to be linked to different data sets, without the need to modify each layer directly. For example, suppose the intention was to reason about a feature like 'rivers'. Within a high-level ontology, this may just be represented as a primitive (perhaps with some attached text comments), without actually fleshing out the definition logically. Similarly, at the base level of the data, there may only be a single polygon representing all inland water, or the data may have been segmented into smaller polygons by a user, which may be arbitrary or may be related to features the user was interested in. In order to relate the two layers, therefore, some method of mapping ontologies onto datasets is required. The extent to which such a grounding is performed will impact upon the ability to work with the data and extract features.

A basic mapping would be to simply label the segmented polygons already present in the data with appropriate labels. This is grounding the ontology upon the data, in the sense that it is now specifically stated that a polygon is a particular instance of a feature. However, this approach is clearly limited for several reasons: first, the approach is limited by the segmentation of the data, as sometimes the segmentation will correspond to the user's needs, whereas other times it will not. The segmentation may, in fact, be intended for a different context or usage; for example, it may correspond to legal boundaries as opposed to general feature boundaries, or may correspond to different features than those that are required in a particular usage. Often the available GIS data will just be an arbitrary segmentation, meaning polygons contained within do not correspond to any particular feature.

Another limitation is the lack of justification given for the grounding. This grounding is unsuitable for determining similar features, for instance, or determining the reason why something was labelled as a particular feature. Returning to the previous point of the concept being recognized from the data, this basic form of grounding would only allow us to identify this particular *instance* of a concept, as opposed to identifying the required attributes that determine the concept. It would therefore be preferable that the grounding approach used, adds to the definition of the feature and thus provides justification for the grounding. This is particularly important when handling vagueness, as a judgement may be required as to whether something is 'large' or 'small'; thus, it should be possible to show how such a judgement would affect the classification of 'rivers' versus 'streams', for example.

A first improvement, therefore, is to now expand the grounding layer to allow judgements to be made on such vague predicates as size, as proposed by Bennett et al. (2005). Attributes of the feature are identified and used to perform the grounding:

$$\mathsf{river}(x) \leftrightarrow \mathsf{flowing}(x) \wedge \mathsf{large\_linear}(x) \wedge \mathsf{natural}(x) \tag{6.1}$$

$$\mathsf{stream}(x) \leftrightarrow \mathsf{flowing}(x) \wedge \mathsf{small\_linear}(x) \wedge \mathsf{natural}(x) \tag{6.2}$$

These equivalences are adapted from those proposed by Bennett et al. (2005), where the above predicates were instead listed as a set of attributes related to a particular concept. The primitive definitions of 'river' and 'stream' have now been expanded to now depend upon vague attributes such as 'flow', 'size' and 'linearity' (both $\mathsf{small\_linear}$ and $\mathsf{large\_linear}$ have the property of being 'linear' according to some standpoint). By varying the standpoint parameters, the truth value of these predicates will vary, hence the classification of features will also vary.

The next improvement is regarding the segmentation of the data, or the boundaries that are in place. As discussed by Smith (1995, 2001) and Smith and Varzi (1997), there exist different types of boundaries, referred to as *bona fide* and *fiat* by Smith. The distinction between the two was discussed by Smith (2001):

"Fiat boundaries are boundaries which owe their existence to acts of human

> decision or fiat, to laws or political decrees, or to related human cognitive phenomena. Fiat boundaries are ontologically dependent upon human fiat. Bona fide boundaries are all other boundaries. They are those boundaries which are independent of human fiat.”

The geographical domain can thus be seen to contain both bona fide and fiat boundaries; for example, the North Sea has bona fide boundaries in terms of its shorelines, but fiat boundaries in terms of its boundary with the Atlantic Ocean (Smith and Varzi, 1997; Smith, 2001). Returning to the pre-defined boundaries example for grounding, the use of such rigid, pre-defined segmentations is problematic when no such boundary exists in reality. Indeed, whilst the demarcation in use may be a generally agreed upon boundary between features, it is not, in fact, a concrete bona fide boundary, but instead is fiat and subject to human interpretation.

To improve upon this, the ability to define boundaries as required is preferable. Rather than having data containing pre-defined boundaries, the data would instead be segmented into regions as required, which could then be labelled accordingly. This raises potential problems surrounding the domain and the logical handling of it, which will be discussed in subsequent sections.

In summary, the grounding level is intended as an intermediate step between the ontology and data levels. This allows the user to bridge the gap between the two levels in a cohesive manner, by fleshing out the meaning of primitives in the ontology level, in relation to attributes and regions in the data level. By retaining a separate layer, ontologies can thus be grounded upon different data sets and perspectives, using different grounding layers. For example, a ‘river’ may be a two-dimensional feature when using two-dimensional topographic data, or a three-dimensional feature when using bathymetric data. By changing the grounding level used, either of these can be used to form the same primitive feature of ‘river’.

6.3 Logic Choice and Query Handling

6.3.1 Overview

When developing the ontology, it is important to consider the knowledge representation language and domain required in order to most effectively model concepts. With logic languages, there is a trade-off between expressivity and decidability, both of which are desirable. For example, first order logic is a highly expressive logic language, but it is also potentially undecidable. In order to achieve decidability, restrictions upon the language are required, as is the case with description logic (Baader et al., 2003). Description logics can be viewed as decidable fragments of first order logic (Schild, 1991), hence, if decidability is crucial, it may be preferable to translate first order logic sentences into description logic (if possible, which in general is not the case). On the other hand, if this restriction means desired definitions cannot be represented accurately, then a more expressive language, such as first order logic, should be used.

6.3.2 Logical Language Choice

The most popular language at present for ontologies is OWL[1], which is the language used for the semantic web. OWL comes in different forms, separated in terms of restrictiveness: OWL-Lite is the most restrictive, OWL-Full the least (and thus most expressive), with OWL-DL situated between these. OWL-DL is a derivative of the description logic $\mathcal{SHOIN}(D)$. Editors such as Protégé[2], Swoop (Kalyanpur et al., 2006) and Ontotrack (Liebig and Noppens, 2005) are providing increasingly user-friendly methods of developing ontologies, which coupled with reasoners such as FaCT++ (Tsarkov and Horrocks, 2004), Racer (Haarslev and Möller, 2001) and Pellet (Sirin et al., 2007), mean ontologies can be confirmed to be consistent, as well as infer implicit knowledge from an explicit knowledge base. It is for this reason that OWL-DL is typically used rather than the more expressive OWL-Full.

However, for the problem domain being considered in this thesis, OWL and description logics are unsuitable. The first problem is the handling of numbers: OWL restricts numbers, making the use of standpoints tricky or unclear. This was also noted by Raskin et al. (2004), where examples of how basic scientific concepts can be defined in terms of numeric concepts are given, thus the limited support for this in OWL is a problem.

A more pressing problem, however, is complications that arise due to reduced expressivity. As previously noted, reasoning about spatial regions using RCC is required. Due to its limited expressivity, OWL-DL, at present, cannot handle RCC effectively. Katz and Grau (2005) proposed that by adding reflexive roles to OWL-DL, a translation to RCC would be possible, due to the ability to translate RCC into the modal logic S4 (Bennett, 2000), which, in turn, could then be translated to OWL. The problem of spatial reasoning with description logics is also discussed by Wessel (2002), where it is noted that the ‘proper part’ relation could potentially lead to infinite models.

A more comprehensive analysis of combining spatial relations with the semantic web is discussed by Grütter and Bauer-Messmer (2007). It is noted that approaches which

[1]Web Ontology Language (OWL) Reference: http://www.w3.org/TR/owl-ref/ (Visited, July 2006)
[2]Protégé is an open source ontology editor http://protege.stanford.edu (Visited, April 2007)

require changes to the OWL specification may not be desirable, since this may result in a loss in features favourable to OWL-DL, such as compatibility with the Resource Description Framework (RDF). Instead, it is proposed that a hybrid system may be a better approach, where the spatial relation work is performed in a specialised system.

In conclusion, although OWL-DL is the usual choice for ontologies at present, it does not appear to be suited to spatial problems. This is due to restrictions in place on the expressiveness of the language. Although some of these restrictions may be removed in future revisions to OWL (such as proposed for OWL 1.1 (Cuenca Grau et al., 2006)), the hybrid approach proposed by Grütter and Bauer-Messmer (2007) seems the best solution to the problem. This would allow integration with description logic at a later stage if desired, whilst still allowing spatial reasoning to be handled effectively. Similarly, OWL-Full may allow more expressivity due to being less restrictive, but does not appear to offer any advantages over first order logic in this respect.

The choice of language used for this stage was again Prolog, as proposed for the data input stage in Chapter 4. As noted previously, Prolog is a declarative programming language which stems from formal logic, and hence is suited to handling logical queries. It does this by attempting to find resolution refutation of the negation of the query. Retaining the program in Prolog means cohesion between the different stages is maintained.

The most popular implementations of Prolog are SWI-Prolog[3], which is a free and open source implementation, and SICStus Prolog[4], a commercial implementation. SWI-Prolog has strong links with the semantic web, with libraries available to translate Prolog to XML and OWL fragments, but is also slower than SICStus. It is for this reason that programming was performed primarily in SICStus, as the Prolog engine used proved more efficient for large amounts of data. However, translating the system code to ensure that it works in both implementations is reasonably straightforward. Thus a more limited version of GEOLOG could be constructed for use with SWI-Prolog, whereby the potential size of the data is more limited or pre-computed to speed up calculations.

6.3.3 Matter Types

The approach used in this thesis to interpret matter stems from the approach proposed previously by Bennett (2001c), whereby a series of basic *matter types* are used. How these matter types are to be implemented will now be discussed:

First, the mass nouns that will form the basic matter types used must be determined. For GEOLOG, the three basic mass nouns used are *land*, *sea* and *water*. The reason that

[3]SWI-Prolog homepage: http://www.swi-prolog.org/ (Visited, June 2006)
[4]SICStus Prolog homepage http://www.sics.se/isl/sicstuswww/site/index.html (Visited, June 2006)

sea is referred to as separate to *water*, is that in this case *water* refers to inland water, which is of interest to the problem at hand. For a more complete system, the two would be sub-regions of a larger concept; *water* would refer to any water, of which *sea* and *inland water* are types. For this system, it is sufficient to use *water* to represent inland water.

Because the inland water network can be segmented into new regions, the inheritance of such attributes must be considered. This was discussed by Belussi and Cristani (2000), where different forms of inheritance were proposed depending upon the attribute in question, such as downward and upward inheritance. Here, inheritance relates to the attributes shared by regions that are connected. For example, consider two regions a and b whose RCC relation is $\mathsf{PP}(b, a)$. If some attribute was assigned to a that had downward inheritance, then all regions which are part of a would also inherit that attribute, hence b would also be assigned the attribute. Similarly, if some attribute was assigned to b that had upward inheritance, all regions which b was part of would inherit the attribute, hence a would also be assigned the attribute.

With matter types used in this system, they are noted to have downward inheritance; thus, if a sub-region was defined of a region with matter type m, the sub-region would also inherit the attribute of being matter type m. Thus, all sub-regions of the inland water network will inherit the attribute *water*.

6.3.4 Relevant Domains of Predicates

With first order logic, it must be made clear what the domain of quantification is. For example, when dealing with inland water networks, the sea may need to be taken into consideration, for example, when determining the mouth of the river or estuaries. Thus, when something is referred to as being ‘water’, it should be clear what this means; does it mean any type of water, or does it mean a specific type such as seawater? In addition, is the boundary of the region in question bounded in some way, or can it be any shape?

Another problem that arises, is whether the domain is finite or infinite. As already noted, the intention is to use RCC to express topological relations that hold between regions. However, the standard models of RCC are infinite domains. Typically, the sets of all regular closed (or regular open) subsets of Cartesian space (either two or three dimensional). Real spatial data usually consists of finite sets of polygons, but the domain of quantification in the standard RCC would include not only these polygons but also all possible ways of carving these up into further polygons.

A method of working with the data within a finite domain needs to be developed, which is adequate to characterise the domain to ensure that it is relevant to any given spatial query. As discussed by Pratt and Schoop (1998), the full set of regions contains many regions that are not of interest, such as tiny regions or obscure shapes with convoluted boundaries, thus it would be preferable to work only with the set of regions that are useful or that interesting features could be derived from. For example, if only inland water features are to be considered, the only regions that need to be segmented are inland water regions, and thus it may be sufficient to represent the land as a single polygon.

Thus, the definition of regions is here restricted to polygons, which has previously been proposed by Haarslev et al. (1998) and Grütter and Bauer-Messmer (2007). The domain will consist of polygons generated from the data, with further polygons derived from this polygonal information through geometry based predicates such as linearity as described in Section 5.3.1. With water regions, the domain could vary considerably, depending upon the restrictions used. The size of this potential domain will now be considered.

6.3.4.1 Domain Size

First, the largest possible domain that could feasibly be used is considered here. The largest domain would be to allow any set of points to be classified as a region; thus if there were $n \times n$ points, there could be 2^{n^2} regions. This is based upon an assumption that the data is of finite resolution, where there exists a smallest possible single point depending upon the granularity of the data. If the domain was of infinite resolution (such as spanning all real numbers), there would be no upper bound on the size of the domain.

A domain allowing regions to be points (or combinations of points) is far larger than is needed or can be generated using GEOLOG, but all models that are generated can be inserted into this domain. Thus, any generated models do still form a complete subdomain that could be inserted into the larger, full domain. The upper limit on the number of polygons that can be generated by GEOLOG can now be considered.

For the problem domain, a skeleton representing the overall topology of the input polygon has previously been generated. It is possible to obtain the input polygon from a given Medial Axis by generating the original maximal inscribed circles centred at all the points on the Medial Axis and finding the union of all these. Section 5.2 showed how it is possible to generate polygons from the skeleton by taking an edge or sets of edges and determining the boundaries associated with these edges. The smallest possible parts will therefore be polygons generated from a single skeleton edge.

By limiting the domain to polygons generated from connected sets of edges in the skeleton, there is now an upper limit on the total possible number of regions that can be generated, providing there is a finite set of edges. As noted in Section 5.2, for every connected set of edges, there exists a unique polygon; therefore, if there is a finite set of edges, there will be a finite set of polygons that can be generated from this set. This is of an order similar to the size of the power set of the input set, though this will be lower and dependent upon the amount of branching within the graph.

This is because the sets that generate polygons have an additional requirement in place: that they are a connected set of edges. By this it is meant that for any set of connected edges, there exists a path from every point to all other points in the set. Single edges can be considered to be connected to themselves. Thus, the upper and lower bounds of the number of possible polygons is related to the graph structure of the skeleton.

Figure 6.1 shows examples of different graphs, together with the total number of connected sub-graphs and $|\mathcal{P}(n)| - 1$ where n is the number of nodes and $\mathcal{P}(n)$ is the power set of nodes. The reason for subtracting 1 from the total is because the power set includes the empty set, and for the purposes of calculating the possible number of polygons, only non-empty sets are of interest. For each graph, a node can be considered to be a polygon and an edge between two nodes represents that two polygons share at least one polygon edge.

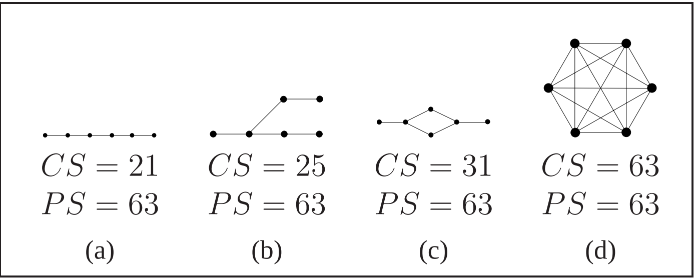


Figure 6.1: Examples of different graphs together with the number of connected sub-graphs within the graph, and $|\mathcal{P}(n)| - 1$ where n is the number of nodes and $\mathcal{P}(n)$ is the power set of nodes. For each, CS represents connected sub-graphs and PS represents $|\mathcal{P}(n)| - 1$. For comparison, $n = 6$ in all examples.

In Figure 6.1a, all nodes are of degree 1 or 2, hence connected subgraphs are paths between two nodes. Because of the structure of the graph, there exists one path between any pair of nodes, hence the total number of paths within n nodes is the same as ${}^{n}C_2 = \frac{n!}{2(n-2)!}$ (the number of ways of choosing 2 numbers from n options where ordering is important). In addition to this, each node can be considered on its own, thus the total number of connected sub-graphs is ${}^{n}C_2 + n$ or $\frac{n!}{2(n-2)!} + n$. Because this is the simplest graph that would be generated for an input polygon, this is also the lower bound for the number of sets.

Figure 6.1d is an example of a complete graph, where each node is connected to all other nodes. Since the graph is complete, the total number of sub-graphs is equal to $2^n - 1$, which is equal to $|\mathcal{P}(n)| - 1$, where n is the number of nodes. This is therefore an upper bound of the possible size of the domain. In practice, the graph characterising the Medial Axis skeleton will be very sparsely connected, thus the total number of possible polygons will be considerably lower than this upper bound.

For Figures 6.1b – 6.1c, the number of connected sub-graphs will be larger than the lower bound. In Figure 6.1a, each connected sub-graph had only two nodes of degree 1 (the start and end nodes of the path), whereas in Figure 6.1b, the multiple branching due to the node of degree 3 means it is possible for a sub-graph to have more than 2 nodes of degree 1, thus increasing the number of connected sub-graphs. In Figure 6.1c, the number of sub-graphs increases further due to the cycle within the graph. Because of this, there may no longer be a unique path between two nodes, adding further to the number of sub-graphs contained. The branching problem due to the nodes of degree 3 increases the total as before. For graphs similar to these, there is no known formula that relates the number of connected sub-graphs to n.

From the examples above, therefore, it follows that it is only possible to calculate the upper and lower bounds of the size of domain of all possible polygons generated from a simplified Medial Axis skeleton. Since the nodes in the examples above represent polygons and an edge represent a connection between two polygons, they need to be reversed to represent the Medial Axis skeleton. Thus, in the examples above, the nodes represent a skeleton edge (since each set of skeleton edges generates a unique polygon), and the edges represent a node shared by two skeleton edges.

Therefore, for a skeleton with n edges, the total number of polygons that can be generated is equivalent to the size of $|n|$, which is bounded as $\frac{n!}{2(n-2)!} + n \leq |n| \leq 2^n - 1$. In general, the Medial axis skeleton will be sparsely connected; thus, the domain of polygons will resemble a variation of Figure 6.1b. In addition, if any islands occur within the dataset, then part of the graph will be a loop similar to Figure 6.1c.

Ideally, the full domain of polygons would be generated in advance, to ensure that all possible instances of queries already exist. However, as already shown, even limiting the domain to polygons generated from connected sets of skeleton edges, still results in large numbers of potential polygons. For example, if the skeleton contained 1,000 edges, then the number of polygons P will be within the range $500,500 \leq P \leq 2^{1,000} - 1$. The skeletons of the datasets considered in this thesis all have several thousand skeleton edges, hence generating the full domain is infeasible due to memory and storage requirements. Therefore, instead of generating the full domain, a more suitable approach is to consider generating only the polygons *relevant* to a query, thus allowing a particular query to be solved.

6.3.4.2 Generating Relevant Polygons

Looking at this in more detail, consider the following query:

$$\mathsf{water}(x) \tag{6.3}$$

The intended meaning of the predicate $\mathsf{water}(x)$ is that x is a 2-dimensional region that is entirely covered by the matter type *water*. But since every water polygon is divisible into an infinite number of smaller polygons that are also covered by water, this query has an infinite number of answers. However, it is unlikely that any user of a GIS (which is the intended user of GEOLOG) would be interested in this infinite set of regions. Moreover, the set of water polygons is completely characterised by the set of maximal connected water polygons. The terms 'maximal' and 'connected' will now be defined to show their meaning in this thesis. In brief, 'connected' refers to a region being self-connected, whereas for a region to be 'maximal' it must satisfy a given attribute and not be part of a larger region with that attribute (thus could not be extended further).

First, it is necessary to define a formula $\mathsf{sum}(x, y)$, which represents the spatial sum or union of two regions. From this, it is possible to define self-connectedness to be equal to the sum of a set of connected regions (Giritli, 2003):

$$w = \mathsf{sum}(x, y) \leftrightarrow \forall z \big[\mathsf{C}(z, w) \leftrightarrow [\mathsf{C}(z, x) \vee \mathsf{C}(z, y)]\big] \tag{6.4}$$

$$\mathsf{CON}(x) \equiv_{df} \forall yz[\mathsf{EQ}(x, \mathsf{sum}(y, z)) \rightarrow \mathsf{C}(y, z)] \tag{6.5}$$

Equation 6.4 states that w is equal to the spatial sum of regions x and y iff all parts of w are connected to either x or y[5]. This spatial sum is then used in Equation 6.5 to define self-connectedness (CON); if x is self-connected, any two regions whose spatial sum is equal to x must be connected to each other. Thus x is a single connected region; it is possible to travel from any part of x to any other part of x without actually leaving the region.

From these, a definition of maximality is given as:

$$\mathsf{MAX}[\phi(x)](y) \equiv \phi(y) \wedge \mathsf{CON}(y) \wedge \neg\exists z[\phi(z) \wedge \mathsf{CON}(z) \wedge \mathsf{PP}(y, z)] \tag{6.6}$$

[5]Here, it is assumed that all regions are non-empty.

A region is defined as being maximal for a given attribute ϕ iff it is a self-connected region[6] satisfying that attribute, and there does not exist a larger self-connected region which it is a proper part of that satisfies the attribute also.

Now consider the query:

$$\mathsf{water}(x) \wedge \mathsf{max\_linear\_part}(x) \tag{6.7}$$

The intended meaning of this predicate is that *x* is a maximal 2-dimensional linear polygon that is also water. Section 5.3.1 described a method of generating 'linear' polygons, which are maximal (no linear polygon is a proper part of a larger linear polygon under the same standpoint) and self-connected (for a given linear polygon, all parts of the polygon can be reached from all other parts). For this example, it is assumed that the threshold for linearity is fixed, hence a region is always marked as either linear or non-linear.

However, since each line is marked as linear or non-linear individually, it would, in fact, be possible to carve these up into different combinations if the requirement of results being maximal polygons was relaxed, as a sub region of a linear polygon considered independently may also satisfy the requirements of being linear. Again, this query could have a large range of answers consisting of different combinations of linear points. However, it is unlikely that a single linear point or a smaller linear region inside a larger one would be of interest; thus, again, only the maximal connected linear polygons need be considered.

By generating polygons in this manner, there is a strong intuition that GEOLOG contains all the regions needed in order to answer queries of interest to a GIS user, with predicates segmenting the base regions into maximal connected polygons (based upon measurements relative to standpoints) allowing the domain to be segmented into relevant regions. An implementation is therefore needed that is capable of dealing with regions with implicit geometrical boundaries, that are determined by (but not explicitly present in) the base polygons, without explicitly modelling potentially infinite geometrical dissections of space. Thus, when presented with Equation 6.7, the implementation should be able to return only the maximal regions, as opposed to infinitely carving up potential regions as 'linear'.

The implementation used to control the quantification, referred to as *effective generator relations*, was introduced by Bennett et al. (2008). The intention is that these represent logically, algorithms that would be implemented within a GIS to carve data into

[6]It is not strictly necessary for a maximal region to be self-connected, and examples exist of regions that are disconnected but represent the maximal covering of an attribute. In this thesis, however, maximal is defined to require the region in question is self-connected.

more useful regions, thus bridging the gap between the data segmentation level (where linear regions are identified, for example), and the logic reasoning level where these features are to be used. Thus, the inclusion of effective generator relations allows the system to perform both the data segmentation and logical reasoning.

The aim is to construct an ontology that is computably tractable over a concrete domain, by constraining quantification in such a way that all spatial regions that are relevant to the evaluation of a given formula are either present in an initial finite set of entities, or are members of further finite sets that can be computed from the initial entity set. This is performed via a relatively limited modification of first order logic.

Let $\Gamma(t_1, \ldots, t_m; x_1, \ldots, x_n)$ be a relation, such that given any input m-tuple of ground terms $\{t_1, \ldots, t_m\}$ it is possible to effectively compute the set of all output n-tuples $\{x_1, \ldots, x_n\}$, such that $\Gamma(t_1, \ldots, t_m; x_1, \ldots, x_n)$ holds. Thus, given a finite set of input tuples, there is a finite set of output tuples such that some pair of input and output tuples satisfies Γ.

A possible definition of linear using this form could be max_linear_part(r, r'), where given an input polygon r there are a finite number of polygons r' corresponding to maximal linear parts with respect to some linearity predicate within r. This is finite in this case due to the limitations of the data being used, namely the process used to generate polygons from the skeleton derived from the input data-set. As previously noted, for each skeleton edge (or connected set of skeleton edges), there is an associated polygon that can be generated. Since the skeleton is made up of a finite number of edges, it follows there will be a finite number of associated polygons, a subset of which would correspond to max_linear_part(r, r').

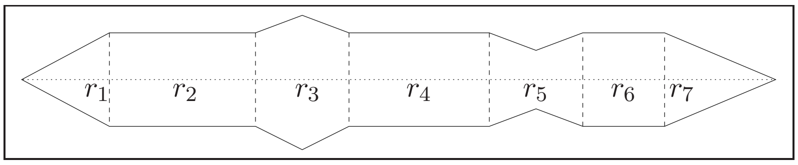


Figure 6.2: An example of how the effective generator relations are used. Here the effective generator relation max_linear_part(r, r') has been applied to the input polygon R, represented by the continuous black line. The dotted line represents the skeleton of the polygon. The result of this effective generator relation is the segmentation into 5 different regions, labelled $r_1 - r_7$, where regions whose width does not vary along the skeleton are marked as linear. Therefore, the results of applying the effective generator relation are max_linear_part$(R, \{r_2, r_4, r_6\})$.

An example of this is given in Figure 6.2, where an example of max_linear_part(r, r') has been implemented. In the example, polygon R has been segmented according to the effective generator relation. For the purposes of this example, linear is defined as no variation in the width of the maximal inscribed circles along the skeleton. Thus, the regions r_2, r_4 and r_6 would be marked as linear, as these represent the maximal lengths of skeleton where the width along the skeleton is not changing. Thus, the results of the effective generator relation would be $\mathsf{max\_linear\_part}(R, r_2, r_4, r_6)$.

Given the effective generator relation Γ, the following form of controlled quantification can now be defined:

$$(\forall x_1, \ldots, x_n : \Gamma(t_1, \ldots, t_m; x_1, \ldots, x_n))[\phi(x_1, \ldots, x_n)] \tag{6.8}$$

Let Base represent the input polygon for a dataset, which can be represented as a single polygon or a series of smaller polygons whose union is equivalent to the single polygon (for some functions, having a single, large initial polygon would prove slower than having a series of smaller polygons). By restricting the variables $t_1, \ldots, t_m$ to quantifications over Base or domains specified by other effective generators, the range of each variable is also restricted to Base or a set of entities that can be computed from Base by applying algorithms corresponding to a series of effective generator relations.

Semantically, Equation 6.8 is equivalent to:

$$(\forall x_1, \ldots, x_n)[\Gamma(t_1, \ldots, t_m; x_1, \ldots, x_n) \rightarrow \phi(x_1, \ldots, x_n)] \tag{6.9}$$

Equation 6.7 can be represented using this format, allowing the segmentation of linear polygons. Further, the closeness predicate described in Section 5.3.3 to define regions ‘close-to’ other regions could also be represented. Thus, using effective generator relations allows instantiations over predicates, ensuring the domain contains all instances needed to answer a given query. The decision as to whether to pre-compute these functions or evaluate as required will be considered later in this section.

6.3.5 Solving First Order Logic Queries with Model Checking

6.3.5.1 Prolog Overview

The intended usage of GEOLOG is that a user can enter first order logic queries, which GEOLOG can then run, either confirming that the query is true, or returning results that satisfy that query. This requires a method of parsing the query to ensure that it can be input into Prolog, then handling the query correctly.

In Prolog, clauses are written in the form:

$$H \text{ :– } B_1, B_2, \ldots, B_n.$$

These are intended to be of the same form as a Horn clause, with a head literal to be proven by a series of literals that make up the body. Thus, to prove H (the head), the body of literals must be proven also: prove B_1, prove B_2 through to proving B_n. Prolog is thus equipped to handle logical conjunction and disjunction with simple parsing from the first order logical query to Prolog code. However, problems may arise when handling negation within queries, due to the method Prolog uses to handle negation. Further, although Prolog does include existential quantification for some predicates, it does not necessarily produce the results that we would prefer. Thus, it is important that an approach is used that will allow the use of quantification and negation correctly when they are present in a given query.

Negation in Prolog is handled using Negation as Failure, as proposed by Clark (1977). Thus $\neg\mathsf{p}$ is derived from the failure to derive p. Whilst this handles most cases satisfactorily, it is not the same as logical negation. For example, consider the following set of clauses in Prolog:

$$\begin{array}{lll} \mathsf{TV\_show}(X) & \text{:–} & \mathsf{simpsons}(X). \\ \mathsf{TV\_show}(X) & \text{:–} & \mathsf{bigbrother}(X). \\ \mathsf{bigbrother}(a). & & \\ \mathsf{simpsons}(b). & & \end{array}$$

In Prolog, clauses that consist of a head only are called *facts*, and thus are treated as true. In addition, capital letters represent *variables*, whilst lower case letters represent *atoms*. The clauses above are meant to represent that *Big Brother* and *The Simpsons* are TV shows, of which a and b are examples. Running the query $\mathsf{TV\_show}(X)$ would thus return a and b as solutions. Consider the following clauses following on from the previously given set:

$$\mathsf{enjoys1}(dave, X) \text{ :– } \mathsf{TV\_show}(X), \backslash\text{+ } \mathsf{bigbrother}(X). \tag{6.10}$$

$$\mathsf{enjoys2}(dave, X) \text{ :– } \backslash\text{+ } \mathsf{bigbrother}(X), \mathsf{TV\_show}(X). \tag{6.11}$$

The intention of these examples is a clause that states $dave$ enjoys a TV show that isn't *Big Brother*, as negation in Prolog is represented by $\backslash+$. Further, the two are logically equivalent, since $A \wedge \neg B \equiv \neg B \wedge A$. However, whilst Query 6.10 will succeed in returning b as a result, Query 6.11 would fail to return anything.

In each case, Prolog tests to see if each of the clauses holds. If it has a variable, Prolog will attempt to find an instantiation of that variable that holds true, then pass this along to the next clause to see if it still holds. For Query 6.10, instantiating X to a or b holds for the first clause, but only b will hold for the second clause, thus b is returned as a solution. With Query 6.11 though, Prolog attempts to see if $\backslash+ \mathsf{bigbrother}(X)$ holds by checking that $\mathsf{bigbrother}(X)$ fails, which it will not do due to the existing fact. Because this clause fails, the whole clause fails automatically, thus no answer is returned. Negation in Prolog thus needs to be used with caution, with particular attention paid to the ordering of the clauses. However, it cannot be guaranteed that the first order logic queries will be ordered in a way suitable for Prolog; hence, negation should be handled correctly wherever it appears in the query.

6.3.5.2 Model Building

The solution proposed here is to use a model building approach such as the one proposed for natural language processing by Bos (2003) and Blackburn and Bos (2005), which can be performed in Prolog as shown in the online literature for a Computational Semantics course[7]. The approach does not build a full first order theorem prover; rather it provides a method of checking models built using first order formulae. To do this, a model of knowledge is built, then used to test the formulae to see if it is true or false, and if so return any polygons that match the formulae (if desired).

The approach used here for model building is similar to that of MACE (McCune, 2003), where an attempt to reduce the connectives within the query is undertaken through inferences, then parts of the query are solved by extracting information from the model, until the query is proven to be consistent (and thus can return the values which it is consistent for) or inconsistent. Such an approach is ideally suited to Prolog, since rules can easily be defined to translate a query into a tree, whereby each of the leaves is solved individually and the truth values determined are passed up the tree to solve further parts.

It is important to clarify the use of model within this context, particularly the completeness of the model to be used. A model is an assignment of values to all predicates and relations. Thus, a complete model would consist of all predicates and relations and the values assigned to them. However, when answering a query, it is sufficient to restrict attention to partial models, which only assign values to the predicates and relations occurring within the query. This is because the extension of any complex predicate is determined compositionally by the extension of its atomic constituents. Thus, the com-

[7]Online literature for Computational Semantics course at MiLCA:
http://www.coli.uni-saarland.de/projects/milca/courses/comsem/html/index.html (Visited, October 2007)

plete model of all knowledge about a domain is not required; only the partial model that contains enough knowledge to answer a given query.

When presented with first order formulae, it is first necessary to construct the model of the formulae. This is done by taking each of the predicates in the query individually and finding all possible instantiations of that predicate, independent of the rest of the query. For a query such as $\mathsf{EC}(x, y) \vee \mathsf{DC}(y, z) \ldots$, the set of all polygons that have the RCC relation EC would be found, then the process would be repeated for all polygons that have the RCC relation DC. All logical constructors are ignored at this stage, as the intention is to construct models of all possible values. The previously noted query may generate a model with a set of predicate instances, such as $\{\mathsf{DC}(1, 3), \mathsf{DC}(2, 3), \mathsf{EC}(1, 2), \ldots\}$, which implicitly generates a possible domain of $\{1, 2, 3, \ldots\}$.

An important consideration at this stage is whether to generate predicate instances as they are required by the query, or whether to pre-compute the knowledge within the model (or part of the knowledge). For example, the calculation of RCC relations was discussed previously in Section 5.4; thus, one option would be to calculate the RCC relations contained within a query as they appear. However, a more efficient approach is to instead calculate all RCC relations in advance and store these as asserted facts within Prolog. This is more practical, given that the RCC relation between two polygons will not change (unless one or both of the polygons is modified), hence the calculation need only be performed once. Thus the relations are stored using an asserted predicate $\mathsf{RCC\_STORE/3}$, where for each pair of polygons x and y the RCC-8 relation c is stored as $\mathsf{RCC\_STORE}(x, y, c)$.

To ensure the domain contained within the partial model is sufficiently large, all positive instantiations of a predicate are stored. Thus, for RCC relations, the RCC relations for all pairs of existing polygons are stored, since the internal Prolog predicate is $\mathsf{RCC\_STORE}$, and all instantiations of this predicate results in all stored RCC relations being returned. Such an approach is useful for the handling of negation, as will be discussed later in the section.

To store the model, GEOLOG asserts the results of each predicate as facts within Prolog, with the suffix "$\mathsf{\_mem}$" attached to signify the fact is to be stored in memory for the model. For example, if a predicate had the results $\phi(1), \phi(2) \ldots$, these would be asserted as $\phi\mathsf{\_mem}(1), \phi\mathsf{\_mem}(2)$. Prolog can use these asserted facts to instantiate free variables within the query quickly, by testing whether an instance of a particular predicate has been asserted with a value or not. This also means that information about predicates need only be collected once during model construction. Finally, storing in this manner aids the handling of negation and quantification, as discussed later in this chapter.

Once this model is constructed, the query can be tested against it. The aim for a given query is to instantiate the variables to ensure the entire query is satisfied. If there are no free variables within the query, then all that is required is to return whether the query is true or false (e.g. testing whether a particular relation holds within the model), whereas if there are any free variables, it is required that the values for which the query holds true are returned. This instantiation should also be independent of ordering, to allow the user to enter the query in any form they wish, e.g. $\mathsf{EC}(x, y) \vee \mathsf{DC}(y, z) \leftrightarrow \mathsf{DC}(y, z) \vee \mathsf{EC}(x, y)$. A process of normalisation is therefore required to convert the query into a normal form.

6.3.5.3 Normalising Queries

The normal form chosen for GEOLOG was adapted from Negation Normal Form, where negation only occurs immediately before elementary propositions, and the only logical connectives used are $\{\neg, \wedge, \vee\}$. Quantification, however, was modified, and only existential quantification is allowed within a query. In NNF, both existential and universal quantification are allowed, with a rewrite rule used to move negation inward. However, existential quantification is more suited for use in Prolog, thus the modified NNF used by GEOLOG will allow negation to occur immediately before existential quantification.

To convert into NNF, the following rewrite rules are used:

$$\phi \rightarrow \psi \equiv \neg\phi \vee \psi \tag{6.12}$$

$$\neg(\phi \rightarrow \psi) \equiv \phi \wedge \neg\psi \tag{6.13}$$

$$\neg(\phi \wedge \psi) \equiv \neg\phi \vee \neg\psi \tag{6.14}$$

$$\neg(\phi \vee \psi) \equiv \neg\phi \wedge \neg\psi \tag{6.15}$$

$$\neg\neg\phi \equiv \phi \tag{6.16}$$

$$\neg\forall x[\phi(x)] \equiv \exists x[\neg\phi(x)] \tag{6.17}$$

$$\neg\exists x[\phi(x)] \equiv \forall x[\neg\phi(x)] \tag{6.18}$$

Logical implication can be replaced using Equations 6.12 and 6.13, and negation can be moved inwards using De Morgan's Laws (Equations 6.14 and 6.15). Equation 6.16 removes double negation. As noted above, the modified NNF used by GEOLOG only allows existential quantification, hence Equation 6.17 can be used by GEOLOG, whereas 6.18 cannot not be used.

NNF can be converted into the stronger Conjunctive Normal Form (CNF) or Disjunctive Normal Form (DNF) by applying the distributivity laws, for example to input the query into automate theorem provers (which typically use CNF or DNF). However, applying the distributivity laws would potentially be computationally expensive and is not necessary for solving the query in Prolog, hence NNF is more suitable.

To normalise the query into the form used by GEOLOG, requires translating the predicates using the rewrite rules discussed above, with the additional rules for handling quantification correctly. This translation is represented here using:

$$\mathsf{TR}(\phi) \rightarrow \mathsf{TR}(\phi')$$

where ϕ is a first order logic formula and ϕ' is the result of applying a translation towards the modified NNF. TR is evaluated recursively, until the formula is fully normalised. The translations used are as follows:

$$\mathsf{TR}(\phi \wedge \psi) \rightarrow \mathsf{TR}(\phi) \wedge \mathsf{TR}(\psi) \tag{6.19}$$
$$\mathsf{TR}(\phi \vee \psi) \rightarrow \mathsf{TR}(\phi) \vee \mathsf{TR}(\psi) \tag{6.20}$$
$$\mathsf{TR}(\phi \rightarrow \psi) \rightarrow \mathsf{TR}(\neg\phi) \vee \mathsf{TR}(\psi) \tag{6.21}$$
$$\mathsf{TR}\big(\neg(\phi \wedge \psi)\big) \rightarrow \mathsf{TR}(\neg\phi) \vee \mathsf{TR}(\neg\psi) \tag{6.22}$$
$$\mathsf{TR}\big(\neg(\phi \vee \psi)\big) \rightarrow \mathsf{TR}(\neg\phi) \wedge \mathsf{TR}(\neg\psi) \tag{6.23}$$
$$\mathsf{TR}\big(\neg(\phi \rightarrow \psi)\big) \rightarrow \mathsf{TR}(\phi) \wedge \mathsf{TR}(\neg\psi) \tag{6.24}$$
$$\mathsf{TR}(\neg\neg\phi) \rightarrow \mathsf{TR}(\phi) \tag{6.25}$$
$$\mathsf{TR}(\neg\phi) \rightarrow \neg\mathsf{TR}(\phi) \tag{6.26}$$
$$\mathsf{TR}(\exists x[\phi]) \rightarrow \exists x[\mathsf{TR}(\phi)] \tag{6.27}$$
$$\mathsf{TR}(\neg\exists x[\phi]) \rightarrow \neg\exists x[\mathsf{TR}(\phi)] \tag{6.28}$$
$$\mathsf{TR}(\forall x[\phi]) \rightarrow \neg\exists x[\mathsf{TR}(\neg\phi)] \tag{6.29}$$
$$\mathsf{TR}(\neg\forall x[\phi]) \rightarrow \exists x[\mathsf{TR}(\neg\phi)] \tag{6.30}$$

6.3.5.4 Determine Variables

The variables within the query need to be replaced with Prolog variables, to allow them to be instantiated. This replacement needs to be consistent, for example, all instances of x within the query should be replaced with the same Prolog variable (such as X). This can be accomplished through the use of Prolog's built-in functions for dismantling and building terms and multiple passes through the query: A first pass through collects the set of variables used within the query. This set is then mapped to a set of Prolog variables, and another pass through the query is performed, replacing each variable with the Prolog variable designated by the mapping. This would, for example, replace the query $\mathsf{person}(x) \wedge \mathsf{food}(y) \wedge \mathsf{eats}(x, y)$ with $\mathsf{person}(X) \wedge \mathsf{food}(Y) \wedge \mathsf{eats}(X, Y)$, as

X and Y are Prolog variables.

6.3.5.5 Solving Queries

Once the query is in the modified NNF, GEOLOG can use Prolog's built in search to solve the query. As already noted, GEOLOG first constructs the model of the query by finding all possible instantiations of predicates individually, irrespective of logical connectives. These results are then asserted as facts, allowing Prolog to then use these facts to determine instantiations that solve the query.

By normalising the query using the translations given previously, a tree-like structure is generated within the system, where the logical connectives can branch into other parts. The leaf nodes represent predicates and the final result is generated by determining the truth values at leaf nodes and passing these upwards through connectives. This normalisation also means the query is translated down to the call level of Prolog, where a predicate $\phi(x)$ can be answered using the Prolog command:

$$:\!-\ \mathsf{call}(\phi(X)).$$

where $\mathsf{call}(\phi(X))$ runs the predicate $\phi(X)$, returning a value for X if one can be found that satisfies the predicate, or failing if no such value can be found. Because all possible instantiations of a predicate in a query are stored as asserted facts (as noted earlier), it follows that $\mathsf{call}(\phi(X))$ succeeds iff $\exists x[\phi(x)]$ within the partial model, and therefore $\mathsf{call}(\phi(X))$ fails iff $\neg\exists x[\phi(x)]$ within the partial model. Therefore, quantification can be handled correctly, whilst the handling of negation is improved over using Negation as Failure.

Because $\mathsf{call}(\phi(X))$ succeeds iff $\exists x[\phi(x)]$, it follows that existential quantification can be handled using this approach, by treating the existential formula as a query in itself. Thus, to solve the first order logic formula $\exists V[\phi]$, where V is the set of variables over which ϕ is to be quantified, GEOLOG attempts to solve the query ϕ by instantiating all variables in V, the values of which can then be used in other parts of the query.

With negated equivalence ($\neg\exists x[\phi(x)]$), Prolog's built in use of Negation as Failure is sufficient to determine whether to return true or false, as it is only necessary to determine whether $\phi(x)$ can be instantiated to be true or not. If any such instantiation exists, then $\neg\exists x[\phi(x)]$ is clearly false, otherwise it is true. From this, universal quantification can be handled, as it is translated into existential quantification as noted above.

A potential problem may arise with quantification when there are multiple instances of quantification in a query that use the same variable. For example, consider the query

$\exists x[\phi(x)] \wedge \exists x[\psi(x)]$. Through the variable replacement stage, this becomes $\exists X[\phi(X)] \wedge \exists X[\psi(X)]$. When solving the query, X would become bound to a value such that $\phi(x)$ is satisfied, which would then be passed to $\psi(x)$ and tested to see if such an instantiation of $\psi(x)$ existed. Thus the query has been treated as $\exists x[\phi(x) \wedge \psi(x)]$, or 'there exists an x such that x is a ϕ and x is a ψ'. However, the actual meaning is that there exists an instance of ϕ and an instance of ψ, which is different; in the first version there needs to exist a value that x can be instantiated for both predicates, whereas in the second meaning there need only exist an instantiation of both predicates and not necessarily with the same value of x.

This could be rectified at the variable replacement stage, by requiring that each quantification uses a different variable. The above example would thus become $\exists X[\phi(X)] \wedge \exists Y[\psi(Y)]$, which would be the intended meaning. GEOLOG assumes that existential quantifiers are uniquely given, hence the problem above would not occur, but a solution can be implemented if required, which future work on GEOLOG could consider.

Negation can now be handled irrespective of the position of the negated predicate or whether the predicate has uninstantiated variables or not. As noted above, if a particular instantiation of a predicate is not part of the model, it is implicitly false. It follows from this that $\mathsf{call}(\neg\phi(X))$ succeeds iff $\neg\exists x[\phi(x)]$ within the partial model, thus testing whether $\neg\phi(x)$ can be performed by searching the model for instantiations of $\phi(x)$. If all the variables within a predicate are instantiated, the model is tested to see if it contains the predicate with the given instantiation; if it does not, then the negation of that predicate is true, else it is false if the model does contain it.

Because the domain of the model is explicitly stored, it is also possible to handle negated predicates containing uninstantiated variables. This is performed by instantiating the variables using values from the domain such that the predicate is false (i.e. if the x in predicate $\neg\phi(x)$ was uninstantiated, a value would be assigned from the domain for which $\phi(x)$ is false, hence $\neg\phi(x)$ is true). For example, if GEOLOG was presented with the query $\neg\mathsf{water}(x)$, it would find all polygons which did not have the attribute water associated with them, which, given the limitations placed upon matter types, would return the land and sea polygons.

However, the representation of negation here is not negation in the pure logic sense, since it can only return instances that exist within the model already. For example, with linearity, the dataset is segmented by marking each skeleton edge as linear or non-linear, then forming polygons from maximal connected sets of these. Thus, the query $\mathsf{linear}[l](x) \wedge \mathsf{water}(x)$ would return all polygons that have the attribute 'linear' associated with them for a given threshold. However, the query $\neg\mathsf{linear}[l](x) \wedge \mathsf{water}(x)$ will only return polygons existing within the domain already that do not have the attribute 'linear' attached. These are not the only polygons that could satisfy the query; for instance, the union of a linear polygon and a connected non-linear polygon would satisfy the query but would not have been generated by the linearity segmentation.

Therefore, the use of negation still requires caution when used in queries. For instance, to refer to the regions that are 'left over' from the linearity segmentation, the term 'expansive' was introduced, as opposed to non-linear. Another requirement is that every variable must occur in at least one positive occurrence, hence the example above is limited to the existing domain only.

The final consideration is generating the set of all results for a query (if more than one solution exists). This can be achieved using the built-in Prolog function findall(T, G, L), where L is the list of all the instances of the Template T for which the goal G succeeds. In GEOLOG, G will be the query being considered, and L the list of variables for which results are required e.g. $\phi(x) \wedge \exists y[\psi(x, y)]$ becomes the Prolog goal:

$$\mathsf{findall}([X], \mathsf{fol\_solve}(\phi(X) \wedge \exists Y[\psi(X, Y)]), Answer).$$

where fol_solve is the function used by GEOLOG to evaluate a query, and $Answer$ is the list of all solutions to the query. This list can be converted into a set to reduce multiple occurrences of results.

Because of the way GEOLOG uses findall to generate multiple solutions, it is worth noting a slight difference in the way existential quantification is interpreted within the query, and the repercussions of this. For example, consider the query $\phi(x) \wedge \exists y[\psi(x, y)]$. This query is true if for an instantiation of x, $\phi(x)$ is true and there is at least one instantiation of y such that $\psi(x, y)$ is true. Therefore, when searching for all solutions to this query, GEOLOG should ideally only test each possible instantiation of x once, since it does not matter if there are multiple instantiations of y that makes the query evaluate to true.

However, because findall works through all possible solutions, for every instantiation of x it will find all instantiations of y that satisfy the query, hence the need to perform a set ordering operation upon the results. Future improvements to GEOLOG would ideally rectify this, ensuring that instantiations of free variables are only tested once.

6.3.5.6 Example Query Handling

The query solving process can be summarised with the following steps:

- Construct the Model - This is performed by determining the predicates that make up the query, then finding all possible instantiations for each predicate. These are stored as asserted facts.

- Normalise the query - By applying the translation rules listed previously, the query is converted into a modified form of NNF, thus negation can only occur immediately before primitives and existential quantification. The only logical connectives are $\{\neg, \wedge, \vee\}$ and existential quantification is the only form of quantification used.

- Convert the variables into Prolog variables - This is performed using multiple passes and Prolog's built-in term dismantling and building, generating a consistent set of variables through the query.

- Solve the Query - Using Prolog's built in search, find instantiations of all the variables that ensure the query as a whole is satisfied.

An example of working through a query is now provided. The query to be considered is $\mathsf{A}(x, y) \wedge (\mathsf{B}(x, z) \vee \mathsf{B}(y, z))$, with no particular meaning attached to the predicates outside of the query and the domain that will be provided. The query is already in the modified NNF described previously, thus no translations are required. For the purposes of this example, the variables $\{x, y, z\}$ are assumed are to be Prolog variables also, as the variable conversion stage is trivial for this example.

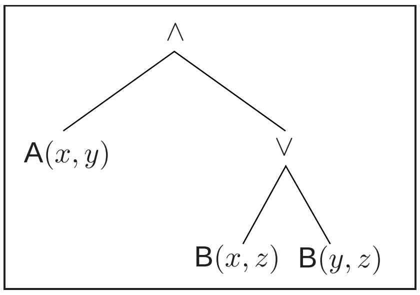


Figure 6.3: An example of a tree representation of the query $\mathsf{A}(x, y) \wedge (\mathsf{B}(x, z) \vee \mathsf{B}(y, z))$. The tree is traversed using depth-first search, where the predicates are visited in the order $\{(\mathsf{A}(x, y), \mathsf{B}(x, z)), \mathsf{B}(y, z)\}$.

The representation of this query in a tree-like structure is shown in Figure 6.3. Suppose the following partial model was constructed for the query, where M is the set of predicates stored in the model and D is the domain explicitly stored by these predicates:

$$\begin{aligned} D &= \{1, 2, 3, 4, 5\} \\ M &= \{\mathsf{A}(1, 2), \mathsf{A}(2, 5), \mathsf{A}(4, 2) \\ &\quad \mathsf{B}(1, 5), \mathsf{B}(3, 3), \mathsf{B}(5, 3)\} \end{aligned}$$

The model is stored as a series of asserted facts with the suffix "$\mathsf{\_mem}$" attached, to ensure Prolog searches the stored facts as opposed to testing the predicate. Thus the model above would be stored as a set of facts such as:

$$\mathsf{A\_mem}(1,2). \tag{6.31}$$
$$\mathsf{A\_mem}(2,5). \tag{6.32}$$
$$\vdots$$
$$\mathsf{B\_mem}(5,3). \tag{6.33}$$

For this example, the original predicate is referred to for clarity, rather than the asserted memory predicate. To solve the query, an instantiation of the set of variables $\{x, y, z\}$ needs to be found that ensure that the query is satisfied. Using a depth-first search, the first predicate that is visited is $\mathsf{A}(x, y)$; thus Prolog would search for an instantiation of this predicate using the Prolog command:

$$\mathsf{:\!-call}(\mathsf{A}(x, y)).$$

The first result found would be $\mathsf{A}(1, 2)$, which means x is now set to 1 and y is set to 2. Prolog would now move on to the other part of the logical conjunction, first trying to solve $\mathsf{B}(x, z)$. As x has been instantiated to 1, Prolog would now search the model using

$$\mathsf{:\!-call}(\mathsf{B}(1, z)).$$

This would find $\mathsf{B}(1, 5)$. The final predicate does not need to be visited as it is part of a logical disjunction, thus the query is already satisfied. A first solution to the query would thus return $\{x = 1, y = 2, z = 5\}$.

Prolog would then repeat the process, trying out all possible combinations from the model to determine which hold true. Thus, the solution $\{x = 2, y = 5, z = 3\}$ would also be returned, as although $\mathsf{B}(2, 3)$ is false, $\mathsf{B}(5, 3)$ is satisfied by the model. A final search using $\mathsf{A}(4, 2)$ would also return false, as there is no instantiation of z that would make $\mathsf{B}(x, z) \vee \mathsf{B}(y, z)$ true. Once all combinations have been exhausted, Prolog returns the set of solutions found. Had the query instead been $\mathsf{A}(x, y) \wedge (\mathsf{B}(x, z) \wedge \mathsf{B}(y, z))$, no solution would be found within the model above, as there would be no instantiation of the set $\{x, y, z\}$ that satisfies the query.

6.3.5.7 Limitations of the approach

The main potential drawback of this approach is its scalability, as has been noted previously by Claessen and Sörensson (2003): First, the model size will be directly linked to the number of objects within the domain, as well as the number of relations that exist between the objects (and hence the arity of the functions used). For example, in order to store all RCC relations in the model would require n^2 entries, where n is the number of polygons within the model at present. In general, the worst case number of entries for a given function or arity a is n^a, though this assumes that there is only one relation that holds for each combination within the function. It follows on from this that the number of objects present will affect the time taken to solve the query, as for every variable in a query the worst case is that that variable needs to be tested for all objects in the model. Thus the more variables present and the larger the model, the longer it will take to find all solutions.

Another drawback to the approach is the restriction to finite domains. Using this model based approach means it is not possible to query about objects which do not explicitly exist already in the defined domain or can be generated in such a manner that the resultant set is finite. This means there is a need to place restrictions upon the usage of RCC. Restricting the domain in such a manner has been discussed previously by Haarslev et al. (1998) and Grütter and Bauer-Messmer (2007), thus a similar approach is used here; namely the restricting of RCC to polygons already *existing* within the domain, as opposed to arbitrary regions. Thus when given a query such as $\mathsf{P}(x, y)$, GEOLOG will only return pairs of *existing* polygons that satisfy the relation of one being part of the other, rather than generating regions that satisfy the property.

Restricting the domain is also problematic for the handling of negation, as noted previously. Although the handling of negation improves upon Negation as Failure (since it allows negation to occur anywhere within the query), it is still limited by the generated domain. Thus, queries involving negation can only return instances that exist already, as opposed to generating all possible instances that also satisfy the query.

6.3.5.8 Testing

The approach now needs to be tested, to determine the time taken to build a partial model for a particular query, as well as the time taken to find solutions. The queries to be considered are relatively simple in terms of the predicates involved, and with the exception of the first query, will all be based upon the RCC relation EC. As was noted previously, when an RCC-8 relation is present within a query, then all possible RCC-8 relations are used in the construction of the partial model of that query; thus the models generated for each query will be the same. The variation in time to construct the models should therefore come from Prolog analysing the query to determine the predicates used.

With each query, the intention is an increase in the number of variables within the query, thus further increasing the complexity of the query and the time required to search for all possible solutions. This includes the need for quantification for later queries. Thus although the queries are similar due to their usage of EC, they do represent an appropriate test of GEOLOG, since they allow the measurement of aspects such as time taken to construct a partial model and solve a query, as well as the impact of increasing the domain size or the number of variables present within the query.

The queries that are to be considered are as follows:

$$\mathsf{Query1}(x) \equiv \mathsf{EQ}(x,x) \tag{6.34}$$

$$\mathsf{Query2}(x,y) \equiv \mathsf{EC}(x,y) \tag{6.35}$$

$$\mathsf{Query3}(x,y) \equiv \mathsf{EC}(x,y) \wedge \exists z[\mathsf{EC}(x,z) \wedge \mathsf{EC}(y,z) \wedge \neg(x = z \vee y = z)] \tag{6.36}$$

$$\begin{aligned}\mathsf{Query4}(x,y) \equiv{}& \mathsf{EC}(x,y) \wedge \exists z[\mathsf{EC}(x,z) \wedge \mathsf{EC}(y,z) \wedge \neg(x = z \vee y = z)\wedge \\ & \forall w[\mathsf{EC}(x,w) \wedge \mathsf{EC}(y,w) \rightarrow (w = z)]]\end{aligned} \tag{6.37}$$

$$\begin{aligned}\mathsf{Query5}(x,y) \equiv{}& \mathsf{EC}(x,y) \wedge \exists u,v[\mathsf{EC}(x,u) \wedge \mathsf{EC}(x,v) \wedge \mathsf{EC}(y,u) \wedge \mathsf{EC}(y,v)\wedge \\ & \neg\big((x = u) \vee (x = v) \vee (y = u) \vee (y = v) \vee (u = v)\big)\wedge \\ & \forall z[\mathsf{EC}(x,z) \wedge \mathsf{EC}(y,z) \rightarrow \big((z = u) \vee (z = v)\big)]]\end{aligned} \tag{6.38}$$

These queries do not necessarily represent anything useful that would usually be tested for, but are designed to put increasing pressure on the approach through complexity and increased numbers of variables. In addition, they represent queries that may form parts of larger queries. A natural language explanation of these queries now follows: Query1 (6.34) finds all polygons that are equal to themselves. Query2 (6.35) finds pairs of polygons that are externally connected. Query3 (6.36) finds pairs of polygons that are externally connected and share at least one other polygon that both are externally connected to. Query4 (6.37) modifies this to find pairs of polygons that are externally connected to each other and one further shared polygon. Query5 (6.38) expands this notion again to pairs of polygons externally connected to each other and two other polygons. Thus, these are queries which may occur in other queries, when there is a need for specific numbers of polygons with specific relations between them.

Because the polygons are stored with a unique identifier number, later queries can be

Table 6.1: A comparison of the times taken to construct the model for each of the example queries. All times are in milliseconds and the results of finding the average time of constructing the model 100 times.

Number of Polygons	Query1	Query2	Query3	Query4	Query5
19	6.2	6.4	6.8	7.3	7.8
47	33.8	34.2	37.5	38.0	38.2
89	155.9	155.8	157.4	158.3	160.0
145	382.3	382.4	390.0	390.5	394.5
218	737.3	737.7	738.3	740.7	742.0
301	1,334.3	1,335.0	1,342.0	1,354.7	1,396.0
371	2,021.3	2,022.0	2,024.0	2,037.3	2,040.7
460	3,279.0	3,279.3	3,288.7	3,289.7	3,291.3
563	5,183.0	5,183.3	5,196.7	5,280.0	5,295.5
671	6,925.3	6,926.7	6,930.0	6,936.7	6,946.7

evaluated more quickly, by adding an ordering upon the polygons. For example, for a predicate $\mathsf{EC}(x, y)$, GEOLOG will return the identifier numbers for two polygons that are externally connected. Since $\mathsf{EC}(x, y) \leftrightarrow \mathsf{EC}(y, x)$, it would be inefficient to search for both solutions, thus an alternative would be to include the ordering $x < y$ within the query. Although these are not included in the queries above, they were included internally within GEOLOG, to reduce the time taken.

The queries were tested using 10 sets of polygons, each increasing in number to test the scalability of the approach. The time taken to construct a model of the query was tested. Each query was run 100 times, and an average time in milliseconds was recorded. The results of these runs are shown in Table 6.1.

GEOLOG finds all solutions for a given predicate, and due to the method in which RCC relations are stored, will return all RCC-8 relations stored within the system. Thus, the time taken to construct the model for a particular domain size is relatively similar, due to the same model being constructed. With Query3 – Query5, the additional predicate of equality ($=$) needs to be considered, hence the increase in time. Rather than storing all possible pairings for the equality, GEOLOG instead stores the predicate singly, and can test for solutions as required when the predicate is determined; since the domain is explicitly stored, GEOLOG can find values from the domain that will satisfy equalities and inequalities.

An additional overhead for Query3 – Query5 is the length of the query itself, which means a slight increase in the time taken by GEOLOG to determine all the predicates contained within the query. This could have been measured separately, but was negligible in comparison to the effect of the size of the domain; as can be seen in Table 6.1, the time taken to construct the model for queries 1 and 5 only varies by approximately 20ms, despite the increase in size and additional predicate.

Next to consider, is the relation between the number of polygons in the domain and the time taken to construct the model. As would be expected, there is a strong correlation between the two, as shown in Figure 6.4, where Query1 was graphed (the time taken for all 5 queries was similar enough that only one query need be considered for clarity). This also shows that even for large numbers of polygons, the time required to construct the model is reasonably small, though this may increase depending on the number of predicates used in the query.

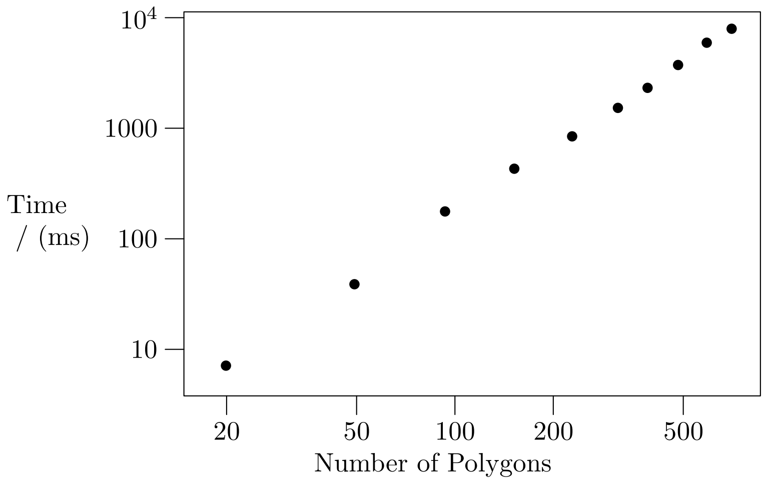


Figure 6.4: A graph of the time taken in milliseconds to construct the model of Query1 for a given number of polygons. Both axes have been scaled logarithmically. The graph highlights the correlation between the two.

Now that it is confirmed that GEOLOG can construct models quickly, it is necessary to test the performance for solving the queries. GEOLOG is first tested to see the time taken to find the first possible solution, thus confirming the satisfiability of the query in relation to the constructed model. This is performed by evaluating the query with existential quantification; for example, Query1 is tested using $\exists x[\mathsf{EQ}(x, x)]$.

The results of evaluating the queries in this manner are shown in Table 6.2. As noted, for some queries, GEOLOG was unable to find a solution within the internal time constraints of SICStus. The time taken to find any solution will, in part, be dependent upon the ordering of the model, since if the first element considered solves the query it will clearly be very fast, whereas if it has to search the entire model to find a solution the time required will be dependent upon the size of the model. For Query1 and Query2, the time

Table 6.2: A comparison of the times taken to find the first solution for each query. All times are in milliseconds and the results of performing the query evaluation process 100 times. Dashed entries represent instances where GEOLOG could not complete the query evaluation due to the time constraints in SICStus Prolog.

Number of Polygons	Query1	Query2	Query3	Query4	Query5
19	0.2	0.2	4.2	21.0	100.2
47	0.9	1.0	30.7	1,475.9	12,708.7
89	5.1	4.8	120.7	105,200.4	124,980.3
145	11.3	14.3	316.2	438,330.0	-
218	20.3	20.3	710.3	1,340,380.8	-
301	65.6	63.6	1,481.7	-	-
371	85.5	74.3	2,661.7	-	-
460	161.7	161.2	3,164.0	-	-
563	246.3	253.7	6,282.7	-	-
671	310.0	312.0	7,251.0	-	-

required remains low as there is only a single function to satisfy, hence the first instance of this in the model is sufficient. For Query3, the introduction of additional variables means a dramatic increase in the potential required search, hence the sharp increase in the time taken. Query4 and Query5 require universal quantification, and thus require the whole model to be searched to confirm there is not a counter example that would prove the query false. Thus, the time required to solve these queries takes significantly longer, to the point that they cannot be solved once the size of the domain becomes too large.

The size of the domain and the number of variables contribute to the time required to evaluate a query. The more variables that are contained within the query, the more possible combinations that need to be tested, hence the increase in time for each query at a given domain. Also, as the size of the domain increases, the time required to find a single solution increases. This follows from previous observations, since a larger domain means more possible values for the variables to be assigned to and also a larger domain means it takes longer to search through the model to find solutions. As before, a strong correlation between the size of the domain and the time taken can be shown as in Figure 6.5, where the time taken to solve Query1 and Query3 is compared.

The time required is now also affected by another factor: the number of possible solutions. Whilst Query1 and Query2 had similar times to find a single solution, Query1 is the quickest due to there only being *n* results where *n* is the size of the domain, whereas in the worst case Query2 would return n^2 results (if everything was externally connected

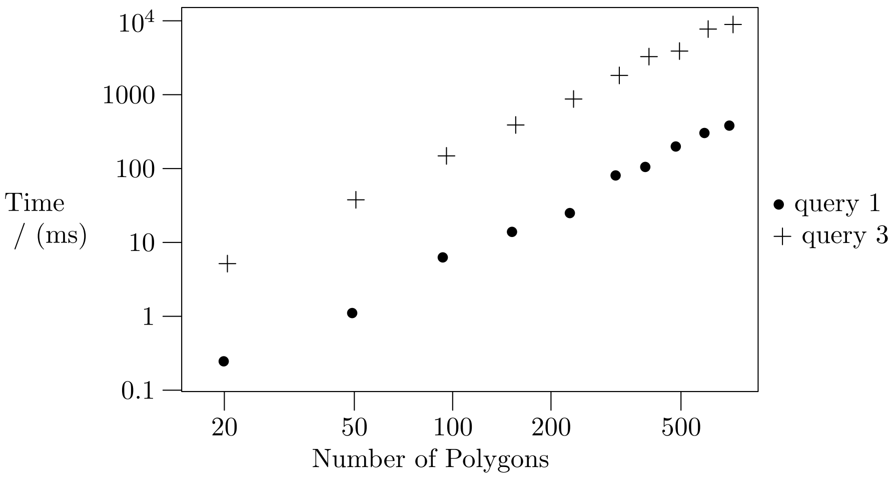


Figure 6.5: A graph of the time taken in milliseconds to find a single solution for Query1 and Query3 for a given number of polygons. Both axes have been scaled logarithmically.

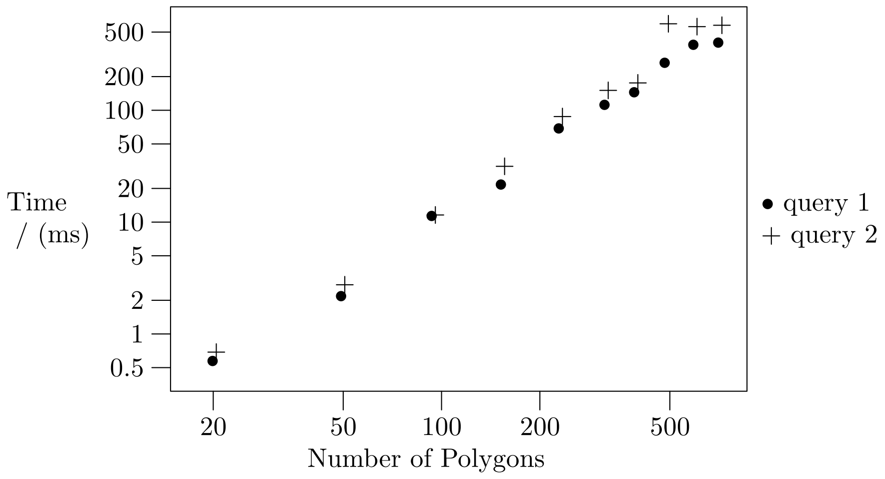


Figure 6.6: A graph of the time taken in milliseconds to find all solutions for queries 1 and 2 for a given number of polygons. Both axes have been scaled logarithmically.

to everything else). Similarly, as noted previously, existential quantification is handled by finding values to quantify with, consequently, instantiations may be tested multiple times. Thus, although there are the same number of free variables in Query2 and Query3, for each instantiation of $\{x, y\}$ in Query3, GEOLOG may find multiple instantiations of z which satisfy the existential quantifier, hence the number of possible solutions found is larger. Query4 has fewer results than Query3 due to the universal quantifier, but this addition also means evaluating an instantiation takes longer, hence the significant time increase.

In general, the worst case complexity for evaluating a query in GEOLOG will be n^m, where n is the number of entities within the domain and m is the number of variables that

Table 6.3: A comparison of the times taken to evaluate the queries and find all solutions. All times are in milliseconds and the results of finding the average time of constructing the model 100 times. Dashed entries represent instances where GEOLOG could not complete the query evaluation due to the time constraints in SICStus Prolog.

Number of Polygons	Query1	Query2	Query3	Query4	Query5
19	0.5	0.6	412.3	650.8	1,324.5
47	1.9	2.4	14,1673.7	21,866.5	81,480.4
89	9.9	10.1	141,550.5	241,580.6	1,016,290.4
145	18.9	27.5	818,920.7	1,652,650.2	-
218	60.0	76.6	-	-	-
301	97.5	131.3	-	-	-
371	126.3	153.1	-	-	-
460	231.7	517.3	-	-	-
563	335.7	487.7	-	-	-
671	351.0	501.7	-	-	-

can be instantiated within the query (these include quantified variables due to the way quantification is handled by GEOLOG).

Next, the time taken to evaluate the query and return all possible solutions for the given model and domain size must be considered. The results of these tests are shown in Table 6.3, where the evaluation of queries was again performed 100 times and an average time obtained. The results are similar to the those for a single solution, where the time increases as the domain increases. For Query3 – Query5, it is not possible to find all solutions once the domain grows too large. In Figure 6.6, two of the queries are, again, graphed to show the correlation between domain size and time required.

As a comparison, data-sets were output into Mace4, to test whether it was possible to evaluate the queries by building a model using that package. Because the queries contain two variables, if a model could be constructed the query would be represented by a 2-dimensional matrix, and it would thus be possible to determine which polygons satisfied the query quickly. The results of this are shown in Table 6.4. A time limit of 2000 seconds was placed, rather than allowing Mace4 to run continuously. This time limit was approximately the same as the internal constraint of SICStus, and also longer than the longest time recorded by GEOLOG to evaluate a query.

Within this time limit, Mace4 was unable to solve Query5 at all, and was also unable to find solutions for the fourth dataset onwards. As Mace4 is not designed to handle large domains and is also designed to be more general purpose than GEOLOG, which is

Table 6.4: A comparison of the time taken to evaluate the queries using model generation in Mace4, where, if a model can be constructed, there also exists a solution to the problem. The results are averages of 5 runs within Mace4 and are in milliseconds. Dashed lines represent a failure to find a result due to time or memory constraints. Mace4 was unable to solve for domains larger than 89, hence the lack of results beyond this size.

Number of Polygons	Query1	Query2	Query3	Query4	Query5
19	166.7	163.3	473.3	6,500.0	-
47	2,603.3	3,196.7	1,5303.3	-	-
89	56,566.7	63,773.3	224,330.0	-	-

specific to the given problem, the differences in performance are not surprising.

Thus, a model based approach is a suitable method of answering queries about the data, and a specialised system designed for this purpose is better suited to handling the problem than using a more general purpose program like Mace4. The key factors for the time required to solve the query have been shown to be the size of the domain, the number of variables within the query and the overall complexity of the query (such as the presence of universal quantification). Ways to specify the domain being used will now be considered, as well as how to limit and work with the domain.

6.3.6 Generating the Domain

The use of model checking approaches is reliant upon a finite domain. Section 6.3.4 discussed how this can be implemented logically, such that effective generator relations were defined logically to allow the generation of required polygons. As noted previously, the domain needs to be large enough to handle particular queries, ensuring all relevant polygons are available for a query. Ideally, the domain would not need to be restricted, but as already shown, the size can easily become too large to be able to use for computational or storage reasons.

To handle quantification through model checking requires searching through the entire model to answer queries; for example, with universal quantification it is required that the entire model is searched, to determine if a counter example exists to the predicate being quantified. Further, the increase in time required to handle queries as the domain size increases has been shown, even for simple queries. It is therefore not feasible to generate all permutations of polygons in advance, as it could render GEOLOG unusable for even simple queries.

6.3.6.1 Generating the Domain in Advance

A proposed solution, therefore, would be to generate the *required* sets of polygons beforehand, where no polygon is generated that could not be used in a potential query. This extends the notion that there will be plenty of polygon combinations that will never be generated, and, indeed, for queries based upon standpoints there may only exist a specific set of polygons that would ever be used. Thus, prior to using GEOLOG, all required effective generator relations would be ran across the set of allowable threshold values for standpoints, generating all possible polygons that may be used for potential queries. In some cases, these relations may be nested, relying on the results of other relations to be generated first. If this was the case, then the relations would need to be ran in the correct order, ensuring that each effective generator relation had all instances required to be evaluated correctly.

For example, if the linearity query is considered only, the minimum linearity value that could be used would be 1.0, since it does not fit into the given definition to use anything smaller (and would be an empty set). In fact, the minimum value may be slightly higher than this, depending upon the sensitivity of the data. Similarly, the largest value would be the value at which all larger measures produce the same results. This does not necessarily generate a single polygon representing all water within the data, since as discussed in 5.3.1, some points will never be marked as linear due to being too near the ends of the skeleton (either the point where the skeleton meets the land boundary, or points near where the skeleton is cut off as it extends into the sea). With upper and lower values specified, the program could be ran repeatedly and all possible polygons determined.

First, for each point the ratio between the radius and minimum and maximum radii would be calculated. Thus, for each line the minimum standpoint value required to make it be considered linear is now known. This will be the maximum of the two linearity values calculated. For example, if one end was calculated to be 1.2 and the other is 1.5 then that particular line will always be marked as linear providing the standpoint is greater than, or equal to, 1.5. From this, connected sets for all the possible threshold values could be calculated; thus, extending the previous example, 1.5 would be used as a threshold value, and the associated set of edges that are considered linear at this threshold would be found. Finally, the set of thresholds would be used to generate and store all the required polygons. Thus, when queries are evaluated, all possible polygons are available. An initial threshold is entered, and GEOLOG would look up which polygons satisfy that particular value. Therefore, the model is limited to only the polygons required.

A clear drawback to this approach, however, is the requirement of pre-computation. For every effective generator relation, it would be necessary to calculate all possible values, which could be particularly problematic for storage if nested relations are used. For example, if the above example was extended to include closeness, for every linearity threshold value, it would be necessary to determine the results of varying the closeness threshold. Further, although the domain would represent all possible instances that could occur, there may, in fact, only be a small set of thresholds that would ever be used, hence the domain is larger than required.

6.3.6.2 Generating a Fixed Point by Generating Polygons "Just in Time"

It therefore follows that an approach that allows the user to generate the polygons required "just in time" is used. As noted by Bennett et al. (2008), the generation of a complete domain of entities is impractical and unnecessary in practice, thus generating all required instances as a preliminary to query interpretation is preferable. This is performed by integrating model generation with query answering: First, confirm for a given query that all required polygons are generated. If this is not the case, then the appropriate effective generator relations are evaluated to generate these polygons. Once all required instances are generated, the model building and query answering stages detailed previously can be evaluated to find a solution.

A problem with this stage is the variability of the domain; if an effective generator relation needs to generate new polygons, the domains will vary before and after the query has been evaluated. The proposed solution is to repeat this phase until a *fixed-point* is reached, where no new polygons would be generated by evaluating the effective generator relations again. An overview of this process is as follows:

1. Evaluate all effective generator relations once and generate new polygons if required.
2. Construct the partial model M of the query and store as a set.
3. Re-evaluate the effective generator relations, to determine if a second iteration of them will result in new polygons.
4. Construct the partial model M' of the query and again store as a set.
5. If $M = M'$, then a fixed-point has been reached and the query can be evaluated. If not, M' replaces M and points 3-5 are repeated until the fixed-point is reached.

In order to reach this fixed-point, the effective generator relations used must complete within a finite number of steps. The previous requirement of functions generating maximal polygons (where maximal is relative to the measurements used in the function) should therefore help to achieve this goal. For example, consider the previously described predicates of linearity and closeness.

With linearity, the ratios derived between the widths at points will always be constant; hence, for a given threshold value, the set of linear edges will also be constant. Since linearity requires maximal polygons generated by finding maximal sets of connected edges, this only requires a single iteration to find all linear polygons. A second iteration of this relation would return the same results, hence a fixed-point is reached.

Recalling the closeness definition given in Section 5.3.3, a polygon is marked as ‘close-to’ two linear polygons iff it is connected to both and is close to both. This is performed using graph operations; given the graph of an expansive polygon, the largest connected sub-graph is found that is connected to the nodes shared with linear polygons, as well as satisfying the requirement that all nodes in this sub-graph can be traversed from the shared nodes using a distance less than or equal to the threshold used. As with the widths at each point, the distance between two nodes along the skeleton graph is static, hence for any given set of linearity and closeness thresholds, the results generated will be constant. Closeness therefore, reaches a fixed-point; either the closeness polygon exists already (the threshold is large enough that the polygon close to both linear stretches is equal to the expansive polygon between them) or can be generated to be maximal using the described algorithm.

As an alternative, an effective generator relation is now considered that would not reach a fixed-point (or rather, it would due to the granularity of the dataset but in actuality could continue for an infinite time if the dataset was infinite). This relation measures the imaginary property of $\mathsf{central}(P, P')$, where for a given polygon P, P' represents the region of P that is within a particular distance of the central point of P. Here, the central point of P is defined as the centre of the maximal inscribed circle of P. This is not the only way such a point could be defined but is sufficient for this example. To define the ‘central’ region of P, the threshold used is a percentage of the radius of the maximal inscribed circle of P, hence a circular region is defined. However, without additional constraints, this will clearly never reach a fixed point if threshold t is in the range $0\% < t < 100\%$.

An example of this is shown in Figure 6.7, with the outer region representing a polygon P. The threshold used here was 50% of the radius of the maximal inscribed circle, which for P generates the largest circle in Figure 6.7, here referred to as P'. If restrictions had been used upon the relation, such as limiting to only expansive polygons (such as P), then the relation would have reached a fixed point. However, in this example, no such restrictions were in place, allowing any polygon to be used as an input. Thus, P' could also be input into the central relation, generating the next largest circle in Figure 6.7. This would continue inwards, since for every polygon P, there would exist a polygon P' that satisfies the relation central(P, P').

Because a finite dataset is being used, a fixed-point would be reached due to the granularity of the data; a polygon would be generated that is a single point, which when entered into the relation would return the same point due to rounding the results. However, the number of polygons generated by this stage would have generated a domain too large to be manageable.

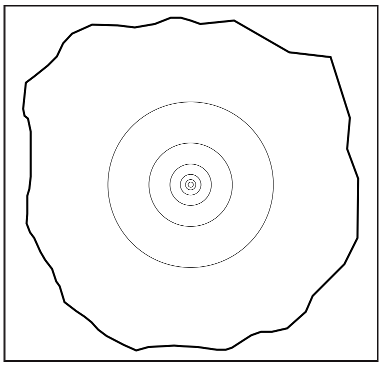


Figure 6.7: An example of an effective generator relation that does not have a fixed point. The relation generates a region that is ‘close’ to the centre of a polygon P, with ‘close’ defined as a percentage of the minimum distance to the edge of P from the centre. The example here used a value 50% of the width. The circles represent successive applications of the function, since each new region P will have a region P' that satisfies this.

Therefore, effective generator relations need to be designed to reach a fixed-point without relying upon the granularity of the data to achieve this. Typically, it would be expected for relations to reach this fixed-point after one iteration (as with linearity and closeness), though multiple iterations may be used. For example, suppose with the central relation, an additional restriction was added, stating that the radius of the maximal inscribed circle must be greater than or equal to a given distance. If this was used, the relation would go through multiple iterations generating successively smaller regions using the first threshold (percentage of radius), until the result would be a region smaller than the second threshold (minimum radius). This would be a fixed-point.

6.3.6.3 Implications for RCC of using Fixed-Point Approach

The next problem to arise from this approach, however, is the handling of RCC in relation to this fixed-point approach. For example, consider the following attribute for feature:

$$\textsf{feature}(y) \leftrightarrow \exists x[\textsf{water}(x) \wedge \textsf{PP}(y, x)] \tag{6.39}$$

If this was represented using an effective generator relation, $\mathsf{eff\_feature}(P, P')$ would take an input polygon P that satisfies the predicate water and return a polygon P' which is a proper part of P. However, when evaluated this relation would never reach a fixed-point (except due to the granularity of the data), since each iteration of the relation would generate a new polygon which would in turn have a proper part that could be returned in a later iteration of the relation. Whether these are collected one at a time (each iteration only finds a single proper part that has not been previously found until all are found) or in a single iteration (for each polygon P, find all proper parts), the result would be similar to the previously discussed $\mathsf{central}$ relation.

To avoid this problem, RCC relations are not able to generate new polygons alone; hence, queries involving RCC relations refer only to polygons that have already been generated within the model. For example, with Equation 6.39, it would not be suitable to allow the segmentation of new polygons, as the query would run until all proper parts of the base polygons have been generated.

Instead, GEOLOG would not use $\mathsf{feature}$ as an effective generator relation, but only evaluate over polygons already present in the dataset. Thus, no further polygons are generated by the query; the results of the query is a set of polygons already present in the domain that satisfy the predicate water and are proper parts of another polygon in the domain that also satisfies the predicate water. This restriction is a necessary trade-off in order to be able to work with a finite domain of data.

6.3.6.4 Conclusion

Because of this implementation, effective generator relations need to be designed in advance at present, as opposed to a user designing their own. This is to ensure that the underlying function in Prolog used by the effective generator relation will reach a fixed-point. This initially sounds as limiting as the previous suggestion of generating all in advance, but if a wide and useful range of such attributes is provided, this is not the case. Instead, the user is able to generate the polygon sets needed for their queries, and also only allow queries that the grounded ontology could handle.

This approach is the most suitable of the proposed solutions, as the domain will contain all regions necessary to evaluate the queries, which means the domain is restricted only to regions of interest. There are still limitations to this approach, such as the impact this has on negation and the use of RCC, but these are necessary trade-offs in order to be able to reason with a finite domain. As noted previously, the full domain is too large to be used by a model building approach, thus restricting the domain to as small as possible to be able to handle queries is the best approach.

6.4 Query Storage

The storage of the queries is important, as definitions may be dependent upon other queries within the domain. GEOLOG therefore needs a method of being able to look up definitions within the system and evaluate as required. In GEOLOG, queries are stored using an asserted fact:

$$\textsf{definition}(Name, Variables, FOL).$$

where $Name$ is the name of the query, $Variables$ are the set of variables to be returned by the query and FOL is the first order logic formula query which must be satisfied. For example, the previously given example query feature would be represented as:

$$\textsf{definition}(\textsf{feature}, [y], \exists x[\textsf{water}(x) \wedge \textsf{PP}(y, x)]).$$

This means the definition of feature is "any y which satisfies the query $\exists x[\textsf{water}(x) \wedge \textsf{PP}(y, x)]$". Logic operators would be stored as words (e.g. $\wedge$ would be replaced with and), but are left in here for clarity. Storing in this manner allows GEOLOG to look up definitions as they appear within queries being evaluated. For example, if a query referred to feature, GEOLOG could look up this definition and use it in the query, replacing occurrences of $\textsf{feature}(x)$ with its full definition, expanding the query. This would require ensuring that when expanding the query, new variables, not present in the rest of the query already, are introduced. Thus, expanding the query in this manner increases the number of variables to be satisfied, which could impact upon the query evaluation time. In addition, the results of feature would not be stored for future reference, meaning it needs to be evaluated every time it occurs within a query.

The proposed solution is to incorporate a memory style predicate to results, allowing them to be recalled later. This is similar to the approach used for RCC relations, where the RCC-8 relation was calculated and stored for each pair of polygons, ensuring the calculation was only performed once. For queries that label polygons as satisfying a particular definition, the process used is as follows:

1. Convert the query into an appropriate label: If the query does not use any thresholds, then the label associated with that query would just be its name, e.g. $\textsf{feature}(x) = \textsf{feature}$. If the query does contain thresholds, then incorporate into label e.g. $\textsf{linear}[l](x) = \textsf{linear}(l)$. This allows a polygon to be labelled as satisfying vague predicates at different thresholds.

2. Determine if query has been evaluated previously: when a query has been evaluated, GEOLOG asserts a fact stating this, such as $\mathsf{has\_run}(X)$, where X is the label generated previously.

3. If the query *has not* been evaluated previously: Evaluate the query, then for each polygon in the results, add the label to that polygon's set of attributes. Once this has been completed, assert that the query has been run.

4. If the query *has* been evaluated previously: Instead of re-evaluating, search the attribute sets of polygons, returning polygons that include the previously described label in their attribute set.

By using this approach, whenever GEOLOG finds a predicate within a query, it can look up whether it has been evaluated previously; if it has, then it can use the previously found results, otherwise it can evaluate the predicate as a query independently, feeding the results of this into the larger query. Complex queries can be built up using smaller queries, which, could in turn, be used in other queries also. This builds an ontological structure into GEOLOG and the resultant definitions. Although the approach only uses queries that have singular variables, it could be expanded to store the results of queries with different numbers of variables. For example, a query with no free variables (such as defining a proposition about the domain, like ”no region is land and water”), could be stored as a fact with the additional attribute of whether the query was true or false, allowing future evaluations to refer to this instead. With multiple free variables (such as relations between regions), a similar approach to RCC storage could be used, storing the set of polygons and the label that associates the relation between them.

6.5 Result Generation and Output

GEOLOG is now able to take first order logic queries as input and return sets of polygons that match these results. As noted previously, to solve a query, a model is first constructed of all possible values for each predicate, then the consistency of the query is tested against this model. If there exists a solution whereby all the free variables in the query can be instantiated to make the query true, then the query is considered to be satisfiable in the model. If there does not exist such an instantiation, then the query is considered false for the given domain and model.

A variety of potential outputs are now considered, including how this affects the handling of the query as well as issues with how these may be output. The main variations are in the number of free variables, and whether the query involves a spatial sum or not.

6.5.1 Queries with No Free Variables

When given a query with no free variables, then the output of results is not of interest. Rather, it is of interest whether the proposition represented by the query holds within the model. An example query would be to confirm that no region is both land and water:

$$\neg\exists x[\mathsf{water}(x) \wedge \mathsf{land}(x)] \tag{6.40}$$

Because GEOLOG is only testing for consistency here, a simple response declaring the query is true or false is sufficient for the output.

6.5.2 Queries with One or More Free Variables

Most queries of interest will contain only one free variable, as the aim will be to find and represent regions that have some particular properties. An example of this is Equation 6.41, where the aim is to find all linear regions that are connected to land:

$$\mathsf{linear}[l](x) \wedge \exists y[\mathsf{land}(y) \wedge C(x, y)] \tag{6.41}$$

For the output phase, there are several options. Firstly, GEOLOG can simply output a list of the polygons that match this query, for example if the aim was to just get an idea of the number of regions that satisfy the query. However, a more likely choice by a user, would be to represent the results graphically, to allow the results to be displayed upon a map where these regions are located. Because the points associated with a particular polygon are stored, these co-ordinates can be sent to an appropriate output for display. For example, tcl/tk (Welch, 2000) can be interfaced with Prolog, where tcl/tk acts as the graphical display of the polygons. Tcl/tk also allows the construction of an interface with GEOLOG, enabling a user to use GEOLOG without having to directly enter commands into Prolog.

However, this may not be suitable for all results, such as when they overlap or some are touching each other. In these situations, the user may instead wish to cycle through results one at a time or highlight certain results, to clarify the output. This may also be the case when there are multiple free variables. For example, suppose Equation 6.41 was expanded to return pairs of land and water regions that are connected:

$$\mathsf{water}(x) \wedge \mathsf{land}(y) \wedge \mathsf{C}(x, y) \tag{6.42}$$

The y variable now is also free, thus GEOLOG would return pairs of results. Displaying these results as a list of pairs is simple enough, but a graphical display is not as obvious, as, for example, a region may be part of multiple results or they may overlap. Outputting the results therefore in SVG[8] may be preferable. It would be possible to connect javascript code to the SVG file which would allow the user to determine how to display the output; for instance the user may be able to cycle through the result sets one at a time or 'mouse over' effects could be implemented to change the display depending upon the location of the cursor.

6.5.3 Queries with Spatial Sums

A final result that may be generated is that of spatial sums of regions, which are here represented as having the property of being self-connected, where a region is self-connected if it is not divided into a number of DC parts. These were defined previously in Equations 6.4 and 6.5.

To generate maximal self-connected polygons, an approach similar to a breadth-first search is used, marking neighbours of polygons as they are found. An example of this process is shown in Figure 6.8.

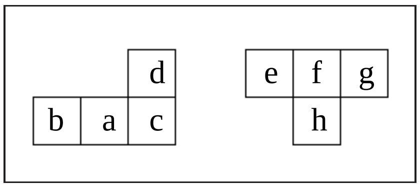


Figure 6.8: An example of how spatial sums are marked. Starting at *a* a breadth-first search returns the set *a,b,c,d*, and then finds polygons remain, repeating the search to get *e,f,g,h*. Depending upon requirements, single regions of these self-connected regions can be formed.

Generating the union of a set of polygons has been studied previously (Barton and Buchanan, 1980; Žalik, 2000), and can be reasonably efficient when combined with an appropriate storage format, such as the winged edge structure described in Section 4.4.3. A simple method, for example, is to pick a point that is known to be on the outside of the set, then trace around the edges, ensuring the path remains on the outside. All intersection points between the regions need to be calculated (as discussed in Section 5.4) to ensure it is known when to switch to a different edge. Further, using RCC it is possible remove any polygons in the set that are proper parts of another member of the set. Thus GEOLOG is already capable of generating new polygons as the maximal self-connected spatial sums of a set of polygons.

[8]SVG: http://www.w3.org/Graphics/SVG/ (Visited, August 2007)

The generation of new regions, however, will slow down other calculations and it may not be necessary to generate this larger polygon. Generally, if the sum of a particular set of polygons is not required for other queries, it is sufficient to just return the result as sets of regions which would generate maximal self-connected polygons if a spatial union operation was performed. In Figure 6.8 it may be sufficient to return the results as $[[a, b, c, d], [e, f, g, h]]$, rather than generating two new polygons. An example of this would be the spatial sum of inland water:

$$\text{inlandwater}(x) \leftrightarrow \text{CON}(x) \land \forall y[\text{P}(y, x) \rightarrow \text{water}(y)] \tag{6.43}$$

Because of downward inheritance, any further segmentation of water polygons will result in new polygons that also have the attribute of water attached. Thus, the answer to Equation 6.43 is always going to be equivalent to the set of all water polygons currently stored in GEOLOG. However, storing this in a single polygon would slow down other calculations, thus it is instead more efficient to return the results as a set (or just draw all the polygons on screen to represent the area the spatial sum would encompass).

6.6 Logical Queries Used

Section 5.3.1 determined some of the attributes that may be of interest to collect from data, such as linearity and closeness. It has also been shown in this chapter how such attributes could be extracted from the data in a logically consistent manner. This section will now determine the queries that are to be used upon the input data to test GEOLOG, including considering what attributes are available to the user and how these would integrate with an upper level geographic ontology.

6.6.1 Basic Features

As noted in Section 6.3.3, the data is first split into basic matter types of land, sea or water:

$$\text{land}(x) \tag{6.44}$$

$$\text{sea}(x) \tag{6.45}$$

$$\text{water}(x) \tag{6.46}$$

As previously noted, the land and sea polygons will remain static within the system, and no further segmentations will be generated; because only inland water features are to be considered, both land and sea are represented by a single polygon without a skeleton, hence determining segments of this would not be possible. New segmentations of water polygons on the other hand, can occur whenever thresholds of vague predicates are changed (and thus the standpoint is changed) or spatial sums are used. This initial set of water polygons could either consist of a single large polygon, or as a 'base' set of smaller polygons, the sum of which is equivalent to the larger polygon.

6.6.2 Basic Segmented Features

The next set of predicates of interest are those generated using effective generator relations and are calculated by GEOLOG. These are the predicates which segment the data into regions of interest, and hence will form the basis of the more involved queries. The principal vague predicate used is for the linearity segmentation:

$$\mathsf{linear}[l](x) \tag{6.47}$$

$$\mathsf{expansive}[l](x) \tag{6.48}$$

Here, $\mathsf{linear}[l](x)$ represents a maximal self-connected linear region relative to standpoint l (as discussed in Section 5.3.1, whereas $\mathsf{expansive}[l](x)$ represents the remaining maximal self-connected regions not marked as linear (as discussed in Section 5.3.2). The threshold is the allowable variation in widths along the skeleton of that region, with higher values allowing more variation in the widths for a region to be marked as linear. As noted in Section 5.3.2, this threshold impacts both linearity and expansiveness, since every skeleton edge is marked as being linear or expansive depending upon the threshold.

As noted in Section 5.3.1, there are alternative methods of marking linearity, since as was shown previously in Figure 5.13, this form of linearity measurement can result in regions marked as linear that were not intended to be marked as such by the user. An alternative measurement was proposed that considered the lengths of the edges of the previously generated polygons in relation to their skeleton. To clarify, for each linear polygon generated previously, there will be a skeleton associated with that polygon, and two edges or 'banks' either side of this skeleton. The lengths of these banks and the skeleton are compared, and if the difference between these is below a threshold, they are marked as 'linear with respect to the edges':

$$\mathsf{linear\_WRT\_edge}[e, l](x) \tag{6.49}$$

$$\mathsf{expansive\_WRT\_edge}[e, l](x) \tag{6.50}$$

Thus, $\mathsf{linear\_WRT\_edge}[e,l](x)$ represents a maximal self-connected region that is linear both with respect to its centre (threshold l) and its edges (threshold e). In a similar fashion $\mathsf{expansive\_WRT\_edge}[e,l](x)$ represents a maximal self-connect region that is not linear with respect to both its centre and its edges. The standpoint threshold e is the allowable difference in the length of the polygon's skeleton in relation to the length of the banks of the polygon. Thus, a higher threshold means the difference in the lengths of the skeleton and the banks is allowed to be larger. For the purposes of this thesis, these predicates are only used to show the difference in linearity segmentation and how there are different possible interpretations of 'linear'; otherwise the original predicate of $\mathsf{linear}[l](x)$ is used.

As discussed in Section 5.3.3, limitations in the linearity segmentation led to the requirement of an artificial 'gap' feature which has been labelled *interstretch*. The basis of this is the vague predicate 'close-to':

$$\mathsf{close\text{-}to}[c,l](x) \tag{6.51}$$

The predicate is dependent upon two standpoint thresholds; the linearity threshold l which generates the linearity segmentation, and a closeness threshold c that determines how close a region needs to be to two linear regions to be marked as 'close-to'. The threshold c relates to the actual distance in kilometres; a region is close to two linear regions if every point on the region's skeleton can be reached from the linear polygons in a distance less than, or equal to, c kilometres.

6.6.3 Queries

Now that the semantics of the basic predicates have been defined, they can be used to define further, higher-level predicates, which can, in turn, be fed into further queries. The first definition to consider is stretch:

$$\begin{aligned}\mathsf{stretch}[l](x) \leftrightarrow \mathsf{CON}(x) \wedge \mathsf{water}(x) \wedge \mathsf{linear}[l](x) \wedge \\ \forall y \big[[\mathsf{CON}(y) \wedge \mathsf{water}(y) \wedge \mathsf{linear}[l](y) \wedge \mathsf{P}(x,y)] \rightarrow \mathsf{EQ}(x,y)\big]\end{aligned} \tag{6.52}$$

where a region is a stretch (relative to linearity threshold *l*) if it is a maximal self-connected region of water that is linear. However, when using the system, it is only necessary to refer to $\mathsf{stretch}[l](x)$, as a polygon marked as linear inherently contains the property of being maximal and self-connected as discussed previously.

For comparison of linearity segmentations, the intention is to also show the difference when using the linearity with respect to the edges measurement:

$$\begin{aligned}\textsf{stretch\_WRT\_edge}[e,l](x) \leftrightarrow\ & \textsf{stretch}[l](x) \wedge \textsf{linear\_WRT\_edge}[e,l](x)\wedge \\ & \forall y\big[[\textsf{CON}(y) \wedge \textsf{water}(y)\wedge \\ & \textsf{linear\_WRT\_edge}[e,l](y) \wedge \textsf{P}(x,y)] \rightarrow \\ & \textsf{EQ}(x,y)\big]\end{aligned} \tag{6.53}$$

As already noted, this is to show the difference in linearity interpretations only, and will not be used in other definitions.

The next defined predicate is *interstretch*, to account for the insignificant gaps that can occur between stretches that the user may wish to fill in. To define maximality, it is easier to break the query into smaller parts, first defining regions that are 'close-to' two linear regions:

$$\begin{aligned}\textsf{interstretch\#}[c,l](x,y,z) \leftrightarrow\ & \textsf{water}(z) \wedge \textsf{CON}(z)\wedge \\ & \exists w[\textsf{expansive}[l](w) \wedge (\textsf{TPP}(z,w) \vee \textsf{EQ}(z,w))]\wedge \\ & \textsf{stretch}[l](x) \wedge \textsf{stretch}[l](y) \wedge \neg(x=y)\wedge \\ & \textsf{EC}(x,z) \wedge \textsf{EC}(y,z) \wedge \forall v[\textsf{P}(v,z) \rightarrow \\ & (\textsf{close-to}[c](v,x) \wedge \textsf{close-to}[c](v,y))]\end{aligned} \tag{6.54}$$

Thus, interstretch# defines a region of water that is a proper part of (or equal to) an expansive region, which is externally connected to two different linear stretches and all parts are 'close-to' both linear stretches. This can thus be used to define *interstretch* as a maximal form of interstretch#:

$$\begin{aligned}\textsf{interstretch}[c,l](x) \leftrightarrow \exists y,z\big[&\textsf{interstretch\#}[c,l](y,z,x) \wedge \forall w[\textsf{CON}(w) \wedge \textsf{water}(w)\wedge \\ & \textsf{interstretch\#}[c,l](y,z,w) \wedge \textsf{P}(x,w) \rightarrow \textsf{EQ}(x,w)]\big]\end{aligned} \tag{6.55}$$

An *interstretch*, therefore, is a maximal interstretch# region.

Now that stretches and *interstretches* have been defined, other key features that are needed to individuate the main channels must to be determined: The occurrence of islands needs consideration, since these will cause problems for the linearity measurement. The occurrence of junctions must also be considered. An island is defined as:

$$\textsf{island}(x) \leftrightarrow \textsf{land}(x) \wedge \forall y\big[\textsf{EC}(x,y) \rightarrow \exists z[\textsf{P}(z,y) \wedge \textsf{water}(z)]\big] \tag{6.56}$$

The intention is to define an island as a region of land that is surrounded by water. The above definition attempts to capture this by defining an island as land that is only connected to regions that have a part that is water. This definition may not be ideal, as it is dependent upon restrictions in place within the domain; for example, as noted in Section 6.3.6.3, the usage of RCC within GEOLOG is restricted to returning polygons that exist already within the domain. In addition, as noted in Section 6.3.3, regions are defined to only consist of a single matter type; if GEOLOG allowed regions to be arbitrary and formed of more than one matter type, it could be possible to perceive a region that would negate the definition. Within this context, however, the definition is sufficient. A more thorough definition would include a stricter definition of 'surrounded', for example, relating this to the edges of the region.

As discussed in Section 5.3.4, the presence of islands can be problematic for linearity measurements, hence the development of an 'island-water' predicate was proposed, to identify regions surrounding islands. Incorporating the vague ratio threshold (as proposed to determine whether to consider the water surrounding an island as separate parts or as a single region) is not straightforward, as there may be more than one island included. It would thus be necessary to represent logically, that the predicate is comparing the area of the outer polygon with the sum of the areas of the islands contained within, and therefore it has been omitted in this thesis.

As was performed for *interstretch*, it is easier to separate the definition into different parts, first defining water that is connected to some island:

$$\begin{aligned}\mathsf{islandwater\#}[l](x) \leftrightarrow \mathsf{CON}(x) \wedge \mathsf{water}(x) \wedge \exists w \Big[& \mathsf{island}(w) \wedge \mathsf{EC}(w,x) \wedge \\ & \forall y \big[\mathsf{P}(y,x) \rightarrow \mathsf{water}(y) \wedge \exists z[(\mathsf{linear}[l](z) \vee \mathsf{expansive}[l](z)) \wedge \\ & \mathsf{EC}(w,z) \wedge \mathsf{O}(y,z) \wedge \mathsf{PP}(z,x)]\big]\Big]\end{aligned} \tag{6.57}$$

Thus, islandwater# is a self-connected region of water that is externally connected to some island[9] and all parts of the region are water that overlap either a linear or expansive region, which is also externally connected to the island. This allows the region to have both linear and expansive parts, whilst limiting to a region that surrounds the island. Because the presence of the island will result in junctions in the skeleton, expansive regions

[9]The reason that externally connected is used as opposed to proper part for the island to water relation, is because the island actually occupies a hole of the same size within the water polygon. Thus, the island is externally connected to the water surrounding it, as opposed to being contained within it.

would be expected to occur at the point where the skeleton forks around the island. This requirement, therefore, restricts the region to the channels around the island and the junctions where the inland water network forks around the island. From this, 'island-water' can be defined as:

$$\begin{aligned}\mathsf{islandwater}[l](x) \leftrightarrow \mathsf{islandwater\#}[l](x) \wedge \forall y[\mathsf{CON}(y) \wedge \mathsf{water}(y) \wedge \\ \mathsf{islandwater\#}[l](y) \wedge \mathsf{P}(x,y) \rightarrow \mathsf{EQ}(x,y)]\end{aligned} \tag{6.58}$$

Thus, 'island-water' is a maximal islandwater# region.

The next feature to consider is the junctions between multiple stretches. The junction of two or more rivers is referred to as a 'confluence'; hence, determining a region that may correspond to this would be useful for the individuation of rivers within an inland water network. An additional requirement is that this region is not touching an island; were islands not present within the dataset, then the junctions formed as the skeleton traverses around the islands would also not be present, hence such junctions are not considered confluences here[10]:

$$\begin{aligned}\mathsf{confluence}[l](x) \leftrightarrow \mathsf{expansive}[l](x) \wedge & \\ & \exists w,y,z[\mathsf{stretch}[l](w) \wedge \mathsf{stretch}[l](y) \wedge \mathsf{stretch}[l](z) \wedge \\ & \mathsf{DC}(w,y) \wedge \mathsf{DC}(w,z) \wedge \mathsf{DC}(y,z) \wedge \\ & \mathsf{EC}(x,w) \wedge \mathsf{EC}(x,y) \wedge \mathsf{EC}(x,z)] \wedge \\ & \neg\exists i[\mathsf{island}(i) \wedge \mathsf{C}(x,i)]\end{aligned} \tag{6.59}$$

A confluence is an expansive region of water that is connected to at least three different stretches of water (that are disconnected from each other), but not to an island.

The necessary features to define a 'major stretch' are now available, which will be the spatial connected sums of stretches, *interstretches* and island-waters. However, this will

[10]This may not always be the case, particularly when an island is very large. One potential solution would be to mark land regions as islands in relation to the surrounding water region. Thus, the island definition given previously would determine a candidate for an island, then 'island-water' would determine whether it is considered as an island or not. As no standpoint threshold is associated with 'island-water', this is beyond the scope of this thesis.

not include confluences, which will instead be left out, thus individuating channels:

$$\begin{aligned}\mathsf{majorstretch\#}[c,l](x) \leftrightarrow \mathsf{CON}(x) \wedge \mathsf{water}(x) \wedge \forall w \Big[\mathsf{PP}(w,x) \rightarrow \\ \exists z \big[\mathsf{CON}(z) \wedge [\mathsf{stretch}[l](z) \vee \mathsf{interstretch}[c,l](z) \vee \\ \mathsf{islandwater}[l](z)] \wedge \mathsf{O}(w,z) \wedge \mathsf{PP}(z,x)\big] \wedge \\ \neg\exists h[\mathsf{confluence}[l](h) \wedge \mathsf{O}(w,h)]\Big]\end{aligned} \tag{6.60}$$

Thus, majorstretch# is a self-connected region of water, where all proper parts are overlapping a stretch, *interstretch* or ‘island-water’ region, and no part is overlapping a confluence. Major stretch, therefore, is the maximal form of this:

$$\begin{aligned}\mathsf{majorstretch}[c,l](x) \leftrightarrow \mathsf{majorstretch\#}[c,l](x) \wedge \\ \forall y[\mathsf{majorstretch\#}[c,l](y) \wedge \mathsf{P}(x,y) \rightarrow \mathsf{EQ}(x,y)]\end{aligned} \tag{6.61}$$

A major stretch is therefore a maximal form of majorstretch#.

6.6.4 Conclusions and Potential Expansions

6.6.4.1 Additional Features

This section has defined a set of queries that could be used to generate features of interest, which in turn would feed into higher level queries. These are not the only features that could be generated using the vague predicate approach. For example, other features that may be collected occur at the extremities of the network, where the skeleton meets a land boundary or the sea. Because of the method of measuring linearity used, there will be regions at the ends of the skeleton where the skeleton meets the boundary that are never marked as linear at any threshold value. Because these may correspond to the source of rivers, it may be desirable to mark them, which could be identified logically (Third et al., 2007).

The other extremity of the skeleton is where it meets the sea, and it may be desirable to mark the region that corresponds to the mouth of the river (or an estuary). Linearity may not be a required attribute of an estuary, but it is a vague feature and requires predicates to handle this vagueness and segment appropriate regions to correspond to an estuary. For example, one possible definition would use the previously defined closeness threshold in relation to the sea; thus, the river mouth is a region that is connected to and close to the sea. Further work is required to determine how to define this region in a suitable manner.

6.6.4.2 Integration with Higher-Level Ontologies

The queries described in this section could be used to define features of interest, which, in turn, could be used to form definitions of features such as ‘river’ or ‘lake’. Although a direct link to these primitives is not defined here, this could be achieved by incorporating in some form of ‘size’ attribute (to show the split between ‘river’ and ‘stream’ for example). Example potential primitive definitions would be:

$$\begin{aligned}\mathsf{river}[c,l,s_1](x) \leftrightarrow\ & \mathsf{CON}(x) \wedge \mathsf{water}(x) \wedge \forall w[\mathsf{PP}(w,x) \rightarrow \\ & \exists p[\mathsf{P}(w,p) \wedge \mathsf{TPP}(p,x) \wedge \\ & (\mathsf{riversource}[l](p) \vee \mathsf{majorstretch}[c,l](p))] \wedge \\ & \mathsf{large\_linear}[s_1](x)\end{aligned} \tag{6.62}$$

$$\begin{aligned}\mathsf{lake}[c,l,s_2](x) \leftrightarrow\ & \mathsf{CON}(x) \wedge \mathsf{water}(x) \wedge \mathsf{expansive}[l](x) \wedge \\ & \neg\exists w,y[\mathsf{stretch}[l](w) \wedge \mathsf{stretch}[l](y) \wedge \mathsf{interstretch}[c,l](w,y,x)] \\ & \wedge \mathsf{large\_expansive}[s_2](x)\end{aligned} \tag{6.63}$$

Here, to disambiguate ‘large’, the approach of having a specific predicate for each has been used, rather than a general predicate of ‘large’ and allowing different contexts as discussed by Bennett (2006). However, the above definitions are only illustrative.

The use of such primitives would allow the results of GEOLOG to be interfaced with a higher-level ontology, thus completing the architecture described previously, where GEOLOG acts as a middle level grounding the ontology level upon the data level. Thus, instead of primitives such as ‘river’ or ‘lake’ referring directly to the data level, they would instead refer to the grounding level to determine a more detailed definition, which would generate the required regions within the data.

In addition, the structure of GEOLOG allows for an ontology to be inherently stored, based upon the query structure described in Section 6.4. For example, the queries developed here to test the datasets could be taken to represent a simple ontology, starting from the basic predicates and moving downward to the more complex queries such as major stretch.

6.7 Summary

This chapter has considered how to ground an ontology upon the data level. It was shown what is meant by grounding in this instance, and how the choice of logic is crucial to handling this grounding correctly. It was shown how queries can be handled using model checking approaches, as well as how problems relating to the domain can be addressed to allow this approach to perform correctly. Issues related to displaying the results both in list and graphical forms were discussed. The final output of GEOLOG is not finalised here, as it is partly dependent upon the requirements of the user. Instead, the output is provided in a format compatible with GIS. Finally, a set of definitions were developed that can be used to evaluate over the test datasets. These are intended to represent predicates that would be of interest to a GIS user, returning segmented regions corresponding to vague features.

Chapter 7

Results of Using the System with Topographic Data

7.1 Introduction

This chapter will examine the results obtained when topographic data is input into GEOLOG, from the data representation stage through to the final output. The values to be used for thresholds are given in Section 7.2. The evaluation criteria for GEOLOG will be discussed in Section 7.3, to establish how the success of the approach is to be measured. Section 7.4 will investigate the results of applying these queries to the first dataset, the Humber Estuary in UK, using the criteria set out in Section 7.3. The next datasets that are considered are the River Tyne and the Stour-Orwell Estuary (both also in the UK), which are examined in Sections 7.5 and 7.6 respectively. An overall analysis of these results is given in Section 7.7, looking at the success as a whole of using GEOLOG to generate features of interest, using the criteria in Section 7.3. The chapter is summarised in Section 7.8.

7.2 Threshold Values Used

The definitions to be used were previously described in Section 6.6. These will be tested over all three datasets, varying the standpoint thresholds for the attributes defined previ-

Table 7.1: The threshold values for which the predicates will be run. In instances where two thresholds are used, one will be kept constant and the other varied. This is highlighted when required below.

Predicate	Threshold Values
stretch$[l]$	$l = \{1.2, 1.3, 1.4\}$
stretch_WRT_edge$[e, l]$	$l = \{1.3\}$ $e = \{1.2, 1.3, 1.4\}$
interstretch$[c, l]$	$l = \{1.3\}$ $c = \{1.0, 3.0, 5.0\}$
islandwater$[l]$	$l = \{1.2, 1.3, 1.4\}$
confluence$[l]$	$l = \{1.2, 1.3, 1.4\}$
majorstretch$[c, l]$	$c = \{1.0\}$ $l = \{1.2, 1.3, 1.4\}$

ously in Section 5.3, to determine the impact this has on the results. For testing purposes, the same set of threshold values will be used for each dataset, to compare the results obtained in each case. These are listed in Table 7.1. Speed tests have previously been run, thus the time taken to compute results is not of concern here. Rather, the results themselves are of interest, and how close they correspond to features of interest.

7.3 Evaluation Process

The evaluation process should be based on objective criteria, to ensure it is clear what would be considered successful results of using GEOLOG. The intention is to produce, in advance, a set of ‘expected’ results for the queries, using criteria set out in advance, then compare the generated results against these.

The evaluation of GEOLOG is based upon the observation made in Section 5.3.1 that linear regions may approximate to a series of overlapping rectangles. Thus, to evaluate the effectiveness of the linearity predicate on a particular dataset (both with respect to the centre and to the edges respectively), a series of rectangles is overlaid on the dataset, to identify the regions that would be expected to be marked as linear stretches. When two rectangles overlap, they are assumed to be part of the same linear stretch (to allow for the stretch curving), hence the linearity predicate should mark them as being part of the same stretch.

The evaluation of the *interstretch* predicate is also related to the overlaid rectangles, filling in ‘gaps’ that occur between linear stretches that the overlaid rectangles approach identified as being expected to be part of the same region. The union of the linear stretch and interstretch regions, therefore, should match the regions overlaid by rectangles as closely as possible.

The land regions expected to be marked as islands are identified prior to testing, and are used along with the results of the overlaid rectangles to identify the expected island-water and confluence regions. For island-water, this will be the region surrounding the island and demarcated by rectangles representing linear stretches that are close to but do not touch the island in question. Similarly, any expected confluence regions are identified by finding regions that are not overlaid with rectangles and touch three different rectangles.

The final evaluation is the spatial sum of the linear, *interstretch* and island-water regions identified above, which is the expected results of the major stretch predicate. This is determined by marking out the regions overlaid by the rectangles, and combining these with the island-water regions, using a spatial sum.

7.4 The Humber Estuary

The first dataset considered was that of the Humber Estuary, on the East Coast of England. The main portion of the dataset is that of the River Humber, which is formed by the confluence of the River Ouse and the River Trent. At the mouth of the river is the easily recognisable feature known as Spurn Head, which is a long thin strip of land that forms the bay-like feature towards the mouth of the river.

7.4.1 Humber Estuary - Initial Data Input

As previously noted, the initial input was obtained from data obtained from the Global Landcover Facility (GLCF)[1], which was vectorised for input into the system. Once the data was input into the system, the Medial Axis and simplified skeleton were determined, with the result of this shown in Figure 7.1a. On the dataset, the River Ouse is the river flowing from the far left of Figure 7.1a, whereas the Trent flows from the bottom of Figure 7.1a. From this skeleton, it is possible to construct the associated polygon representing all inland water, as shown in Figure 7.1b.

The vague predicates are now used to segment the data into more useful segments, based upon geometric observations. For the results, smaller figures are given within each section of the results, with larger figures of the results given in Appendix A, to highlight the differences in more detail.

[1]Landsat ETM+ imagery: http://glcfapp.umiacs.umd.ed:8080/esdi/index.jsp (Visited, July 2006)

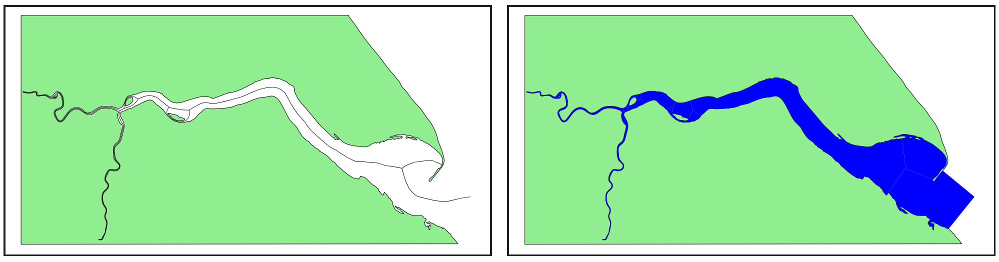

(a) (b)

Figure 7.1: The results of determining the skeleton of the Humber Estuary from the Medial Axis (Figure 7.1a), together with polygon generated from this skeleton (Figure 7.1b), which represents the extent of the dataset that can be considered inland water.

7.4.2 Humber Estuary - Expected results

The evaluation process is performeed prior to the queries being run, to determine what the expected results are. Figure 7.2 shows the results of overlaying the Humber Estuary dataset with rectangles, to determine the regions expected to be marked as linear. The River Trent is expected to be marked as a single linear stretch, whereas the River Ouse is expected to be marked as two seperate stretches, disconnected due to the island (with small linear stretches surrounding the island). For the River Humber, the islands mean that the expected results will be a series of linear stretches disconnected due to the islands. The Humber linear stretches are expected to stop at the bay-like region that connects the Humber to the sea. This Figure is also the basis of the evaluation for the *interstretch* query, as that query should be able to identify 'gaps' that split linear stretches that are expected to be marked as part of the same stretch according to Figure 7.2.

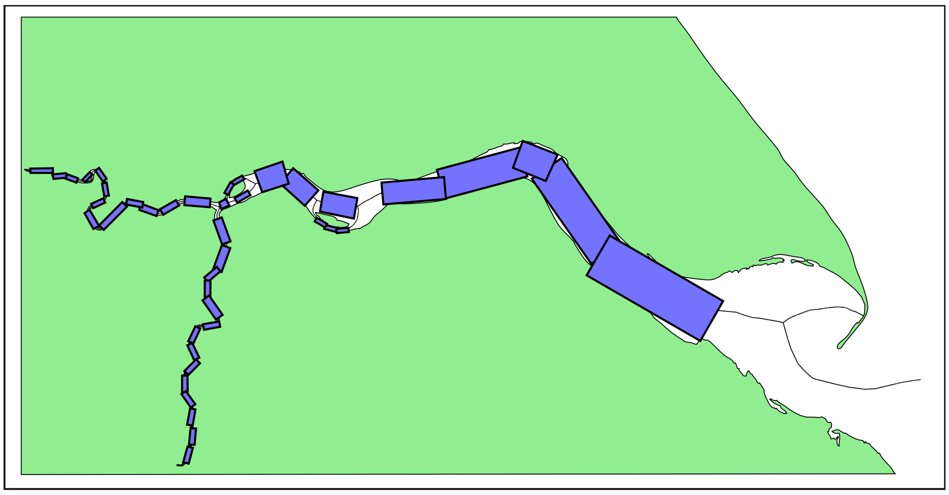


Figure 7.2: The expected results of performing the linearity sgemention on the Humber Estuary.

Figure 7.3 shows the expected results of evaluating the island, island-water and confluence queries on the Humber Estuary dataset. These follow on from the results of Figure 7.2, since the linear regions will form the boundaries to the island-water and confluence regions. Therefore, the ends of the rectangles that are expected to form these boundaries are represented by blue regions in the Figure. An island-water region is expected to form around each of the islands, whereas only one confluence region is expected, at the junction of the three rivers.

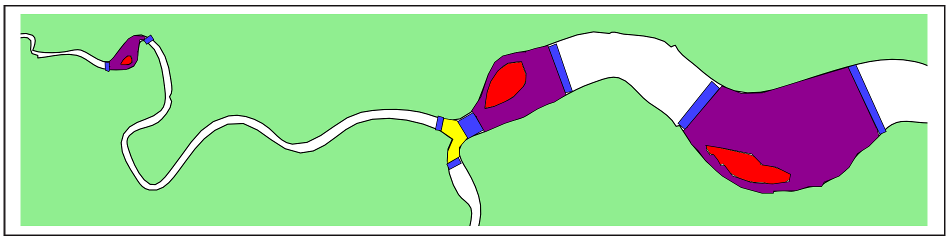


Figure 7.3: The expected results of evaluating the island, island-water and confluence queries on the Humber Estuary. The blue regions represent the ends of the overlaid rectangles that form the boundaries of the island-water and confluence regions. The red regions are the expected islands. The purple regions are the expected island-water regions. The yellow region is the expected confluence region.

Finally, Figure 7.4 shows the expected results of the major stretch query, as a culmination of the previous stages. The regions overlaid by rectangles in Figure 7.2 have been marked out within the data and combined via a spatial sum with the island-water regions in Figure 7.3. Thus, each of the three rivers is expected to be marked as a single stretch respectively, seperated by the confluence region previously identified.

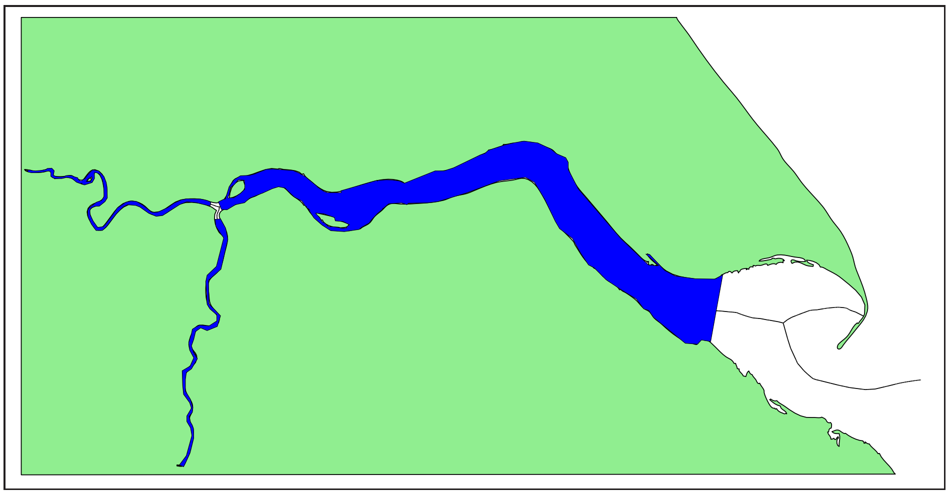


Figure 7.4: The expected results of evaluating the major stretch query on the Humber Estuary. The blue regions represent the regions expected to be marked as major stretches.

7.4.3 Humber Estuary - Linearity Segmentation

The results of applying the previously described approach to segmenting the data into linear regions for two of the thresholds (1.2 and 1.4) are shown in Figure 7.5, with larger figures given for all threshold values in Figures A.1 - A.3 in Appendix A. The linear sections are marked in blue, and, as would be expected, the higher (and thus more relaxed) the threshold is for linearity, the larger the regions that are marked as being linear.

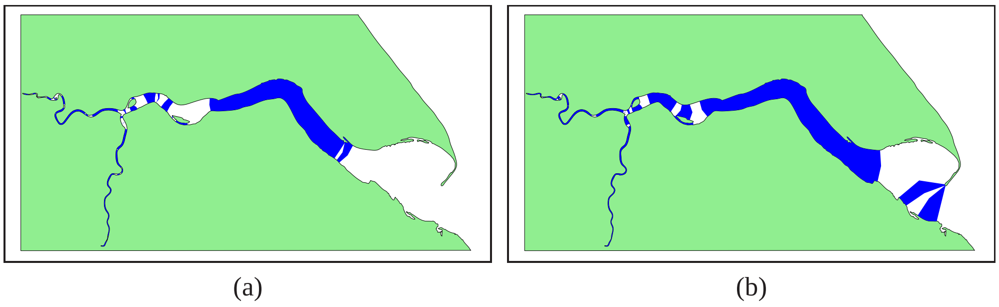


Figure 7.5: The results of performing the linearity segmentation on the Humber Estuary using thresholds of 1.2 and 1.4.

When compared to the expected results in Figure 7.2, the query is most successful at marking the River Ouse and River Trent as linear stretches. In addition, at higher thresholds, the query identifies linear stretches surrounding each of the islands respectively, which more closely matches the expected results. However, there are anomalous results, such as small strips not being marked as linear, generating gaps between stretches that were expected to be part of the same stretch in Figure 7.2. In addition, at higher thresholds, there are linear stretches marked around Spurn Head, which again are not expected.

Both of these anomalies are related to the way linearity is determined with respect to the centre: First, the small non-linear parts are caused when there is a slight widening of the stretch at that point. This results in a longer segment of the skeleton being considered; hence, there is an increased likelihood of the inclusion of a point in the calculation, whose width is too large or small for the point under consideration to be marked as linear. Because the stretch then narrows again, the length of the skeleton considered decreases, reducing the likelihood of variation. Thus, there are results which may be considered ‘anomalies’ at lower threshold values (with higher thresholds reducing the occurrence). With Spurn Head, the problem is that the skeleton is curving around Spurn Head, thus the width is actually varying little, resulting in the linear parts. In Figure A.3 there are two separate linear parts at Spurn Head, due to the inlet on the south bank, which causes the width to increase sharply at that point.

The basic linear segmentation predicate is capable of producing a segmentation of the network into linear stretches that quite closely matches the expected results from Section 7.4.2, with the exceptions of the regions at Spurn Head and some of the gaps between stretches.

7.4.4 Humber Estuary - Linearity with Respect to Edges

The refined linearity measurement discussed on Page 78 is now applied, where a region has to be both linear with respect to the centre and its edges. The results for two of the thresholds (1.2 and 1.4) are shown in Figure 7.6, with larger results of this stage for all thresholds shown in Figures A.4 - A.6. The linear stretch at Spurn Head has now been removed, as would be expected. However, other parts are also removed, and at lower threshold values rather large portions are removed. Again, at higher thresholds more parts remain, whilst still removing anomalous results like Spurn Head.

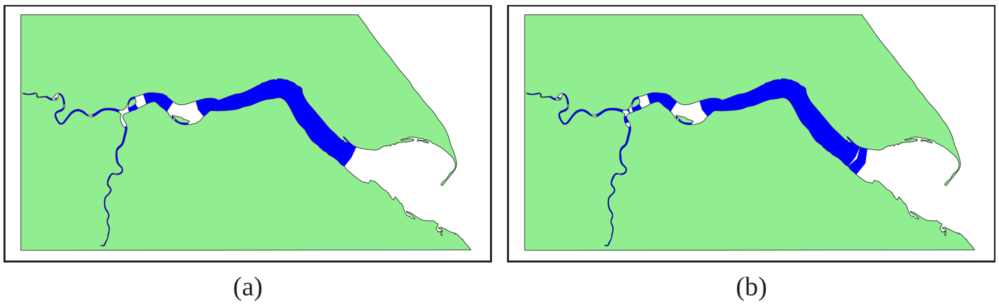


Figure 7.6: The results of performing the linearity segmentation with respect to the edges on the Humber Estuary using thresholds of 1.2 and 1.4.

The refined query was capable of removing the region marked as a linear stretch at Spurn Head, thus more closely matching the expected results discussed in Section 7.4.2.

7.4.5 Humber Estuary - Interstretches

The next query used is to determine the effectiveness of marking *interstretches*, to fill in ‘gaps’ that occur within the data. The results of marking *interstretches* are shown in Figure 7.7 using closeness threshold values 1.0 and 5.0, with larger results shown in Figures A.7 and A.8. Linear regions marked in blue and *interstretches* marked in red.

The results using a threshold of 1.0 and 3.0 were the same, with small gaps filled in along most of the network. At the higher threshold value of 5.0, the large bay-like region is also included, thus showing that as the threshold is increased, larger regions are

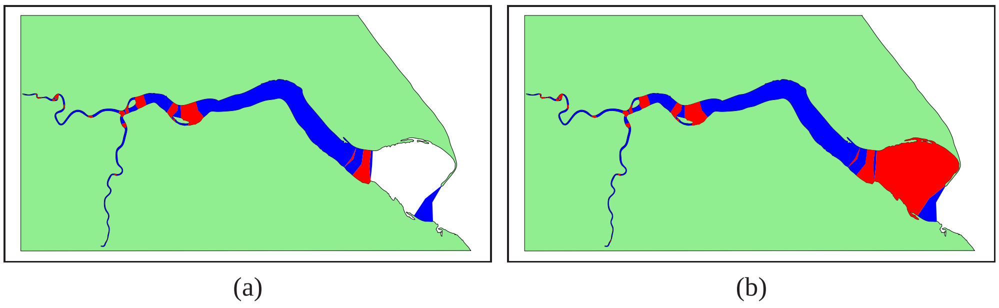


Figure 7.7: The results of marking interstretches on the Humber Estuary using closeness thresholds of 1.0 and 5.0.

identified as *interstretches*. The intention is that the *interstretch* predicate will identify small ‘gaps’ between linear stretches, to allow the stretches to be joined using spatial sums. Thus, the results of this predicate when combined with the linear stretches marked previously should more closely match Figure 7.2.

At lower thresholds, the query is capable of identifying such ‘gaps’ as *interstretches*, though at higher stretches the bay-like region is marked as an *interstretch* also. In addition, the junctions formed at the islands and between the meeting of the three rivers are also marked as *interstretches*, meaning the query marks too many regions as ‘gaps’. If the spatial sum of linear stretches and *interstretches* was taken at this stage, the result would be a single large stretch encompassing all three rivers, which is not the expected result in Figure 7.2. Therefore, whilst the *interstretch* query is able to identify ‘gaps’ between stretches to fill in, it also marks other regions, requiring further queries to refine the results.

7.4.6 Humber Estuary - Island and Island-Water

The determining of islands and corresponding island-water regions are the next queries to be considered. The results of running these queries for threshold values 1.2 and 1.4 are shown in Figure 7.8, with larger results for all thresholds shown in Figures A.9 - A.11. Islands are marked in red and island-water regions marked in blue. The same islands

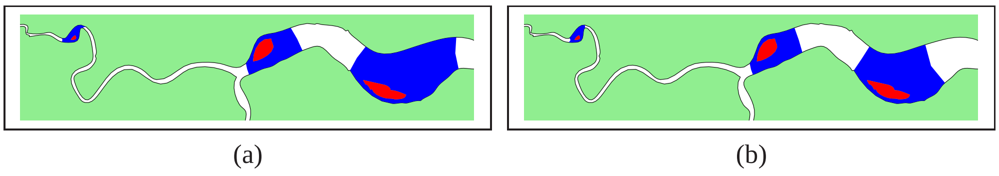


Figure 7.8: The results of marking islands and island-water regions on the Humber Estuary using thresholds of 1.2 and 1.4.

are marked in all three, as this is not dependent upon the standpoint taken. However, the regions marked as island-water shrink as the threshold is increased. This is because the linear stretches to either side of the island-water regions increases due to the relaxed threshold, hence the regions touching the island are reduced. The results of these queries very closely match the expected results in Figure 7.3, with the same three island-water regions identifed, as well as these regions being quite close to the expected size. This is most likely due to the linearity segmentation successfully marking the stretches that form the boundaries to the island-water regions.

7.4.7 Humber Estuary - Confluence

The results of determining confluences are shown in Figure A.12, with the only confluence marked in the data shown. Similar to the marking of island-water, the confluence shrinks as the linearity threshold increase, since the linear regions surrounding it are extended at higher thresholds.

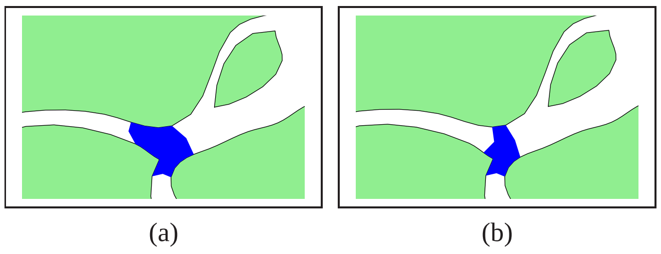


Figure 7.9: The results of marking confluences on the Humber Estuary using linearity thresholds 1.2 and 1.4.

When compared to Figure 7.3, the results of this query are again successful, as only one confluence region was identified, although the expected region was larger than the actual results generated.

7.4.8 Humber Estuary - Major Stretch

The results of determining major stretches for linearity thresholds 1.2 and 1.4 are shown in Figure 7.10, with larger results for all thresholds shown in Figures A.13 - A.15. Major stretches are marked in blue.

The combination of the queries most closely matches the expected result given in Figure 7.4 at the lowest threshold, when there are no linear stretches marked at Spurn Head. At higher thresholds, however, the linear stretch at Spurn Head does appear, and thus is either marked as an additional major stretch, or results in the linear stretch along

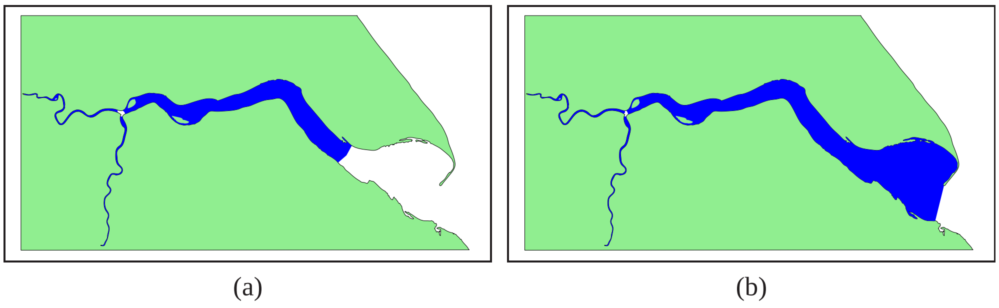


Figure 7.10: The results of marking major stretches on the Humber Estuary using linearity thresholds 1.2 and 1.4.

the River Humber extending across the bay-like region. The inclusion of the confluence predicate means the *interstretch* region that connects all three principal stretches is now marked as a confluence, thus the three main stretches are separate, as was the case in Figure 7.4.

7.4.9 Humber Estuary - Conclusion

Using the Humber Estuary dataset, GEOLOG was able to evaluate the queries and segment the initial data into regions that would be of interest to a GIS user, particularly the individuation of the principal channels. The linearity segmentation was close to the expected result given in Figure 7.3, although there were additional stretches marked at Spurn Head. The *interstretch* query was capable of identifying gaps, though it would have also joined all three of the stretches identified in Figure 7.3 together. Thus, *interstretch* needed to be refined through other queries. The island-water and confluence queries performed as expected, with the confluence query identifying the junction of the three rivers, allowing them to be individuated appropriately at the major stretch stage.

The bay-like region at the mouth of the river is problematic, since it is not clear whether this should be considered a separate feature or not. Depending upon the thresholds used for linearity and *interstretch*, the bay may be part of a major stretch or separate. However, as noted previously, geographic features are typically part of another feature, hence it may be applicable to construct a query that marks the bay region whilst still allowing it to be considered part of the river. Further work would be required to determine the most effective handling of this region.

Overall, GEOLOG was able to match the expected results in Section 7.4.2 quite closely, identifying three main stretches with the major stretch query. However, there were anomolies generated due to the impact of Spurn Head.

7.5 The River Tyne

The next dataset considered is the River Tyne, situated on the North-East Coast of England, which has smaller rivers, such as the River Derwent, feeding into it along its course. In the dataset used, the Tyne is only represented as far as the town of Prudhoe. In addition, only a small portion of the River Derwent is represented.

7.5.1 River Tyne - Initial Data Input

The initial input for the River Tyne was obtained from the Ordnance Survey Digimap Collections[2], which was again vectorised for input into the system in a similar fashion to the Humber Estuary. The Medial Axis and simplified skeleton were then generated, as shown in Figure 7.11a. The River Tyne is the principal channel in the data, with the end of the River Derwent the small channel extending from the south bank.

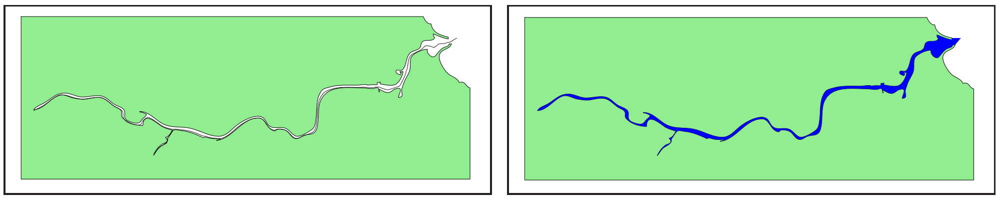

(a) (b)

Figure 7.11: The results of determining the skeleton of the River Tyne from the Medial Axis (Figure 7.11a), together with polygon generated from this skeleton (Figure 7.11b), which represents the extent of the dataset that can be considered inland water.

As with the Humber Estuary, it is then possible to construct a series of base polygons which represents the extent of all inland water, the results of which are shown in Figure 7.11b. The vague predicates are once again used to segment the data, with smaller results presented each section and larger results in Appendix B. Because no islands are contained within the dataset, no results for island or island-water are shown. Evaluating these queries returns false, which is correct for this dataset.

7.5.2 River Tyne - Expected results

The same evaluation process is applied to the River Tyne prior to the queries being run. Figure 7.12 shows the River Tyne dataset overlaid with rectangles, representing the regions expected to be marked as linear stretches. Most of the dataset is expected to be

[2]Ordnance Survey Digimap Collections: http://edina.ac.uk/digimap/ (Visited, August 2006)

marked as part of a linear stretch, with the River Derwent being marked as a single stretch. The River Tyne is expected to be split into three different stretches; one of the splits is due to the junction with the Derwent whereas the other is due to a sudden widening of the river towards the right of the dataset. Again, the *interstretch* query is also tested against this Figure, to evaluate its effectiveness at filling in gaps.

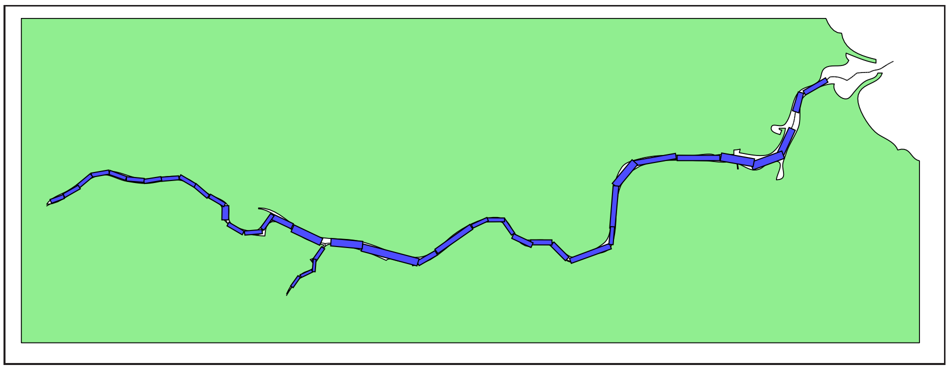


Figure 7.12: The expected results of performing the linearity segmention on the River Tyne.

There are no islands expected within the results, thus there would be no expected island-water regions either. Figure 7.13 shows the expected results of the confluence query, with only one confluence being identified within the data, at the junction of two of the River Tyne linear stretches and the River Derwent linear stretch.

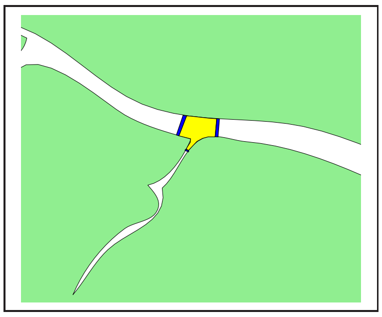


Figure 7.13: The expected results of evaluating the confluence query on the River Tyne. The blue regions represent the ends of the overlaid rectangles that form the boundaries of confluence region. The yellow region is the expected confluence region.

Finally, Figure 7.14 shows the expected results of evaluating the major stretch query on the dataset. The expected result is four distinct major stretches, seperated due to junction of the rivers and the sudden widening that occurs at one point. It is worth noting, however, that this result may not be the preferred result of the system, highlighting potential weaknesses with GEOLOG. This is because the River Tyne is not identified as a single stretch, due to the slight bulge and the junction with the Derwent. Whereas the bulge may be filled in using the interstretch query, the confluence query would always be expected to generate a gap within the River Tyne, because it is unable to indentify the difference between instances of three different rivers and instances of one river joining another (hence two of the three stretches are part of the same river).

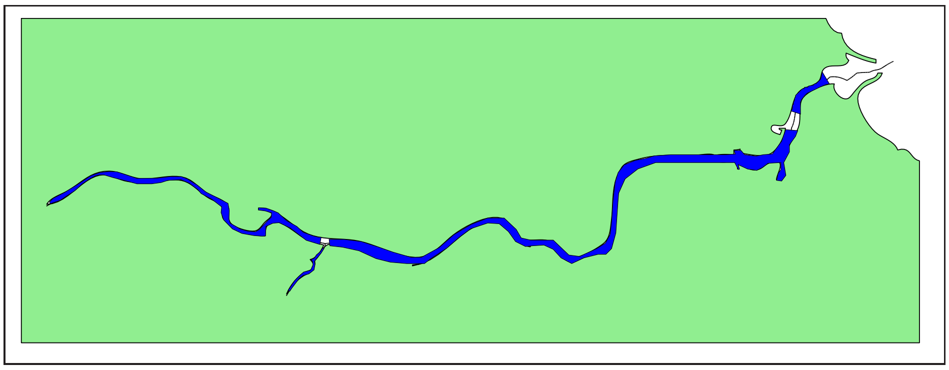


Figure 7.14: The expected results of evaluating the major stretch query on the River Tyne. The blue regions represent the regions expected to be marked as major stretches.

7.5.3 River Tyne - Linearity Segmentation

The results of performing the linearity segmentation using thresholds 1.2 and 1.4 are shown in Figure 7.15, with larger results for all thresholds shown in Figures B.1 - B.3 in Appendix B. Linear segments are displayed in blue. At all thresholds, the results quite

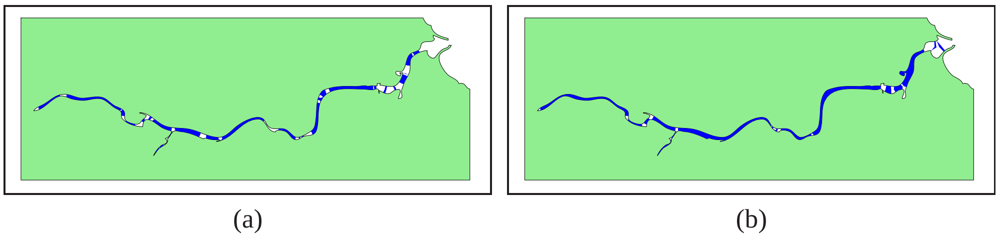


Figure 7.15: The results of performing the linearity segmentation on the River Tyne using thresholds of 1.2 and 1.4.

closely match the expected results given in Figure 7.12, although some gaps do exist. The River Derwent is marked as two stretches due to the sharp bend in it. The expected gap in the River Tyne has been marked as linear, and at higher thresholds linear stretches appear near the mouth of the Tyne, which are again not marked in Figure 7.12.

7.5.4 River Tyne - Linearity with Respect to Edges

The results of performing the linearity with respect to edges segmentation are shown in Figure 7.16, with larger results shown in Figures B.4 and B.5. Linear regions marked in blue. As was the case with the Humber Estuary, linear regions which were close to being a

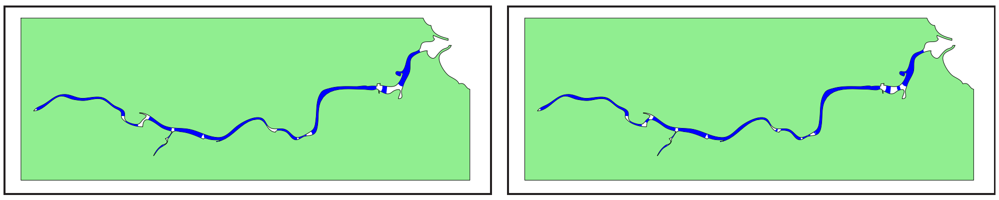

(a) (b)

Figure 7.16: The results of performing the linearity segmentation with respect to the edges on the River Tyne using thresholds of 1.2 and 1.3/1.4 (the results were the same for these thresholds).

single point on one bank were eliminated (such as near the mouth of the river). However, there was little difference in general between the results of the linearity with respect to centre and with respect to edges. There was no difference between using a threshold of 1.3 and 1.4. The results, therefore, more closely match Figure 7.12, due to the removal of the linear stretches near the mouth.

7.5.5 River Tyne - Interstretches

The result of performing the *interstretch* query at all three threshold values is shown in Figure 7.17, with a larger version shown in Figure B.6. Linear regions are marked blue and *interstretches* marked red. The same regions are marked as *interstretches* at

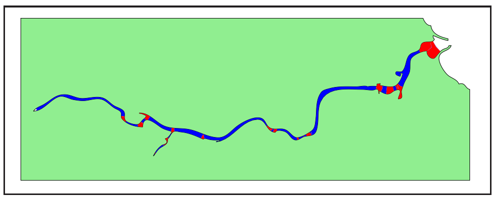


Figure 7.17: The results of marking interstretches on the River Tyne using closeness threshold values 1.0 - 5.0. The results are the same for all values

all threshold values tested. As occurred with the Humber Estuary, the query identified all small gaps between stretches (thus bringing the results closer to the expected results shown in Figure 7.12), but also identified the junction of the Derwent and the Tyne as an interstretch. Thus, the interstretch query was again too inclusive if it were to be the only additional query to fill in gaps. In addition, due to a linear stretch being marked at the mouth of the River Tyne, the bay-like region near the mouth is also marked as an interstretch.

7.5.6 River Tyne - Confluence

The results of performing the confluence query for thresholds 1.2 and 1.4 are shown in Figure 7.18, with larger results for all thresholds shown in Figure B.7. The only confluence found within the dataset occurs at the junction of the Tyne and the Derwent.

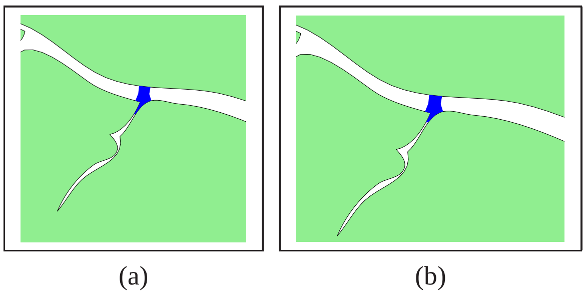


Figure 7.18: The results of marking confluences on the River Tyne using linearity thresholds of 1.2 and 1.4.

The confluence extends further into the Derwent as the value of the threshold is decreased, whilst the region occupied by the confluence in the Tyne stays constant. This closely matches the expected result given in Figure 7.13, though as was noted in Section 7.5.2, this also highlights a limitation of the confluence query, since it is unable to identify the two River Tyne linear stretches as being part of the same river. However, when evaluated against the evaluation criteria discussed in Section 7.3, the query is successful.

7.5.7 River Tyne - Major Stretches

The results of performing the Major Stretch query for thresholds 1.2 and 1.4 are shown in Figure 7.19, with larger results for all thresholds shown in Figures B.8 - B.10.

As would be expected given the previous results in this section, at all threshold values there are three major stretches marked, with the stretch nearest to the mouth lengthening as the linearity threshold is increased. Thus, at the lowest threshold, the results closely match the expected result given in Figure 7.14, although at higher threshold the major stretch nearest the mouth extends further than expected towards the mouth, due to the extra linear region marked.

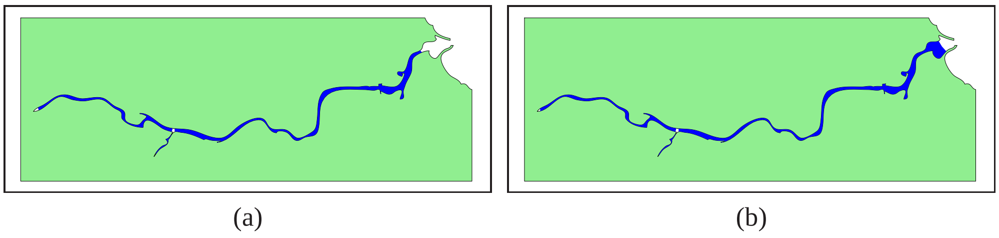


Figure 7.19: The results of marking the major stretches on the River Tyne using thresholds of 1.2 and 1.4.

7.5.8 River Tyne - Conclusion

GEOLOG again was able to generate results that closely matched those set out in Section 7.5.2. The linear segmentation quite closely matched the expected results (although higher thresholds marked regions near to the mouth as linear). The *interstretch* query was also effective at identifying gaps, but again was too inclusive at the junction of the rivers. The confluence query identified the expected junction, and the culmination of these results in the major stretch query meant there were three distinct stretches identified.

However, as noted previously, although the results match the criteria set out for evaluation, they also highlight a limitation of the confluence query. The query is unable to determine the difference between mainstems (the principal river in a network, such as the Tyne) from tributaries (rivers feeding into the mainstem, such as the Derwent), meaning the Tyne is seperated into two stretches. Determining the difference between mainstems and tributaries can be performed through reasoning about the flow network using graphs (Paiva and Egenhofer, (in press; Paiva et al., 1992); for example, considering the angle between different channels. However, incorporating this would mean incorporating the internal graph structure of the skeleton as stored by GEOLOG into the actual reasoning, which is problematic in first order logic, and may in fact require a higher order logic to be handled correctly. Thus, although functions could be written in GEOLOG to determine a primary channel within the skeleton, representing this logically would be difficult.

One option to rectify this would be to treat the angle measurement as a vague predicate, where when the angle between two channels is above some threshold the channels are joined, and allow the threshold to be varied depending upon the intended standpoint. This would allow the two cases to be treated differently; the angle between the two channels of the Tyne is large (close to $180°$) in comparison to the angle between these channels and the Derwent, whereas the angles between the channels of the Humber are all approximately the same.

Another possible vague measurement that could be considered would be to compare the radii of the maximal inscribed circles occurring at the nodes shared by both the confluence and the connected linear regions. The intention would be that if two of the radii were noticeably closer in size than the other radii in the set, the associated channels could be considered to be of the same mainstem and joined together. Otherwise, the channels would all be considered separate. This would require a higher order reasoning like the angle approach, thus further work is required on this problem.

7.6 The Stour-Orwell Estuary

The final dataset tested was that of the Stour-Orwell Estuary, situated in Suffolk and Essex on the East Coast of England. The estuary is formed by the confluence of the Rivers Stour and Orwell meeting at Shotley, although the extent of the estuary is not agreed upon, with some sources marking the estuary boundary as far inland as where the widths of the rivers significantly reduce. The River Orwell flows South from its source, the River Tipping. The River Stour flows East, forming the county boundary between Suffolk and Essex.

7.6.1 Stour-Orwell Estuary - Initial Data Input

The initial input for the Stour-Orwell Estuary was, like the River Tyne, obtained from the Ordnance Survey Digimap Collections. The results of vectorising and determining the Medial Axis and simplified skeleton are shown in Figure 7.20a. The River Orwell is the large channel extending from the top of the dataset. The river is only shown from where it widens significantly at Ipswich. The River Stour extends from the left of the dataset, and again, only the significantly wide region of the river is contained within the dataset. The fork at the end of the dataset is not, in fact, a confluence of two rivers, but the Stour splitting around an island, with the two channels rejoining further along.

The results of generating the base water polygons from this skeleton are shown in Figure 7.20b. Figures for the results for each vague predicate are given in each section, with larger results given in Appendix C.

7.6.2 Stour Orwell Estuary - Expected results

Figure 7.21 shows the expected results of applying the linearity segmentation to the Stour Orwell Estuary dataset. The River Orwell is expected to form two linear stretches, separated by a slight bulge near the top of the dataset. Because of the prescence of islands, the River Stour is expected to be formed of several linear stretches. In addition, because the rectangles only overlap slightly in some cases, the number of ‘gaps’ between stretches is

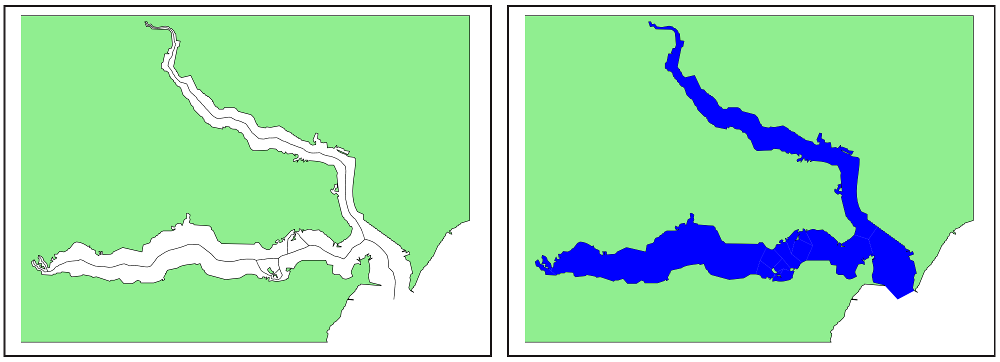

(a) (b)

Figure 7.20: The results of determining the skeleton of the Stour-Orwell Estuary from the Medial Axis (Figure 7.20a), together with polygon generated from this skeleton (Figure 7.20b), which represents the extent of the dataset that can be considered inland water.

likely to increase, hence the *interstretch* query may be required more than for the previous two datasets. There is also an expected linear stretch at the mouth of the estuary.

Figure 7.22 shows the expected results of the island, island-water and confluence queries. For island-water, despite the prescence of several islands, there is expected to be only a single island-water region marked, which encompasses all three expected islands. This is because there is no linear stretch in between the islands that would act as a boundary. For the confluence query, there are two expected confluences within the dataset; one at the junction of the Stour and Orwell and one within the River Stour towards the left of the dataset.

Finally, Figure 7.23 shows the expected results of the major stretch query, as the culmination of the previous stages. The River Orwell is expected to be split into two major stretches, whereas the River Stour will be formed of a several major stretches, including an extensive one that encompasses the island-water regions identified previously.

7.6.3 Stour-Orwell Estuary - Linearity Segmentation

The results of applying the previously described approach to segmenting the data into linear regions for thresholds of 1.2 and 1.4 are shown in Figure 7.24, with larger results for all thresholds shown in Figures C.1 - C.3 in Appendix C. Linear sections are marked in blue.

In comparison to the expected results shown in Figure 7.21, GEOLOG performs better at higher thresholds, since the lowest threshold marks very little as linear along the River Stour. However, parts expected to be marked as linear towards the left of the dataset are not marked as such, which may impact upon later queries.

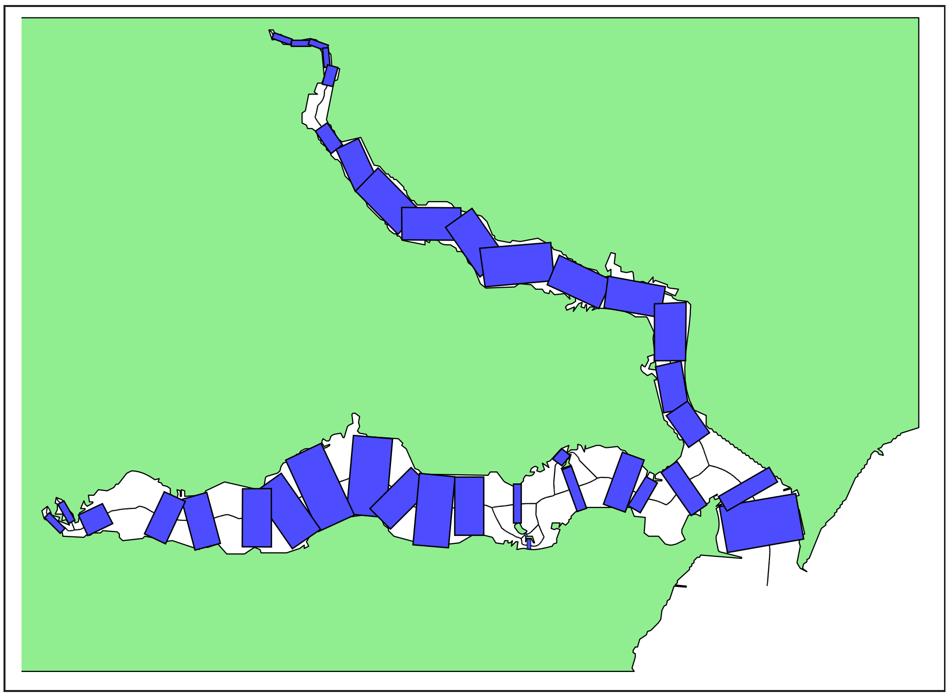


Figure 7.21: The expected results of performing the linearity segmention on the Stour Orwell Estuary.

7.6.4 Stour-Orwell Estuary - Linearity with Respect to Edges

The results of applying the refined linearity definition with respect to the edges are shown in Figure 7.25, with larger results shown in Figure C.4. Linear regions are marked in blue. At all thresholds, the same results are generated, with most linear segments removed. In comparison to Figure 7.21, the refined linearity query removes too many linear stretches, particularly along the River Stour. Thus, the linearity with respect to the edges is too restrictive for Stour Orwell Estuary dataset.

7.6.5 Stour-Orwell Estuary - Interstretches

The results of marking *interstretches* for thresholds of 1.0 and 5.0 are shown in Figure 7.26, with larger results shown in Figures C.5 and C.6. Linear regions are marked in blue and *interstretches* are marked in red. In comparison to Figure 7.21, the *interstretch* query is more effective at the lowest threshold, as at the higher thresholds it is too inclusive, with all expansive regions marked as *interstretches*. Since the threshold is set as an actual distance in kilometres, it may be too long in relation to the lengths of the rivers in this dataset; the River Humber is significantly larger and needs longer distances. This suggests that the distance used for the threshold may need to be related to the length of the rivers

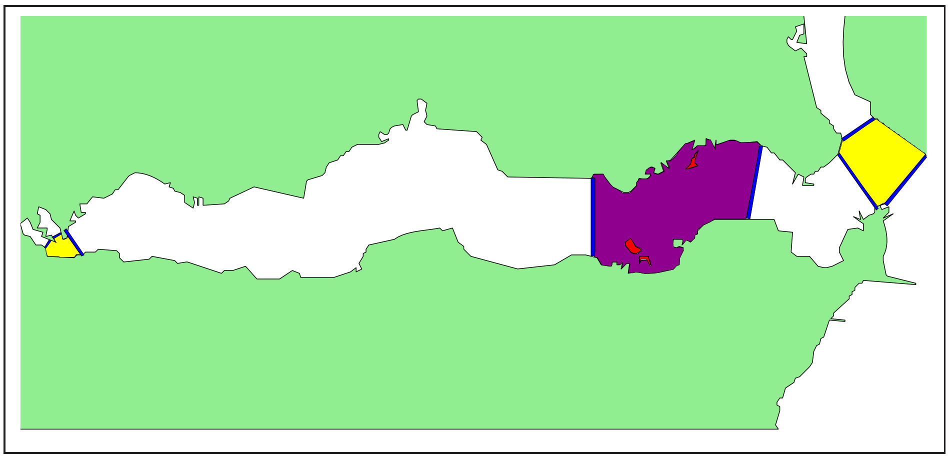


Figure 7.22: The expected results of evaluating the island, island-water and confluence queries on the Stour Orwell Estuary. The blue regions represent the ends of the overlaid rectangles that form the boundaries of confluence region. The red regions represent the expected islands. The purple regions represent the expected island-water regions. The yellow region is the expected confluence region.

within the dataset.

7.6.6 Stour-Orwell Estuary - Island and Island-Water

The results of evaluating the island and island-water queries for thresholds of 1.2 and 1.4 are shown in Figure 7.27, with larger results for all thresholds shown in Figures C.7 - C.9. Islands are marked in red and island-water regions are marked in blue. As with the Humber Estuary, the same islands are marked in all three, as the island query is not dependent upon a standpoint.

The islands within the dataset are examples of the sort of ‘artefacts’ that the system would want to ignore, as they are very small in comparison with the region of water surrounding them. Thus, if a vague predicate was introduced that related the size of the island to the surrounding water to determine island-water regions, in the Humber Estuary dataset there may be thresholds where the water surrounding the island is considered separate and others where they are joined, whereas in this dataset it is unlikely the channels around the islands would ever be considered separate.

When compared to Figure 7.22, the island-water query is most successful at higher thresholds, as at lower thresholds the island-water region stretches into the junction of the two rivers. At all thresholds, the region stretches further to the right of the dataset than was expected. Thus, the island-water query is less successful when used on the Stour Orwell Estuary in comparison to the previous two datasets.

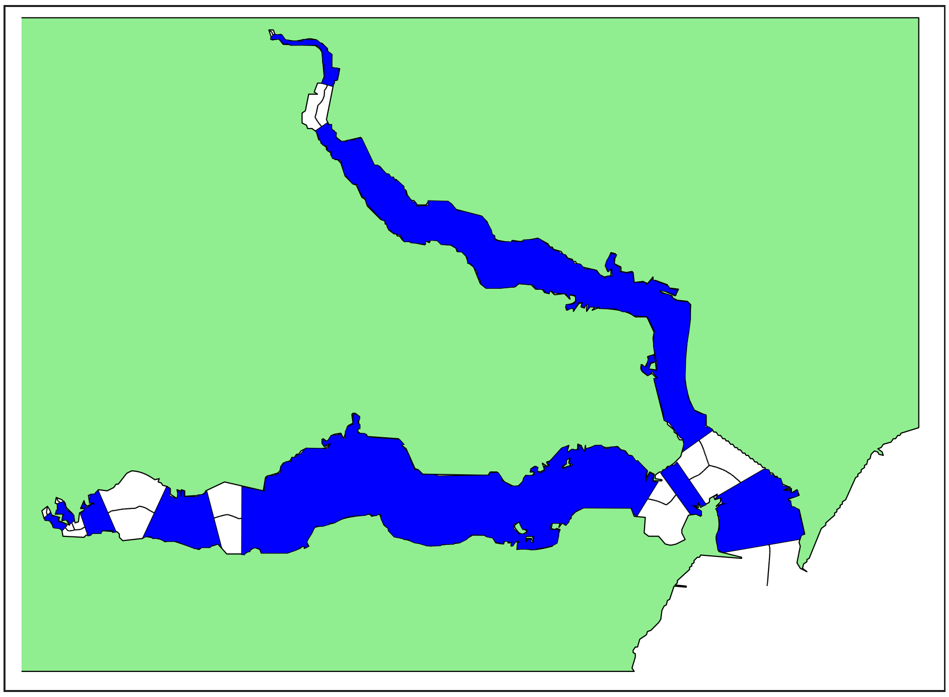


Figure 7.23: The expected results of evaluating the major stretch query on the Stour Orwell Estuary. The blue regions represent the regions expected to be marked as major stretches.

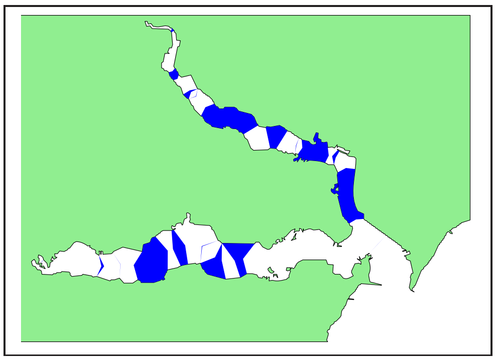
7.6.7 Stour-Orwell Estuary - Confluence

The results of determining confluences for thresholds of 1.3 and 1.4 are shown in Figure 7.9, with larger results shown in Figures C.10 and C.11.

The results of evaluating this query using the Stour-Orwell Estuary dataset vary across all the linearity thresholds used. In Figure 7.22, two regions were expected to be marked as confluences. However, at the lowest linearity threshold, no regions were marked as confluences. At the next linearity threshold used, the query identified two confluences in the same location as the expected results. However, the confluence to the left of the dataset is larger than the expected result given in Figure 7.22.

Finally, at the highest threshold tested, the confluence in the River Stour remains the whereas the confluence at the junction of the two rivers is removed. A closer examination of the reason for this is shown in Figure 7.29. One of the requirements of the confluence query is that the three stretches are disconnected, to ensure they are distinct channels from each other. However, in Figure 7.29, the circled area shows that two of the linear stretches now touch at a single point, hence are not considered disconnected and rending the query false for this confluence.

The results of evaluating the confluence query over the Stour-Orwell Estuary, there-


(a)

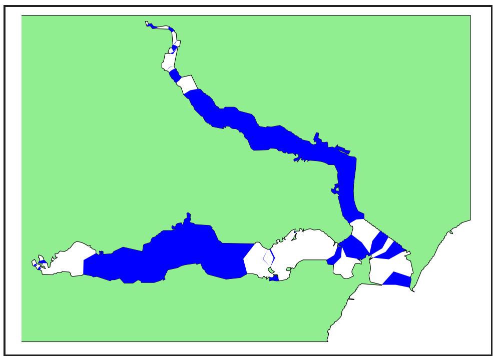

(b)

Figure 7.24: The results of performing the linearity segmentation on the Stour-Orwell Estuary using thresholds of 1.2 and 1.4.

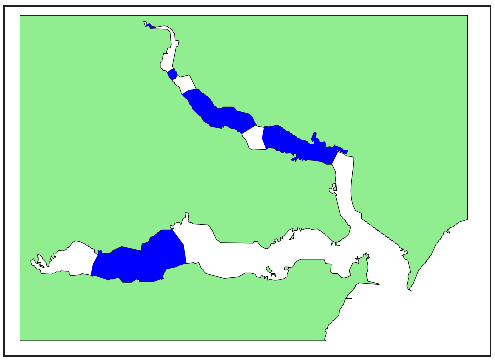


Figure 7.25: The results of performing the linearity segmentation with respect to the edges on the Stour-Orwell Estuary using thresholds 1.2 - 1.4 (the same results are generated for all threshold values).

fore, performs poorly in comparison to the expected results. The query is very sensitive in relation to the linearity threshold; the confluence between the Stour and Orwell does not occur if the threshold is too low or too high. This would be expected to occur in other datasets, since an increase in the linearity threshold could eventually lead to two linear stretches on different channels expanding far enough into the junction between the channels that they touch each other.

7.6.8 Stour-Orwell Estuary - Major Stretch

Finally, the major stretch query was evaluated, with the results of using thresholds of 1.2 and 1.4 shown in Figure 7.30 and larger results for all thresholds in Figures C.12 - C.14. Major stretches are marked in blue. Following on from the results of the confluence query, the success of the individuation of the two rivers into major stretches is mixed, in

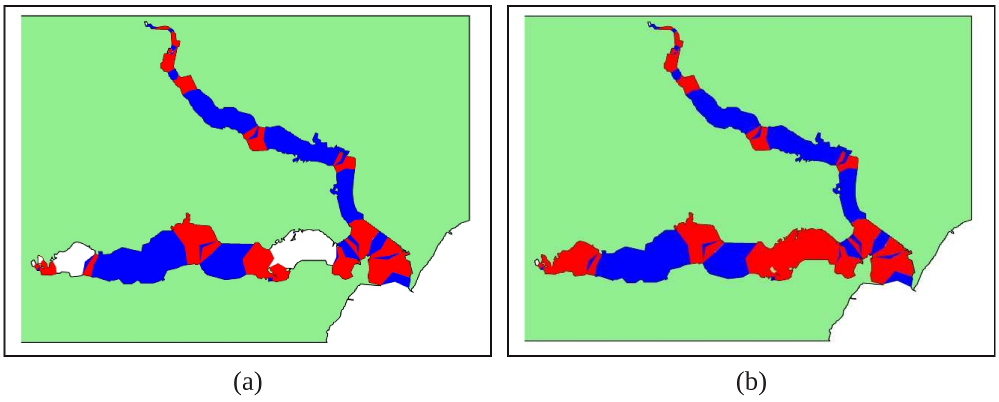


Figure 7.26: The results of marking interstretches on the Stour-Orwell Estuary using thresholds 1.0 and 5.0

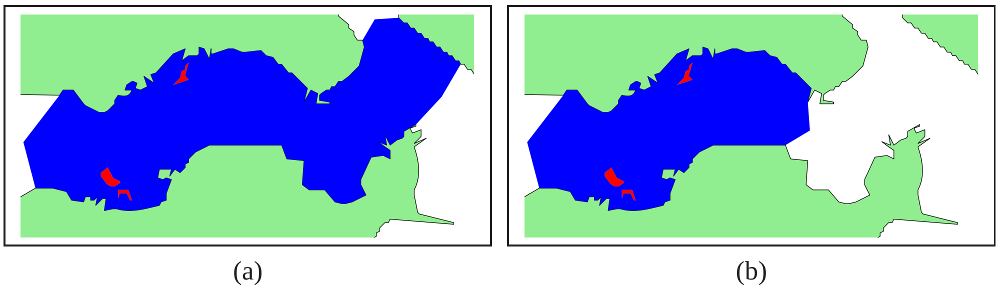


Figure 7.27: The results of marking islands and island-water regions on the Stour-Orwell Estuary using thresholds of 1.2 and 1.4

comparison to the expected results given in Figure 7.23. At the lowest and highest thresholds, the two rivers are part of the same major stretch, due to the lack of the confluence region between the two being (as discussed in the previous section). The results of the middle linearity threshold are threfore closest to the expected results, with each of the rivers marked as a single major stretch (although in the expected results, the Stour was expected to be formed of several major stretches), and an additional major stretch at the mouth of the Estuary.

Similar to the results of evaluating the confluence query, the variation in the results of evaluating the major stretch query is more restricted by the thresholds than seen in the other datasets. Again, there is a small range in which the two rivers are considered separate, depending upon the linearity threshold used. The problems relating to confluences is also clear when dealing with the forks at the end of the Stour, since this region is not marked as part of the main channel.

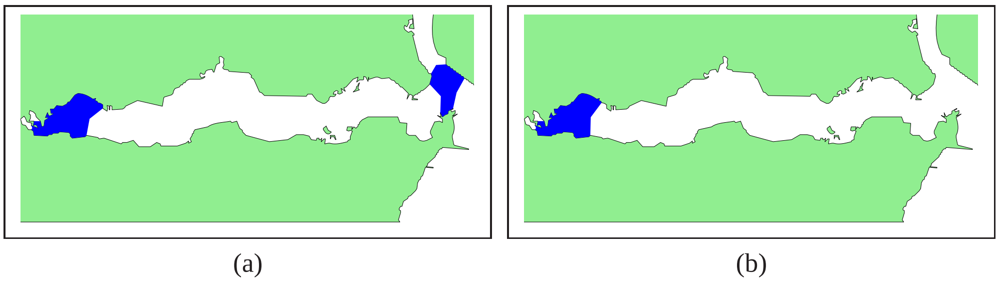


Figure 7.28: The results of marking confluence regions on the Stour-Orwell Estuary using thresholds of 1.3 and 1.4.

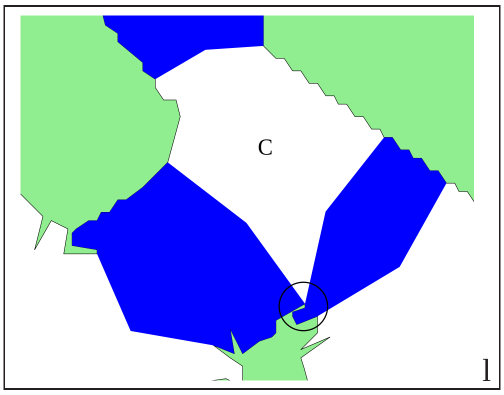


Figure 7.29: A closer look at a confluence candidate region within the Stour-Orwell Estuary at a linearity threshold of 1.4. The linear regions are marked in blue, and the white region labelled C that is connected to all three would seem to be a candidate for a confluence. However, two of the linear stretches are connected, thus the query evaluates to false for this region.

7.6.9 Stour-Orwell Estuary - Conclusion

The Stour-Orwell is the most sensitive to variations in the thresholds used to determine standpoints, and thus varies most in comparison to the expected results outlined in Section 7.6.2. The linearity segmentation stage did not mark large sections as linear that were identified as such in Figure 7.29. This has an effect on the subsequent queries. In the case of *interstretches*, the distances used for the threshold appear to be too large in relation to the data, as all expansive regions were marked as *interstretches* at higher linearity thresholds. This suggests that the distance used should be related to the size of the data set, either by choosing smaller thresholds, or coding the threshold to be determined relative to the overall size of the dataset in question.

The evaluation of the queries defining confluences and major stretches are particularly sensitive to the thresholds, with a small range determining whether the two rivers are separate or not. Thus, the performance of the queries when compared to the expected

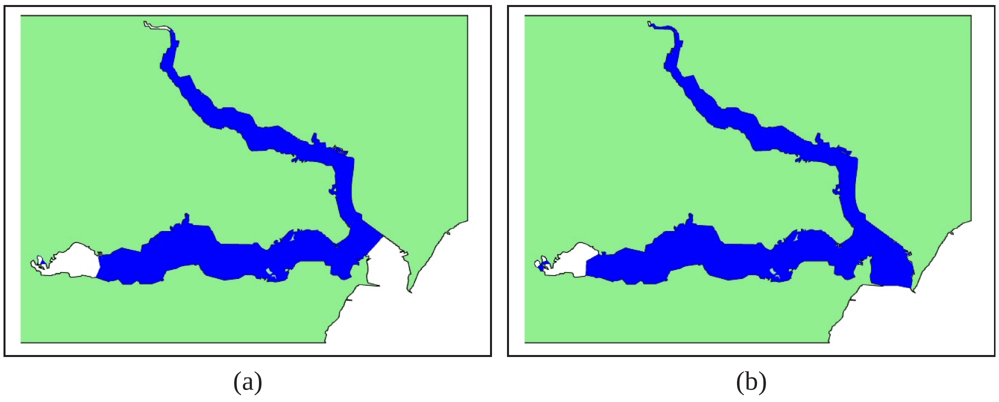


Figure 7.30: The results of marking major stretches on the Stour-Orwell Estuary using thresholds of 1.2 and 1.4.

results was poorer than with the previous two datasets.

Overall, GEOLOG was able to individuate features within the Stour-Orwell Estuary for some thresholds, but was very sensitive to variations. This meant that the results obtained deviated from the expected results at some thresholds used. The performance, therefore, was poorer than with the previous two datasets used.

7.7 Overall Analysis

Now that GEOLOG has been tested using three different datasets, the results as a whole can be considered, to determine what GEOLOG is successful at and where it is deficient, and thus the overall effectiveness of the approach. This includes consideration of the choice of vague predicates to represent vague properties, how effective the thresholds were (both how they were determined and how restrictive they were) and the results generated by evaluating the queries. The success of GEOLOG is measured in terms of how closely the results managed the expected results for each datasets, which stemmed from the overlaid rectangles approach discussed in Section 7.3.

7.7.1 Vague Predicates and Thresholds Used

The principal vague predicate used was that of linearity, with two different definition considered; one which measured linearity relative to the width and another which refined this to consider the lengths of the banks. The first measurement managed to produce results similar to the expected results given beforehand, whereas the refined predicate was too restrictive for some linearity thresholds; in particular, the Stour Orwell Estuary dataset. The threshold could be fine tuned for each dataset to produce better results; for example, the Stour Orwell Estuary produced results closer to the expected results at higher thresholds, whereas for the Humber Estuary a lower threshold was best. As it is a vague predicate, this is expected.

The other vague predicate used was ‘close-to’, using the closeness threshold. Whilst this was capable of marking gaps between linear stretches (to bring the results closer to the expected results), the distance used proved too large in some cases. This may be related to the scale of the datasets, as a 1km gap is more significant in a smaller river (such as the Tyne) than a larger one (such as the Humber). One option to rectify this would be to relate the distance of the threshold to the lengths of the channels within the dataset; for example, the threshold could be in terms of percentage of the total size of the dataset, as opposed to a specific distance. Thus, if the dataset in question was 10km long, a threshold value of 10% would represent 1km, whereas for a 100km long dataset this would represent 10km. This may be problematic, as it may not be clear what should be measured to determine the overall length. Alternatively, a lower threshold value could instead be used on smaller datasets.

7.7.2 Queries

The next consideration is the overall success of the queries used to generate features of interest, building upon the previously discussed vague predicates. With islands, the query is not dependent upon a standpoint; thus, the results of evaluating that query over a dataset are the same, irrespective of the standpoint taken. As was noted with the definition of island, however, the query is in part dependent upon restrictions in place on the domain, such as the presence of distinct matter types. If these restrictions were relaxed (for example, to allow a region to have parts that were both land and water), then the definition may not be suitable. The island-water query was able to identify regions similar to the expected results for the Humber Estuary and Stour Orwell Estuary. However, the Stour Orwell Estuary’s island-water region was larger at lower thresholds.

The confluence query had mixed results when compared with the expected results. The results of the Humber Estuary and the River Tyne datasets closely match the expected results detailed (although, as noted, the confluence region in the River Tyne was not a true confluence, and thus highlighted a limitation of the query). However, the confluence query had different results at different thresholds, varying from no confluence regions being marked, to one or two marked (depending on the threshold).

The final query that was evaluated was the major stretch query, which built upon the previous queries to mark the principal channels within the dataset. The results of this stage were dependent on the results of the confluence query in terms of the individuation of the channels; hence, the problems that occurred for the confluence evaluation impacted on this query also. Thus, improving the confluence query could improve the major stretch marking. For all the datasets, GEOLOG was able to the datasets into major stretches that quite closely match the expected results, although at higher thresholds the query was usually too inclusive (such as including the bay-like regions of the Humber Estuary or the River Tyne and forming a single major stretch in the Stour Orwell Estuary).

Overall, the queries do effectively segment features that would be of use to a GIS user. The major stretches query for example, can be refined to the using different threshold values to ensure the principal channels are individuated.

7.7.3 Conclusion

GEOLOG was successful at segmenting the datasets into features of interest when compared to the evaluation criteria, using first order logic queries to generate the results. The extent to which it was successful and matched the expected results varied between datasets, but this was to be expected given the variation in the datasets considered. As was noted in the aims of the thesis in Chapter 1, the intention is to work with the vagueness, not remove it entirely. Therefore, the adjusting of thresholds to satisfy each dataset individually is to be expected. Further queries could also be developed of a similar nature to those described in this chapter, to segment required features that are even closer to the user's requirements.

7.8 Summary

This chapter has examined the results of using GEOLOG to generate features based upon vague predicates. The formation of the queries was discussed, showing how they were built up from base predicates and vague predicates, through to the final main query of major stretches. These queries were then applied to three datasets; the principal case study of the Humber Estuary, the River Tyne and finally the Stour-Orwell Estuary. The results of evaluating the queries for each dataset was considered individually, before the comparing the results overall.

Chapter 8

Conclusions and Future Work

8.1 Overview

The main aim of this thesis was to develop a method of incorporating representation and reasoning about vague features into geographic information systems, by developing a system that would allow an ontology to be grounded upon a topographic dataset. It was hoped this grounding layer would improve the handling of vagueness within geographic ontologies by introducing a robust method of reasoning about the vagueness without the need to modify existing ontologies (beyond modification to work with the grounding level). To achieve this, a smaller section of the geographic domain was concentrated upon (inland water networks), to determine methods of handling topographic data for this domain.

In Chapter 3, the principal approaches to reasoning about vagueness were compared, to determine an effective approach for handling vagueness within a particular problem. The main points to consider were identified as the input format of the data, the intended framework within which the vagueness is to take place and finally the intended output of the results. It was shown that Supervaluation Theory was best suited to the problem of vagueness within topographic datasets of inland water networks, though Supervaluation Semantics did not explicitly define an implementation. Standpoint Semantics, which uses Supervaluation Theory as a framework, was thus chosen to handle the vagueness within the data.

With the approach for reasoning about the vague features decided upon, the core of the thesis was the development of the GEOLOG system, which takes topographic data as an input, and, through first order logic querying, allows the collection of attributes about the data which can be used to generate segmented regions representing geographic features. Chapter 4 showed that the representation of the input data needs to be considered carefully, to ensure an efficient method of representing the information was selected. In the case of inland water networks, a simplified skeleton generated from the Medial Axis was shown to be an effective approach. Once the data had been converted into the required input format, the collection of attributes and segmentation of the data was considered in Chapter 5. There, it was shown how polygons could be generated from sets of connected edges in the simplified skeleton, as well as how attributes such as 'linearity' could be identified using thresholds to determine whether an edge of the skeleton was marked as having an attribute or not.

Chapter 6 discussed grounding an ontology upon the data level, building upon the attribute collection from the previous chapter. It was shown how this stage could be performed using *effective generator relations*, that would create the required regions in a systematic way. The handling of first order logic queries was performed using model building approaches, where a partial model was built of a query and used to determine whether the query was true or false. The results of this stage could then be output in various ways. Finally, Chapter 7 tested GEOLOG using a set of queries and three different datasets, to assess the suitability of GEOLOG and the grounding approach as a whole.

As was discussed in Chapter 1, this work resulted in several principal achievements: First, a method of working with vagueness within the geographic domain was determined, expanding previous work. This allowed vagueness to be incorporated into definitions of predicates and features, as opposed to the vague aspects of these being removed or ignored. From this, the importance of the grounding level was shown, including the geometric computation stages that were required for this, both the representation of the initial dataset and the calculations of spatial relations and new regions. This again represented an improvement over static datasets, as regions could be generated as required in a controlled manner. Finally, model checking approaches were developed further, to work with larger domains than typically used. This was performed using a streamlining of the domain, expanding as required rather than generating the full domain to begin with. This was shown to be efficient and allowed queries to generate required features within a logical framework.

8.2 Discussion and Future Work

In addition to the achievements, it is worth considering the limitations of this thesis, to determine areas that may require improvement. The work within this thesis could be extended in several ways; thus, looking at these limitations will help evaluate the extensions that would be of value.

8.2.1 Additional Attributes

In this thesis, two attributes were used to define vague predicates, 'linearity' and 'closeness'. Although both were found to perform their intended tasks in most cases, there still remained discrepancies within the results. For example, with linearity there were small regions that were not marked as linear, resulting in the splitting of a single stretch into two smaller, close together stretches. This led to an optional refined definition, where the edges were considered as well as the widths of the stretch. With closeness, determining a suitable threshold to ensure only certain 'gaps' were filled in proved problematic, since relaxing this threshold too much meant larger regions were marked as 'gaps', which was again not the intention. Potential improvements for this attribute, therefore, included relating the threshold to the size of the dataset, or perhaps relating the length of the gap to the widths along it.

However, as argued previously, due to the nature of vagueness it is unlikely that any refinement to these attributes would result in the removal of all discrepancies within all datasets. Thus, rather than attempting to remove vagueness entirely with a universal attribute, it is more appropriate to define attributes that generate the most suitable results in our desired datasets, ensuring the definition is made clear. For example, with linearity, if a user felt the refinement with respect to edges was unsuitable, they would not need to use it.

Another limitation of this thesis was that only two attributes were considered. Although these generated satisfactory results with the queries used, there will be other features for which the attributes would be unsuitable. A potential expansion, therefore, would be to investigate what other vague attributes could be required and how these could be collected. For example, would linearity and closeness be applicable within other areas of the geographic domain and if so, how would they be measured? Also, how many such attributes would be required to completely cover all required features?

Potentially, the number of required attributes to fully define all features could be extensive, depending upon the required level of segmentation required. For example, with geometric attributes, such as linearity, what is the intended boundary; would linear features be required to extend to the land boundary (as is the case in this thesis), or would more arbitrary boundaries that do not necessarily touch the land boundary be allowed? An example of this was discussed in Section 6.3.6.2, where an effective generator relation was described that demarcated a region near to the centre of a polygon. Although this relation was shown to need refinement in order to reach a fixed-point, such an attribute could be implemented. This would require extensions to GEOLOG, since GEOLOG is limited to the generation of polygons in relation to the simplified Medial Axis Skeleton which extend to the land region.

Ideally though, the number of attributes should be kept to a minimum. If there are too many attributes (and thresholds), calculations in GEOLOG may become too computationally complex, as well as render GEOLOG unusable due to the user having to change too many thresholds to generate results. A further extension, therefore, would be to consider the usability of the system, in order to determine the optimum number of attributes and thresholds, as well as to investigate methods of grouping these to ensure the system is still usable.

8.2.2 System Integration

As noted in Chapter 4, the VRONI computation stage is performed separately to the Prolog code of GEOLOG. In addition, a certain degree of pre-computation was required to vectorise the data and check the results of VRONI for errors. These steps were done manually, as opposed to programming parsers to take data in standard geographic formats and check for errors. A more effective system would integrate the input directly into GEOLOG, by including parsers from standard geographic data formats into VRONI, then from VRONI into GEOLOG.

For example, a popular GIS format is Geography Markup Language (GML)[1], which is written using XML. It should therefore be possible to parse data in this format into the required format of VRONI, thus allowing GML data to be input directly into the system. VRONI is written in C and since it is possible to call C from Prolog, it should be possible to create an interface between the two, thus making GEOLOG a fully integrated system.

Potential advantages this may offer would include the ability to calculate information such as the Medial Axis or the simplified skeleton ‘on the fly’, which may allow more flexibility in definitions. For example, as noted in Section 5.3.4, the Medial Axis will vary depending on the presence of islands. Thus, one option would be to calculate the skeleton generated when islands are removed, therefore allowing the linearity calculation

[1]GML Encoding Standard: http://www.opengeospatial.org/standards/gml (Visited, August 2007)

to use alternative skeletons as required.

Another potential enhancement would be to allow more control over the contour partitioning stage, which is also performed manually in this thesis. Instead of using the contours specified, the user could choose their own contours of interest (for example including or excluding bays through the use of different contours), or choose to ignore them entirely. A future extension to the work, therefore, is to integrate VRONI with GEOLOG, to allow data to be input directly without pre-computation, to give the user more flexibility in the skeleton generated and determine the benefits or problems with this. For example, potential benefits include a more flexible system that can be used for all stages, whereas problems include increased computational complexity as opposed to pre-computation of skeletons. Also, it would need to be investigated whether users would in fact want to have control of this stage, or would prefer the data to have been analysed and skeletons generated in advance.

8.2.3 Ontology Integration

Following on from the limitations mentioned in the previous section is the lack of integration with a higher-level ontology. Although the intention of this thesis was to determine methods of handling vagueness within geographic ontologies, the actual integration with an ontology was not considered in depth. For example, although the relations between higher-level primitives such as ‘river’ and the features generated at the grounding level were considered, they were not explicitly linked. Instead, this thesis concentrated upon the construction of features that could be used in the eventual definition of such primitives.

Therefore, an obvious extension would be to integrate the GEOLOG system with a higher level ontology. If interaction with the semantic web was required, then this may involve integration with OWL-DL. Since OWL-DL is written in XML, this could be achieved through parsing output from GEOLOG into this format. For example, if an OWL-DL ontology existed that had primitives such as ‘river’ and ‘lake’ and spatial relations such as RCC-8, GEOLOG could output its results as a series of instances of such features with the RCC-8 relations that hold between them. Thus, if a user were to query the OWL-DL ontology, it would use the results from GEOLOG as its dataset and return features from these results. This would mean the OWL-DL ontology could retain its primitive definition of such features and concentrate on their relation to other aspects of the domain, whereas the GEOLOG grounding level would contain the ‘fleshed out’ definition of the feature. However, this may not offer enough control over the stage, and a solution that combines both levels effectively may be preferable.

Combining the Semantic Web with Prolog has been looked at by Wielemaker (2005) and Wielemaker et al. (2007), where it was noted that Prolog had many attractive properties for Semantic Web applications, such as reasoning and constraint programming. Interfacing between OWL and SWI-Prolog can be handled by the Thea Library[2], whereas PiLLoW (Cabeza and Hermenegildo, 2001) provides support for HTTP protocols (and thus could be used to generate a Semantic Web interface), and is compatible with several versions of Prolog, including SWI-Prolog and SICStus Prolog.

By adding full integration with an ontology level, GEOLOG would offer a more complete solution to the problem of handling vagueness. The ontology level could contain primitives such as ‘river’ or ‘lake’, whereas the grounding level in GEOLOG could also contain an ontology defining how these terms are grounded upon the data through definitions. Maintaining the separation between the two would allow for different higher-level ontologies to be used with the same grounding level.

8.2.4 Expansion of Domain

The domain considered in this thesis was that of inland water networks, and as such represents only a small portion of the geographic domain. Therefore, the approaches developed here may not be applicable to other aspects of the domain: First, the methods developed for representing and extracting information about the domain may only be suitable for the inland water networks. Second, GEOLOG may not be suitable for handling queries across different domains, such as comparing one aspect of the geographic domain (such as inland water networks) with another completely different aspect. These points will now be considered individually, to determine how these could be expanded in future work.

8.2.4.1 Using Different Domains

The use of the simplified Medial Axis skeleton has been shown to be suitable for inland water networks, thus a potential expansion would be to determine if it (or similar derivatives such as the Delaunay triangulation) could be used in other areas of the geographic domain. As an example, consider the classification of built-up areas as a case study, with an example dataset given here as Hepscott Village, derived again from Ordnance Survey Digimap Collections[3]. The dataset consists of a set of polygons representing buildings over a certain size, and is shown in Figure 8.1a. The dataset could easily have included

[2]Thea: An OWL library for SWI-Prolog: http://www.semanticweb.gr/TheaOWLLib/ (visited, August 2007)

[3]Ordnance Survey Digimap Collections: http://edina.ac.uk/digimap/ (Visited, June 2008)

other features such as road or rail connections and classifications of buildings, but for the purposes of this example, buildings are sufficient.

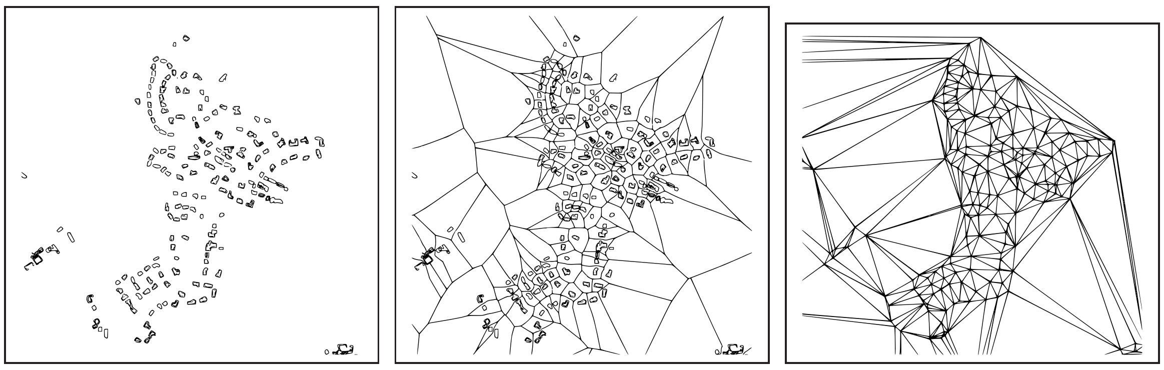

(a) (b) (c)

Figure 8.1: An example of a buildings dataset and how it could potentially be represented. Figure 8.1a shows the initial input of Hepscott Village, Figure 8.1b shows the results of determining the Medial Axis and simplified skeleton, and Figure 8.1c shows the results of determining the Delaunay triangulation.

With built-up areas, there are several features that may be of interest to identify, some of which may be considered vague. One might be the extent to which a particular feature covers the land such as a town or a village. This may already have a legal definition, such as the boundary which is covered by a particular jurisdiction. However, the legal boundaries of such features may not necessarily correspond with where we would intuitively see them. For example, there may exist nearby areas which are considered to be part of a particular built-up area but are not legally defined as such. In addition, the threshold at which something is considered a village as opposed to a town is vague; whilst it may be determined by the population of the area it could also be determined by the overall density and number of the buildings. These are again vague features, which could be handled using GEOLOG.

The first stage is to represent the data for input into GEOLOG. For the example, the Medial Axis was once again obtained using VRONI combined with contour partitioning, where each building was treated as a single contour. The result is now a series of cells, hereby referred to as contour partition cells. The results of this stage are shown in Figure 8.1b.

An alternative additional measurement that can also be obtained using VRONI is the Delaunay triangulation (Delaunay, 1934), if each polygon is replaced instead with a single point (in this case the approximate centre of the polygon was calculated and used). The Delaunay triangulation of a set of points P is a triangulation $DT(P)$ such that no point in P is inside the circumcircle of any triangle in $DT(P)$. The results of this are shown in Figure 8.1c.

This was previously noted in Chapter 4 as a potentially useful measurement of geographic features, as it represents a good method of triangulation, since the process reduces the number of narrow triangles. In addition, if the Delaunay triangulation is represented as a graph with the lengths of the Delaunay edges stored, then it can be used to determine the shortest distances between points. This follows from the Delaunay triangulation being the dual graph of Voronoi diagrams; two points share a Delaunay Edge if their Voronoi regions are connected. Therefore, if a point has n Delaunay edges, the points connected by these edges are the set of n-closest points. Determining the closest is therefore a case of cycling through these edges.

Using both representations could allow different features to be extracted. One potential feature may be to measure the 'density' of buildings, whereby buildings in built-up areas would be expected to closer to each other than buildings outside of built-up areas. This could be measured using the area of the contour partition cells, since a smaller area means that the surrounding buildings are close to that cell. Alternatively, the Delaunay triangulation could be used to determine attributes such as the shortest distance between two buildings, or the number of buildings that are closer than a specified distance to a building. Further work would be necessary to define this attribute correctly, as well as determine other attributes that could be extracted.

The final problem that would need to be considered would be the generation of features from these attributes. For example, with inland water networks, features were generated by identifying stretch of the simplified Medial Axis Skeleton and generating a polygon from this skeleton, expanding outwards to the land boundary. With built-up areas, both the buildings and the space between may need to be joined together, to forma single 'footprint', representing the extent to which that feature covers the land. The suitability of different footprints has been considered previously by Galton and Duckham (2006), where different approaches were proposed, including using the convex hull, the Delaunay triangulation, the Voronoi Diagram and offset circles. These could all be used by GEOLOG.

Thus, GEOLOG could potentially be used to handle other aspects of the geographic domain, by considering the same problems faced with the inland water network domain; how can the data be represented effectively, what attributes can be collected and how can these be used to segment the data into appropriate regions.

8.2.4.2 Combining Different Domains

In this thesis, inland water networks were considered separate to the rest of the geographic domain, as was the example given in the previous section of built-up areas. Often, though, geographic features that are of interest may be in different parts of the domain. For example, the combination of the two domains mentioned above may include the consideration of ‘towns near rivers’ or ‘coastal towns’, which would require definitions that span different sub-domains of the geographic domain. Because inland water networks were considered in isolation, a limitation is the lack of integration GEOLOG allows between different aspects of the geographic domain.

Further expansions to this work would, therefore, consider how such smaller domains could be processed and combined to provide an overall handling of the geographic domain. The work in this thesis supports the idea that it is better to work with smaller domains to handle features rather than attempting to generate a general system covering the entire domain, hence the integration of such smaller domains is crucial. For example, if similar attributes are used in different domains (such as linearity might be), it is important to ensure that the different instances are handled correctly, and that the appropriate calculation is used for each case. Further, queries may need to be modified to take into account larger domains as assumptions that are in place in one domain may not be present in others (such as the separation of matter types).

8.3 Conclusion

This chapter has summarised the achievements and limitations of the work presented in this thesis, as well as considered future expansions. GEOLOG and the handling of vagueness through a grounding level has been shown to improve upon simply ignoring or removing vagueness, by introducing more flexibility and control for the user to define geographic features as required. It is hoped that GEOLOG can act as a basis for future expansion, to further improve the handling of vagueness within geographic ontologies.

Bibliography

A.I. Abdelmoty, M.H. Williams, and N.W. Paton. Deduction and deductive database for geographic data handling. In *Symposium on Large Spatial Databases*, pages 443 – 464, 1993.

P. Agarwal. Ontological considerations in GIScience. *International Journal Of Geographical Information Science*, 19(5):501 – 536, 2005.

J. F. Allen. Maintaining knowledge about temporal intervals. *Communications Of The ACM*, 26(11):832 – 843, 1983.

J.M. Almendros-Jimenez. Constraint logic programming over sets of spatial objects. In *Proceedings of the 2005 ACM SIGPLAN workshop on Curry and functional logic programming*, pages 32 – 42, Tallinn, Estonia, 2005. ACM Press.

F. Baader, D. Calvanese, D.L. McGuinness, D. Nardi, and P.F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*, 2003. Cambridge University Press.

X. Bai, L. J. Latecki, and W. Y. Liu. Skeleton pruning by contour partitioning. In *Discrete Geometry For Computer Imagery, Proceedings*, volume 4245 of *Lecture Notes In Computer Science*, pages 567–579. Springer-Verlag Berlin, Berlin, 2006.

X. Bai, L. J. Latecki, and W. Y. Liu. Skeleton pruning by contour partitioning with discrete curve evolution. *IEEE Transactions On Pattern Analysis And Machine Intelligence*, 29 (3):449 – 462, 2007.

P. Balbiani, J.F. Condotta, and L. Fariñas del Cerro. A model for reasoning about bidimensional temporal relations. In Anthony G. Cohn, Lenhart Schubert, and Stuart C. Shapiro, editors, *KR'98: Principles of Knowledge Representation and Reasoning*, pages 124–130. Morgan Kaufmann, San Francisco, California, 1998.

P Balbiani, J.F. Condotta, and L. Fariñas del Cerro. A new tractable subclass of the rectangle algebra. In *IJCAI*, pages 442–447, 1999.

E.E Barton and I. Buchanan. The polygon package. *Computer-Aided Design*, 12(1):3 – 11, January 1980.

B. G. Baumgart. Winged edge polyhedron representation. Technical report, Stanford University, 1972.

A. Belussi and M. Cristani. Mereological inheritance. *Spatial Cognition and Computation*, 2:467–494, 2000.

B. Bennett. Spatial reasoning with propositional logics. In Jon Doyle, Erik Sandewall, and Pietro Torasso, editors, *KR'94: Principles of Knowledge Representation and Reasoning*, pages 51–62. Morgan Kaufmann, San Francisco, California, 1994.

B. Bennett. Logics for topological reasoning. In *12th European Summer School in Logic, Language and Information (ESSLLI-MM)*, 2000.

B. Bennett. Application of supervaluation semantics to vaguely defined spatial concepts. *Lecture Notes in Computer Science*, 2205:108–123, 2001a.

B. Bennett. What is a forest? on the vagueness of certain geographic concepts. *Topoi - An International Review Of Philosophy*, 20(2):189–201, 2001b.

B. Bennett. Space, time, matter and things. In *FOIS '01: Proceedings of the international conference on Formal Ontology in Information Systems*, pages 105–116, New York, NY, USA, 2001c. ACM.

B. Bennett. Modes of concept definition and varieties of vagueness. *Applied Ontology*, 1 (1):17–26, 2005.

B. Bennett. A theory of vague adjectives grounded in relevant observables. In Patrick Doherty, John Mylopoulos, and Christopher A. Welty, editors, *Proceedings of the Tenth International Conference on Principles of Knowledge Representation and Reasoning*, pages 36–45. AAAI Press, 2006.

B. Bennett, A. Isli, and A. G. Cohn. A system handling RCC-8 queries on 2D regions representable in the closure algebra of half -planes. In *Proceedings of the 11th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems (IEA-AIE)*. LNCS (Berlin: Springer-Verlag), 1998.

B. Bennett, G. Sakellariou, and P. Santos. Supervaluation semantics for an inland water feature ontology. In L.P. Kaelbling and A. Saffiotti, editors, *International Joint Conference on Artificial Intelligence*, volume 19, pages 564–559, University of Edinburgh, Scotland, 2005.

B. Bennett, D. Mallenby, and A. Third. An ontology for grounding vague geographic terms. In C. Eschenbach and M. Grüninger, editors, *Fifth International Conference on Formal Ontology in Information Systems(FOIS 2008)*, pages 280 – 296. IOS Press, 2008.

J. L. Bentley and T. A. Ottmann. Algorithms for reporting and counting geometric intersections. *IEEE Transactions On Computers*, 28(9):643–647, 1979.

P. Blackburn and J. Bos. *Representation and Inference for Natural Language. A First Course in Computational Semantics*. Chicago University Press, Chicago, IL, USA, 2005.

H. Blum. Biological shape and visual science.1. *Journal Of Theoretical Biology*, 38(2): 205–287, 1973.

J. Bos. Exploring model building for natural language understanding. In *Proceedings of Fourth International Workshop on Inference in Computational Semantics (ICoS-4)*, Nancy, France, September 2003.

P. Bose, D. Bremner, and D. L. Souvaine. Computing the tool path of an externally monotone polygon in linear time. In P. Bose, editor, *CCCG*, Ottawa, Canada, 2006.

D. Cabeza and M. Hermenegildo. Distributed www programming using (ciao-)prolog and the pillow library. *Theory Pract. Log. Program.*, 1(3):251–282, 2001. ISSN 1471-0684.

W. M. Cameron and D. W. Pritchard. Estuaries. In M. N. Hill, editor, *The Sea - The Composition of Sea-Water Comparative and Descriptive Oceanography*, volume 2, pages 306 – 324. Harvard University Press, 1963.

L. Carroll. *Alice Through the Looking-Glass*. Macmillan Publishers, 1872.

F. Chin, J. Snoeyink, and C. A. Wang. Finding the medial axis of a simple polygon in linear time. *Discrete & Computational Geometry*, 21(3):405–420, 1999.

K. Claessen and N. Sörensson. New techniques that improve MACE-style model finding. In P. Baumgartner and C. Fermueller, editors, *Proceedings of the CADE-19 Workshop: Model Computation - Principles, Algorithms, Applications*, Miami, USA, 2003.

Keith L. Clark. Negation as failure. In *Logic and Data Bases*, pages 293–322, 1977.

E. Clementini, J. Sharma, and M. J. Egenhofer. Modelling topological spatial relations: Strategies for query processing. *Computers & Graphics*, 18(6):815 – 822, 1994.

A. G. Cohn and S. M. Hazarika. Qualitative spatial representation and reasoning: An overview. *Fundamenta Informaticae*, 46(1-2):1–29, 2001.

B. Cuenca Grau, I. Horrocks, B. Parsia, P. Patel-Schneider, and U. Sattler. Next steps for OWL. In B. Cuenca Grau, P. Hitzler, C. Shankey, and E. Wallace, editors, *OWL: Experiences and Directions*, Athens, Georgia, USA, 2006. CEUR Online Proceedings Series.

Martin Davis, George Logemann, and Donald Loveland. A machine program for theorem-proving. *Commun. ACM*, 5(7):394–397, 1962.

M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry*. Springer, February 2000.

Boris N. Delaunay. Sur la sphère vide. *Bulletin of Academy of Sciences of the USSR*, VII (6):793–800, 1934.

J. Dever, N. Asher, and C. Pappas. Supervaluations Debugged. *Mind*, 2008.

P. Dimitrov, C. Phillips, and K. Siddiqi. Robust and efficient skeletal graphs. In *IEEE Conference On Computer Vision And Pattern Recognition, Proceedings*, volume 1 of *Proceedings - IEEE Computer Society Conference On Computer Vision And Pattern Recognition*, pages 417–423. IEEE Computer Society, 2000.

C. Dolbear, G. Hart, J. Goodwin, S. Zhou, and K. Kovacs. The Rabbit language: description, syntax and conversion to OWL. *Ordnance Survey Research Labs Technical Report*, (IRI-0004), 2007.

D. Dubois, H. Prade, and H. Smets. Partial truth is not uncertainty. *IEEE Expert: Intelligent Systems and Their Applications*, 9(4):15–19, 1994.

D. Dubois, F. Esteva, L. Godo, and H. Prade. An information-based discussion of vagueness. In *10th IEEE International Conference On Fuzzy Systems, Vols 1-3 - Meeting The Grand Challenge: Machines That Serve People*, pages 781–784. IEEE PRESS, 2001.

M. Duckham, K. Mason, J. Stell, and M. Worboys. A formal approach to imperfection in geographic information. *A Formal Approach to Imperfection in Geographic Information*, 25:89–103, 2001.

M. J. Egenhofer. Reasoning about binary topological relations. *Lecture Notes In Computer Science*, 525:144–160, 1991.

M. J. Egenhofer and K. K. Al-Taha. Reasoning about gradual changes of topological relationships. In A.U. Frank, I. Campari, and U. Formentini, editors, *Spatio-Temporal Reasoning*, volume 639 of *Lecture Notes in Computer Science*, pages 196 – 219. Springer, 1992.

M. J. Egenhofer and R. D. Franzosa. Point-set topological spatial relations. *International Journal Of Geographical Information Systems*, 5(2):161–174, 1991.

A. Ergin, E. Karaesmen, A. Micallef, and A. T. Williams. A new methodology for evaluating coastal scenery: fuzzy logic systems. *Area*, 36(4):367–386, 2004.

C. G. Fermuller and R. Kosik. Combining supervaluation and degree based reasoning under vagueness. In *Logic For Programming, Artificial Intelligence, And Reasoning, Proceedings*, volume 4246 of *Lecture Notes In Artificial Intelligence*, pages 212–226. Springer-Verlag Berlin, Berlin, 2006.

K. Fine. Vagueness, truth and logic. *Synthese*, 30:265–300, 1975.

P. Fisher. Sorites paradox and vague geographies. *Fuzzy Sets And Systems*, 113(1):7–18, 2000.

J. A. Fodor and E. Lepore. What cannot be evaluated cannot be evaluated, and it cannot be supervalued either. *Journal of Philosophy*, 93(10):516–535, 1996.

J. D. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes. *Computer Graphics: Principles and Practice in C*. Addison-Wesley Professional, second edition, August 1995.

F. Fonseca, M.J. Egenhofer, C. Agouris, and G. Câmara. Using ontologies for integrated geographic information systems. *Transactions in Geographic Information Systems*, 6 (3):231–257, 2002.

Steven Fortune. A sweepline algorithm for Voronoi diagrams. *Algorithmica*, 2:153–174, 1987.

A. Galton. A formal theory of objects and fields. In Daniel R. Montello, editor, *COSIT*, volume 2205 of *Lecture Notes in Computer Science*, pages 458–473. Springer, 2001.

A. Galton and M. Duckham. What is the region occupied by a set of points? In M. Raubal, H. J. Miller, A. U. Frank, and M. F. Goodchild, editors, *GIScience*, volume 4197 of *Lecture Notes in Computer Science*, pages 81–98. Springer, 2006. ISBN 3-540-44526-9.

B. Ganter and R. Wille. *Formal Concept Analysis*. Springer Verlag, 1999.

Y. R. Ge and J. M. Fitzpatrick. Extraction of maximal inscribed disks from discrete Euclidean distance maps. In *1996 IEEE Computer Society Conference On Computer Vision And Pattern Recognition*, pages 556–561. IEEE PRESS, 1996.

M. Giritli. Who can connect in RCC? In *KI 2003: Advances In Artificial Intelligence*, volume 2821 of *Lecture Notes In Artificial Intelligence*, pages 565–579. Springer-Verlag Berlin, Berlin, 2003.

R. Grütter and B. Bauer-Messmer. Towards spatial reasoning in the semantic web: A hybrid knowledge representation system architecture. In S. Fabrikant and M. Wachowicz, editors, *The European Information Society*, volume XVII of *Lecture Notes in Geoinformation and Cartography*, pages 349 – 364. Springer, 2007.

N. Guarino. Formal ontology and information systems. In N. Guarino, editor, *Proceedings of the 1st International Conference on Formal Ontologies in Information Systems (FOIS '98)*, pages 3 – 15, Amsterdam, The Netherlands, 1998. IOS Press.

N. Guarino and C. Welty. Evaluating ontological decisions with ontoclean. *Communications Of The ACM*, 45(2):61–65, 2002. ISI Document Delivery No.: 515HL.

H.W. Güsgen. Spatial Reasoning Based on Allen's Temporal Logic. Technical Report TR-89-049, International Computer Science Institute, 1947 Center St, Suite 600, Berkeley, CA, 1989.

V. Haarslev and R. Möller. Racer system description. In *IJCAR 2001*, pages 701 – 705. Springer, 2001.

V. Haarslev, C. Lutz, and R. Möller. Foundations of spatioterminological reasoning with description logics. In S.C. Shapiro, editor, *Principles of Knowledge Representation and Reasoning*, pages 112–123. Morgan Kaufman, 1998.

S. Harnad. The symbol grounding problem. *Physica D*, 42(1-3):335–346, 1990.

T. Hatzichristos and M. Giaoutzi. Landfill siting using GIS, fuzzy logic and the delphi method. *International Journal of Environmental Technology and Management*, 6:218–231, November 2005.

S. M. Hazarika and A. G Cohn. Taxonomy of spatio-temporal vagueness: An alternative egg-yolk interpretation. In D.R. Montello, editor, *COSIT/FOIS Workshop on Spatial Vagueness, Uncertainty and Granularity*, Maine, USA, 2001. Springer.

M. Held. VRONI: An engineering approach to the reliable and efficient computation of voronoi diagrams of points and line segments. *Computational Geometry –Theory And Applications*, 18(2):95–123, 2001.

C. E. Herdendorf. Great lakes estuaries. *Estuaries*, 13(4):493–503, 1990.

W. H. Hesselink, M. Visser, and J. B. T. M. Roerdink. Euclidean skeletons of 3D data sets in linear time by the integer medial axis transform. In *Mathematical Morphology: 40 Years On*, volume 30 of *Computational Imaging And Vision*, pages 259–268. Springer-Verlag, 2005.

W. Itonaga, I. Matsuda, N. Yoneyama, and S. Ito. Automatic extraction of road networks from map images. *Electronics And Communications In Japan Part II-Electronics*, 86 (4):62–72, 2003.

Joxan Jaffar and Michael J. Maher. Constraint logic programming: A survey. *Journal of Logic Programming*, 19/20:503–581, 1994.

A. Jakulin and D. Mladenić. Ontology grounding. In *Proceedings of the 8th International multi-conference Information Society IS-2005*, pages 170–173, Ljubljana, Slovenia, 2005.

A. Kalyanpur, B. Parsia, E. Sirin, B. Cuenca Grau, and J.A. Hendler. Swoop: A web ontology editing browser. *J. Web Sem.*, 4(2):144–153, 2006.

Y. Katz and B.C. Grau. Representing qualitative spatial information in OWL-DL. In *Proceedings of the OWL: Experiences and Directions Workshop. Galway*, 2005.

Michael Kohlhase and Alexander Koller. Resource-adaptive model generation as a performance model. *Logic Journal of the IGPL*, 11(4):435–456, 2003.

L. Kulik. *Vague Spatial Reasoning based on Supervaluation*, volume 19 of *Geographical Domain and Geographical Information Systems - GeoInfo Series*. Institute for Geoinformation, Vienna University of Technology, Vienna, Vienna: Institue for Geoinformation, Vienna University of Technology, 2000.

L. Kulik. Spatial vagueness and second-order vagueness. *Spatial Cognition & Computation*, 3(2&3):157–183, 2003.

D. T. Lee. Medial axis transformation of a planar shape. *IEEE Transactions On Pattern Analysis And Machine Intelligence*, 4(4):363–369, 1982.

Douglas B. Lenat. CYC: a large-scale investment in knowledge infrastructure. *Commun. ACM*, 38(11):33–38, 1995.

S. Li and B. Nebel. Qualitative spatial representation and reasoning: A hierarchical approach. *Comput. J.*, 50(4):391–402, 2007.

T. Liebig and O. Noppens. ONTOTRACK: A semantic approach for ontology authoring. *Journal of Web Semantics*, 3(2-3):116–131, October 2005.

L. Liu and L. Chen. A modal supervaluation description logic for characterization of vague concepts: its semantics and a tableau algorithm for it. *Logic Journal Of The IGPL*, 14(6):873–888, 2006.

D. Mallenby. Grounding a geographic ontology on geographic data. In E. Amir, V. Lifschitz, and R. Miller, editors, *8th International Symposium on Logical Formalizations of Commonsense Reasoning (Commonsense 2007)*, pages 101 – 105, Stanford, USA, 2007.

D. Mallenby and B. Bennett. Applying spatial reasoning to topographical data with a grounded geographical ontology. In F. Fonseca and M. A. Rodríguez, editors, *The Second International Conference on GeoSpatial Semantics (GeoS 2007)*, LNCS 4853, pages 210 – 227, Mexico City, Mexico, 2007.

M. Mantyla. *Introduction to Solid Modeling*. W. H. Freeman & Co., 1988. 60949.

M. McAllister and J. Snoeyink. Medial axis generalization of river networks. *CaGIS*, 27 (2):129 –138, 2000. Excellent paper showing how Medial Axis is useful for rivers.

W. McCune. MACE4 reference manual and guide. *The Computing Research Repository (CoRR)*, cs.SC/0310055, 2003.

A. Meijster, J. Roerdink, and W. H. Hesselink. A general algorithm for computing distance transforms in linear time. In *Mathematical Morphology And Its Applications To Image And Signal Processing*, volume 18 of *Computational Imaging And Vision*, pages 331–340. Kluwer Academic Publishers, 2000.

M.E. Mortenson. *Geometric modeling (2nd ed.)*. John Wiley & Sons, Inc., New York, NY, USA, 1997.

J.A.C. Paiva and M.J. Egenhofer. Robust inference of the flow direction in river networks. *Algorithmica*, (in press).

J.A.C. Paiva, M.J. Egenhofer, and A.U. Frank. Spatial reasoning about flow directions: Towards an ontology for river networks. *XVII International Congress for Photogrammetry and Remote Sensing*, 24:318–224, 1992.

G. K. Palshikar. Fuzzy region connection calculus in finite discrete space domains. *Applied Soft Computing*, 4(1):13–23, 2004.

A. Porter, A. Sadek, and N. Hayden. Fuzzy geographic information systems for phytoremediation plant selection. *Journal Of Environmental Engineering-ASCE*, 132(1): 120–128, 2006.

I. Pratt and D. Schoop. A complete axiom system for polygonal mereotopology of the real plane. *Journal of Philosophical Logic*, 27(6):621–661, 1998.

F. P. Preparata and M.l I. Shamos. *Computational Geometry: An Introduction (Monographs in Computer Science)*. Springer, August 1985.

D. Pritchard. What is an estuary: Physical viewpoint. In G.H. Lauff, editor, *Estuaries*, number SS-83 in AAAS Publications, pages 3–5. Washington, DC: AAAS, 1967a.

D. Pritchard. Observations of circulation in coastal plain estuaries. In G.H. Lauff, editor, *Estuaries*, number SS-83 in AAAS Publications, pages 37–44. Washington, DC: AAAS, 1967b.

D. A. Randell, Z. Cui, and A. G. Cohn. A spatial logic-based on regions and connection. In *Principles Of Knowledge Representation And Reasoning: Proceedings Of The Third International Conference (KR 92)*, pages 165–176. Morgan Kaufmann Pub Inc, San Mateo, 1992.

R. Raskin, M. Pan, and C. Mattmann. Enabling semantic interoperability for earth science data. In *NASA Earth Science Technology Conference (ESTC)*, Palo Alto, CA, June 2004.

R. G. Raskin and M.J. Pan. Knowledge representation in the semantic web for earth and environmental terminology (SWEET). *Computer & Geosciences*, 31:1119–1125, 2005.

E. Remy and E. Thiel. Medial axis for chamfer distances: Computing look-up tables and neighbourhoods in 2D or 3D. *Pattern Recognition Letters*, 23(6):649–661, 2002.

E. Remy and E. Thiel. Look-up tables for medial axis on squared euclidean distance transform. In *Discrete Geometry For Computer Imagery, Proceedings*, volume 2886 of *Lecture Notes In Computer Science*, pages 224–235. Springer, 2003.

E. Remy and E. Thiel. Exact medial axis with Euclidean distance. *Image And Vision Computing*, 23(2):167–175, 2005.

D. Reniers and A. Telea. Quantitative comparison of tolerance-based feature transforms. In *Proc. VISAPP'06*, pages 107–114, INSTICC Press, 2006.

J. Renz and B. Nebel. On the complexity of qualitative spatial reasoning: A maximal tractable fragment of the region connection calculus. *Artificial Intelligence*, 108(1-2): 69–123, 1999.

M. A. Rodríguez, M. J. Egenhofer, and A. D. Blaser. Query pre-processing of topological constraints: Comparing a composition-based with neighborhood-based approach. In *Advances In Spatial And Temporal Databases, Proceedings*, volume 2750 of *Lecture Notes In Computer Science*, pages 362–379, 2003.

K. Schild. A correspondence theory for terminological logics: preliminary report. In *Proceedings of IJCAI-91, 12th International Joint Conference on Artificial Intelligence*, pages 466–471, Sidney, AU, 1991.

J. R. Schubel and D. W. Pritchard. Great lakes estuaries: Phooey. *Estuaries*, 13(4): 508–509, 1990.

J. Simpson and E. Weiner, editors. *Concise Oxford English Dictionary*. Oxford University Press, 11 edition, 2004.

E. Sirin, B. Parsia, B. Cuenca Grau, A. Kalyanpur, and Y. Katz. Pellet: A practical OWL-DL reasoner. *Web Semant.*, 5(2):51–53, 2007.

B. Smith. On drawing lines on a map. In *Spatial Information Theory - Proceedings of COSIT'95*, pages 475 – 484. Springer, 1995.

B. Smith. Fiat objects. *Topoi*, 20(2):131 – 148, 2001.

B. Smith and D. M. Mark. Ontology and geographic kinds. In T. Poiker and N. Chrisman, editors, *Proceedings of the Tenth International Symposium on Spatial Data Handling*, pages 308–320, Burnaby, BC, Simon Fraser University, 1998.

B. Smith and A.C. Varzi. Fiat and bona fide boundaries: Towards on ontology of spatially extended objects. In *COSIT '97: Proceedings of the International Conference on Spatial Information Theory*, pages 103–119, London, UK, 1997. Springer-Verlag.

J. Snow. *On the Mode of Communication of Cholera*. John Churchill, New Burlington Street, London, second edition, 1855.

E. Stefanakis, M. Vazirgiannis, and T. Sellis. Incorporating fuzzy logic methodologies into GIS operations. In *International Conference on Geographical Information Systems in Urban, Regional and Environmental Planning*, pages 61 – 68, 1996.

S. Suri, P.M. Hubbard, and J. F. Hughes. Analyzing bounding boxes for object intersection. *ACM Trans. Graph.*, 18(3):257–277, 1999.

M. Taddeo and L. Floridi. Solving the symbol grounding problem: a critical review of fifteen years of research. *Journal Of Experimental & Theoretical Artificial Intelligence*, 17(4):419–445, 2005. Review 0952-813X.

M. P. Taylor and R. Stokes. When is a river not a river? consideration of the legal definition of a river for geomorphologists practising in New South Wales, Australia. *Australian Geographer*, 36(2):183–200, 2005.

A. Telea. An augmented fast marching method for computing skeletons and centerlines. In D. Ebert, P. Brunet, and I. Navazo, editors, *Proceedings of the symposium on Data Visualisation*, volume 22, pages 251 – 259, Barcelona, Spain, 2002. ACM Press.

A. Third, B. Bennett, and D. Mallenby. Architecture for a grounded ontology of geographic information. In F. Fonseca and M. A. Rodríguez, editors, *The Second International Conference on GeoSpatial Semantics (GeoS 2007)*, LNCS 4853, pages 36–50, Mexico City, Mexico, 2007.

E. Tomai and M. Kavouras. From "onto-geonoesis" to "onto-genesis": The design of geographic ontologies. *Geoinformatica*, 8(3):285–302, 2004.

D. Tsarkov and I. Horrocks. Efficient reasoning with range and domain constraints. In *2004 Description Logic Workshop (DL 2004)*, 2004.

A. C. Varzi. Vagueness, logic, and ontology. *The Dialogue. Yearbooks for Philosophical Hermeneutics 1*, pages 135–154, 2001a.

A. C. Varzi. Vagueness in geography. *Philosophy & Geography*, 4(1):49–65, 2001b.

Georges Voronoï. Nouvelles applications des paramètres continus à la théorie des formes quadratiques. Premier mémoire: Sur quelques propriétées des formes quadratiques positives parfaites. *Journal für die Reine and Angewandte Mathematik*, 133:97–178, 1907.

B. B. Welch. *Practical Programming in Tcl and TK*. Prentice-Hall, 3rd edition, 2000.

M. Wessel. On spatial reasoning with description logics - position paper. In I. Horrocks and S. Tessaris, editors, *Proceedings of the International Workshop on Description Logics (DL 2002)*, pages 156–163, 2002.

J. Wielemaker. An optimised semantic web query language implementation in prolog. In *Logic Programming, Proceedings*, volume 3668 of *Lecture Notes In Computer Science*, pages 128–142. Springer-Verlag Berlin, Berlin, 2005. ISI Document Delivery No.: BDF61 English.

J. Wielemaker, M. Hildebrand, and J. van Ossenbruggen. Using Prolog as the fundament for applications on the semantic web. 287:84–98, 2007.

T. Williamson. *Vagueness*. Routledge, London and New York, 1994.

T. Williamson. On the structure of higher-order vagueness. *Mind*, 108(429):127–143, 1999.

L. A. Zadeh. Fuzzy sets. *Information and Control*, 8(3):338–353, 1965.

L. A. Zadeh. Fuzzy algorithms. *Information and Control*, 12(2):94–102, 1968.

L. A. Zadeh. Information and fuzzy sets. *Proceedings Of The American Society For Information Science*, 13:83–83, 1976.

B. Žalik. Two efficient algorithms for determining intersection points between simple polygons. *Computers & Geosciences*, 26(2):137 – 151, March 2000.

Jian Zhang and Hantao Zhang. System description: Generating models by SEM. In *CADE-13: Proceedings of the 13th International Conference on Automated Deduction*, pages 308–312, London, UK, 1996. Springer-Verlag.

Y. Zhou and S. Suri. Analysis of a bounding box heuristic for object intersection. *J. ACM*, 46(6):833–857, 1999.

Appendix A

Results using Humber Estuary Data-Set

This Appendix shows the results of using the Humber Estuary dataset with GEOLOG.

A.1 Humber Estuary - Linearity With Respect to Centre

Figures A.1 - A.3 show the results of running the linearity query at different thresholds. Linear segments are marked in blue.

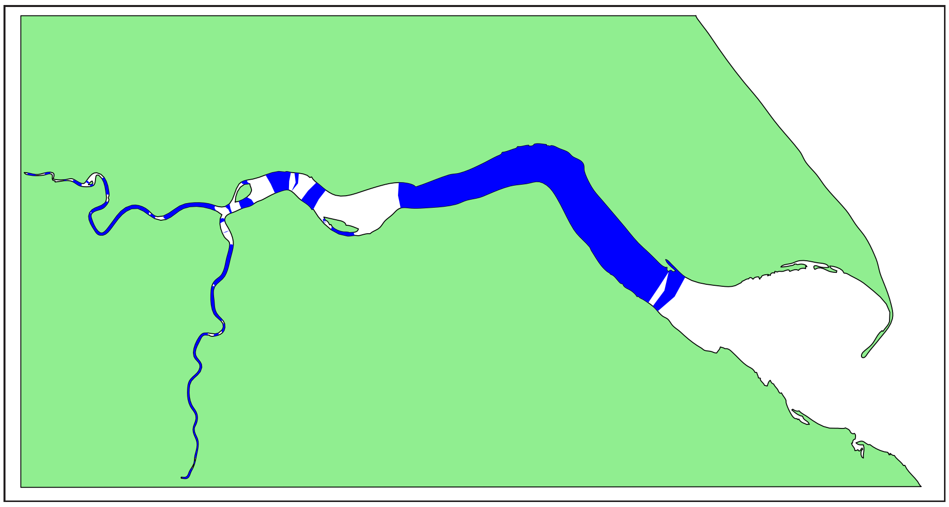


Figure A.1: The results of performing the linearity segmentation on the Humber Estuary using a threshold of 1.2.

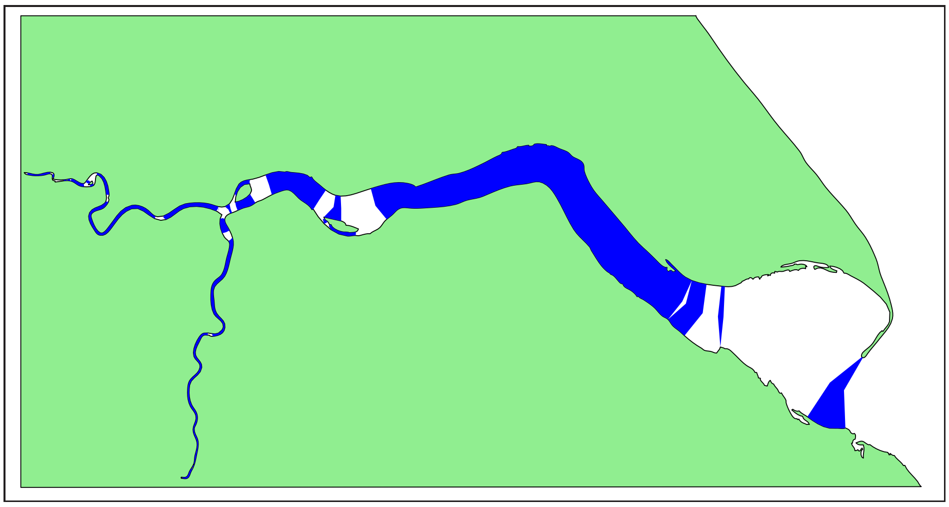


Figure A.2: The results of performing the linearity segmentation on the Humber Estuary using a threshold of 1.3.

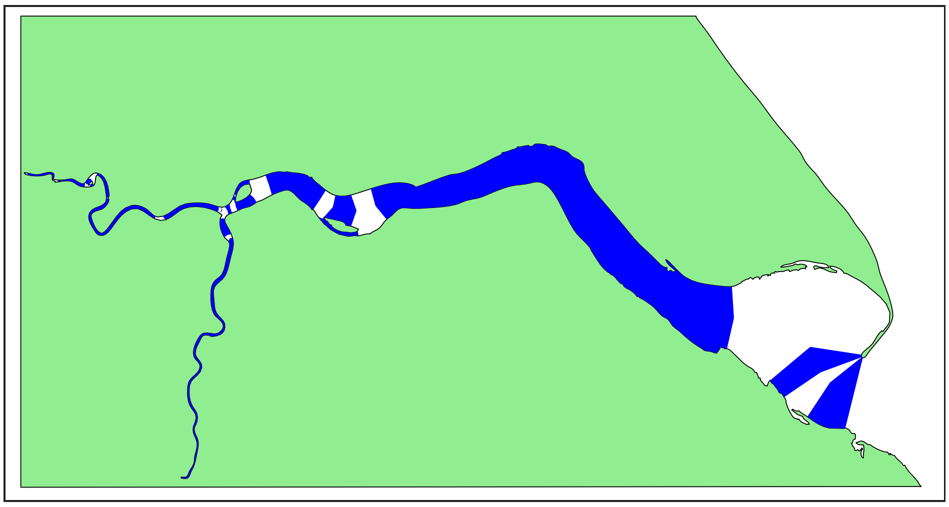


Figure A.3: The results of performing the linearity segmentation on the Humber Estuary using a threshold of 1.4.

A.2 Humber Estuary - Linearity With Respect to Edges

Figures A.4 - A.6 show the results of running the linearity with respect to the edges query at different thresholds. Segments that are both linear with respect to centre and edges are marked in blue.

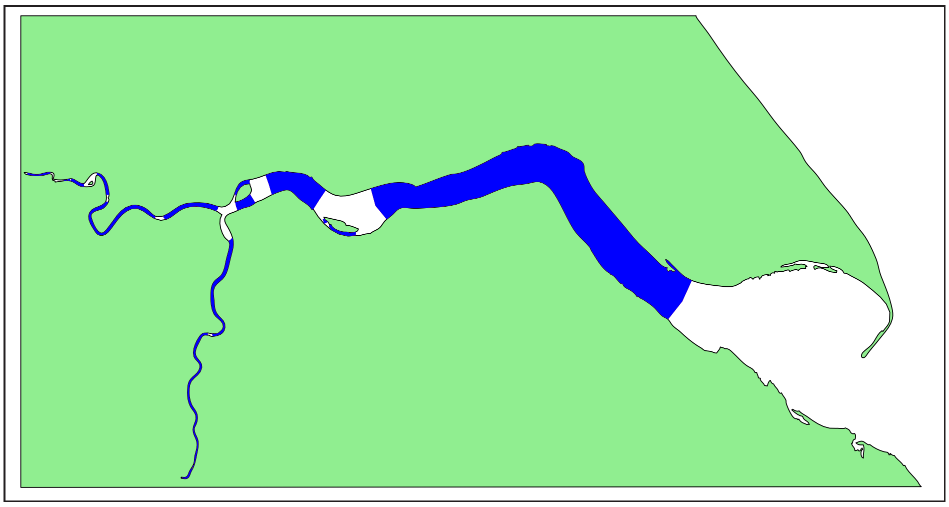


Figure A.4: The results of performing the linearity segmentation with respect to edges on the Humber Estuary using a threshold of 1.2.

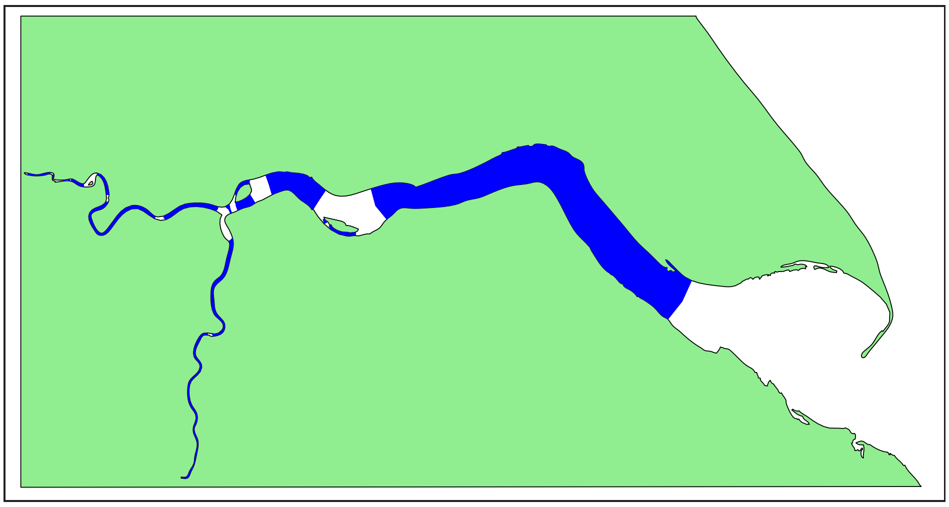


Figure A.5: The results of performing the linearity segmentation with respect to edges on the Humber Estuary using a threshold of 1.3.

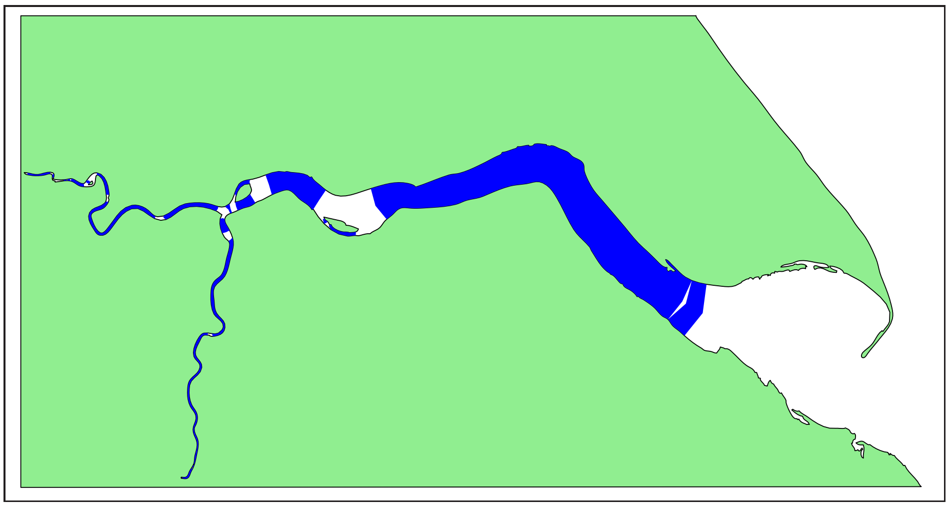


Figure A.6: The results of performing the linearity segmentation with respect to edges on the Humber Estuary using a threshold of 1.4.

A.3 Humber Estuary - Interstretches

Figures A.7 - A.8 show the results of running the interstretch query at different thresholds. Linear segments are marked in blue, interstretches are marked in red.

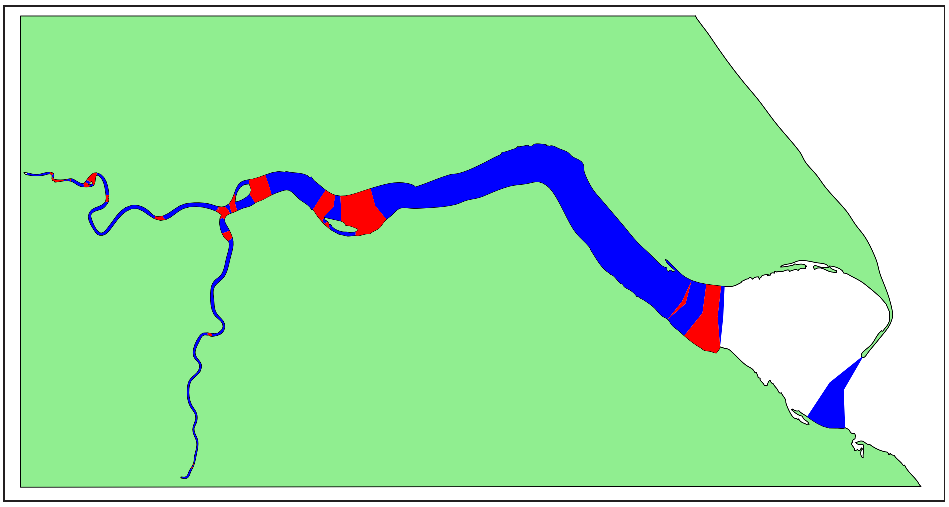


Figure A.7: The results of marking interstretches on the Humber Estuary using a threshold of 1.0 and 3.0 (the results were the same for both thresholds).

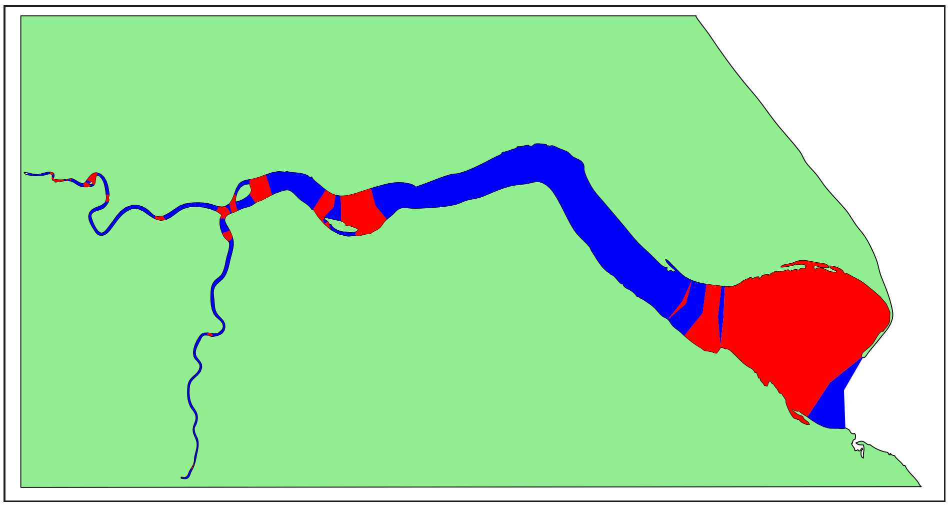


Figure A.8: The results of marking interstretches on the Humber Estuary using a threshold of 5.0.

A.4 Humber Estuary - Island and Island-Water

Figures A.9 - A.11 show the results of running the island and island-water queries at different thresholds. Islands are marked in red, island-water regions are marked in blue.

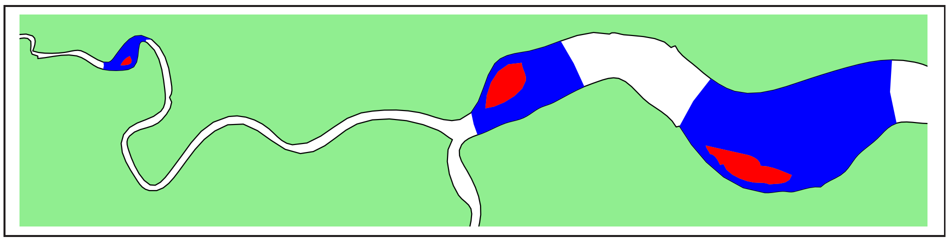


Figure A.9: The results of marking islands and island-water regions on the Humber Estuary using a linearity threshold of 1.2.

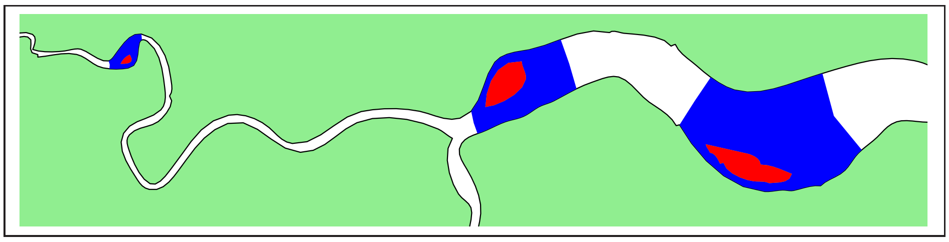


Figure A.10: The results of marking islands and island-water regions on the Humber Estuary using a linearity threshold of 1.3.

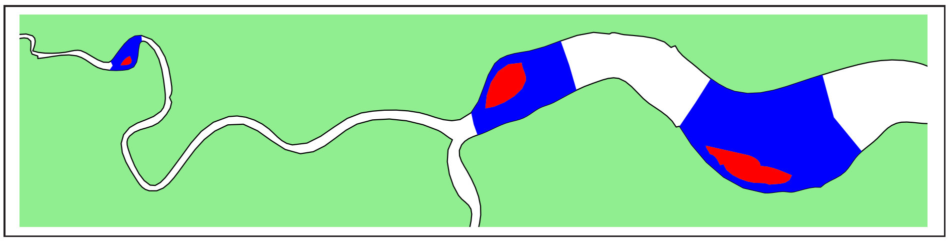


Figure A.11: The results of marking islands and island-water regions on the Humber Estuary using a linearity threshold of 1.4.

A.5 Humber Estuary - Confluence

Figures A.12a - A.12c show the results of running the confluence query at different thresholds. Confluence regions are marked in blue.

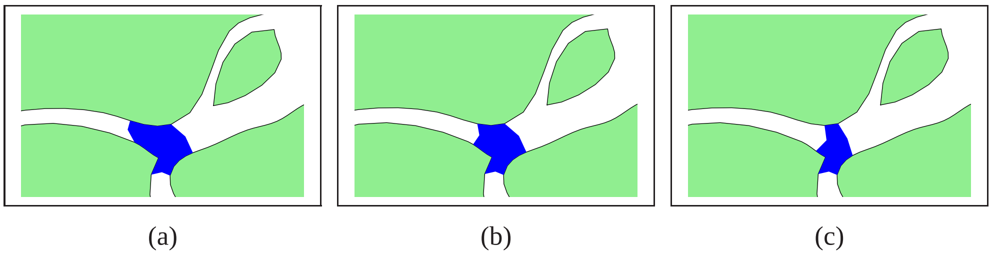


Figure A.12: The results of marking confluences on the Humber Estuary using varying linearity thresholds.

A.6 Humber Estuary - Major Stretch

Figures A.13 - A.15 show the results of running the confluence query at different thresholds. Major stretches are marked in blue.

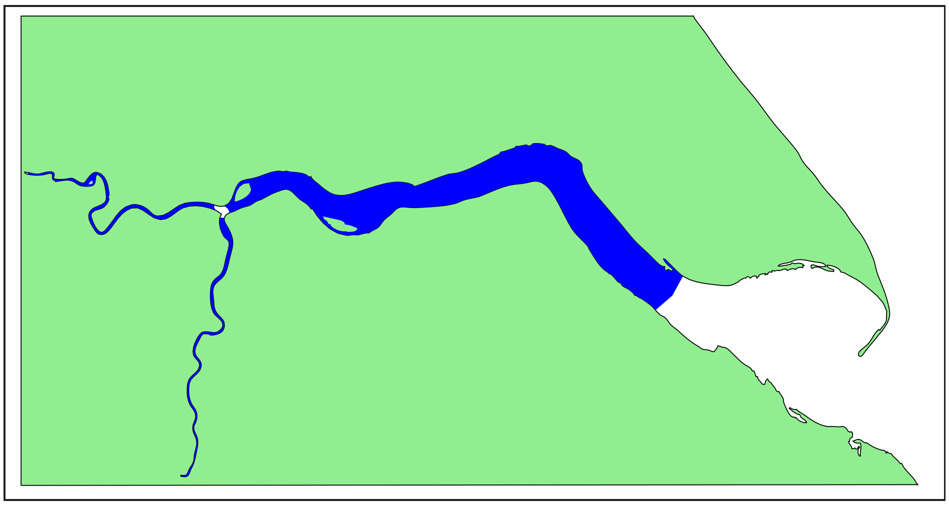


Figure A.13: The results of marking major stretches on the Humber Estuary using a linearity threshold of 1.2.

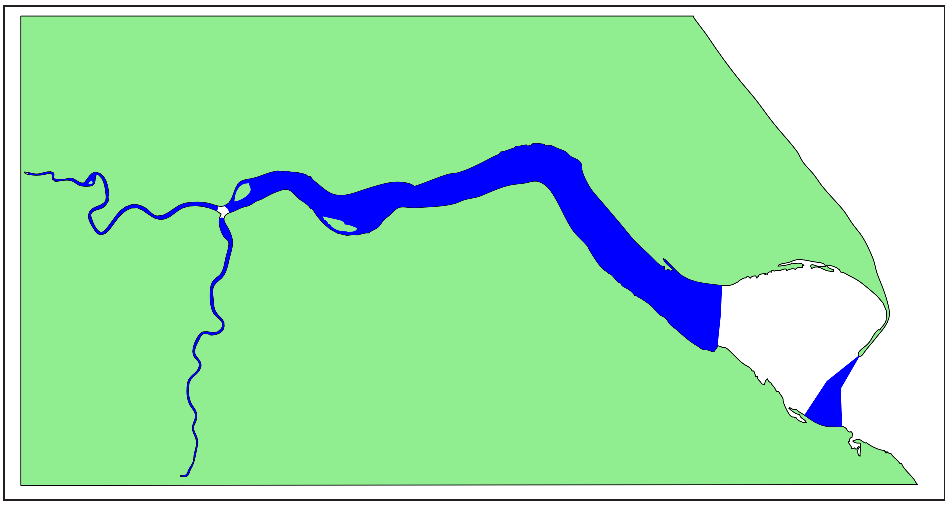


Figure A.14: The results of marking major stretches on the Humber Estuary using a linearity threshold of 1.3.

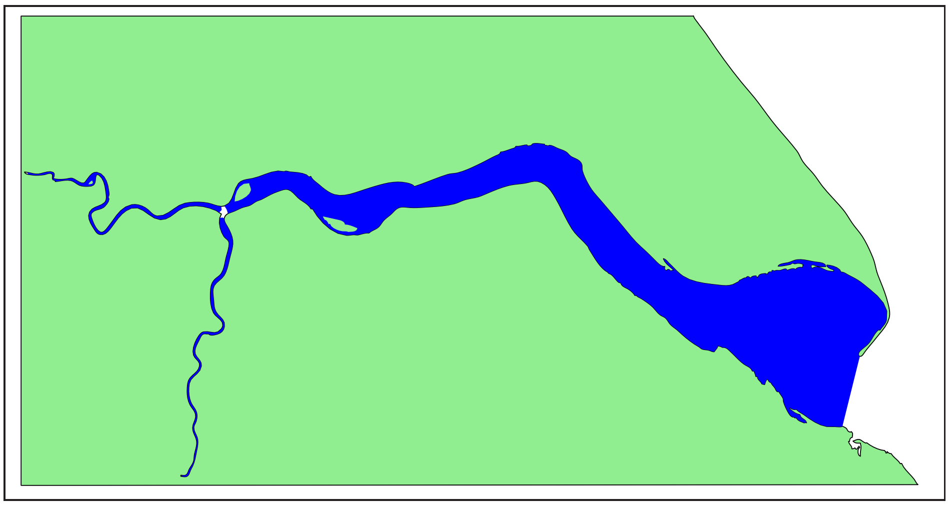


Figure A.15: The results of marking major stretches on the Humber Estuary using a linearity threshold of 1.4.

Appendix B

Results using River Tyne Data-Set

This Appendix shows the results of using the River Tyne Data-Set with GEOLOG. The queries used are discussed in Chapter 7, as well as the different threshold values used for vague predicates in each query.

B.1 River Tyne - Linearity with Respect to Centre

Figures B.1 - B.3 show the results of running the linearity query at different thresholds. Linear segments are marked in blue.

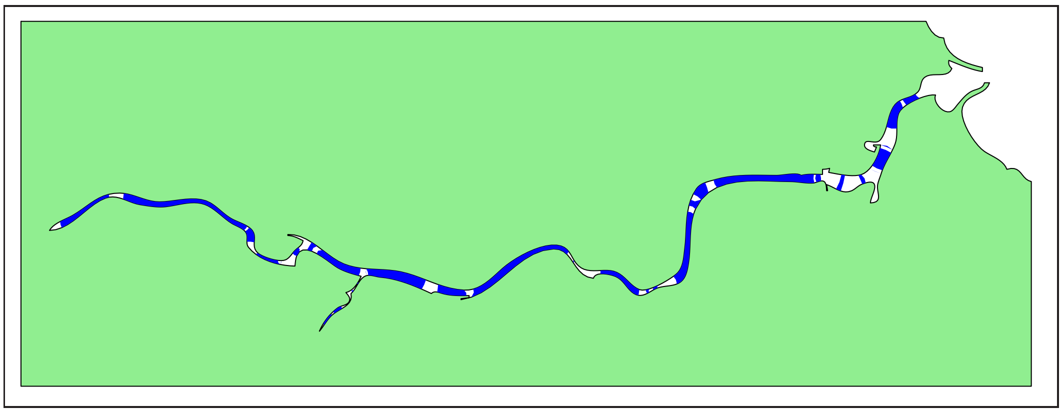


Figure B.1: The results of performing the linearity segmentation on the River Tyne using a threshold of 1.2.

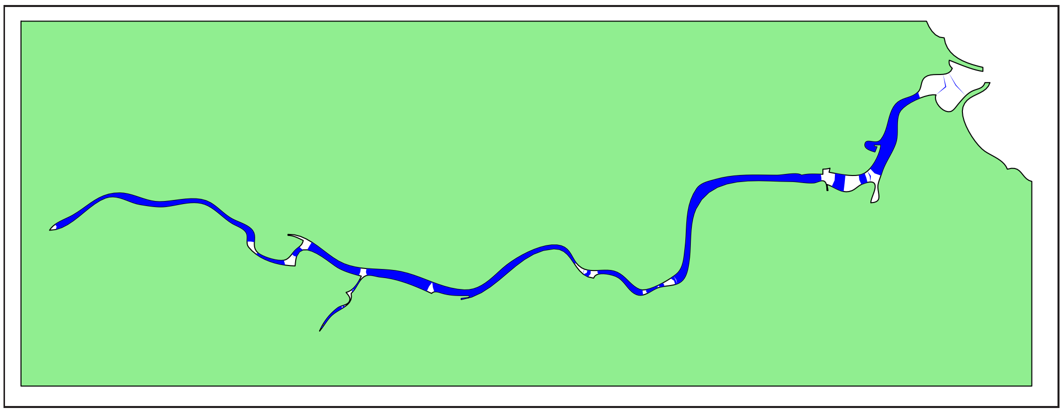


Figure B.2: The results of performing the linearity segmentation on the River Tyne using a threshold of 1.3.

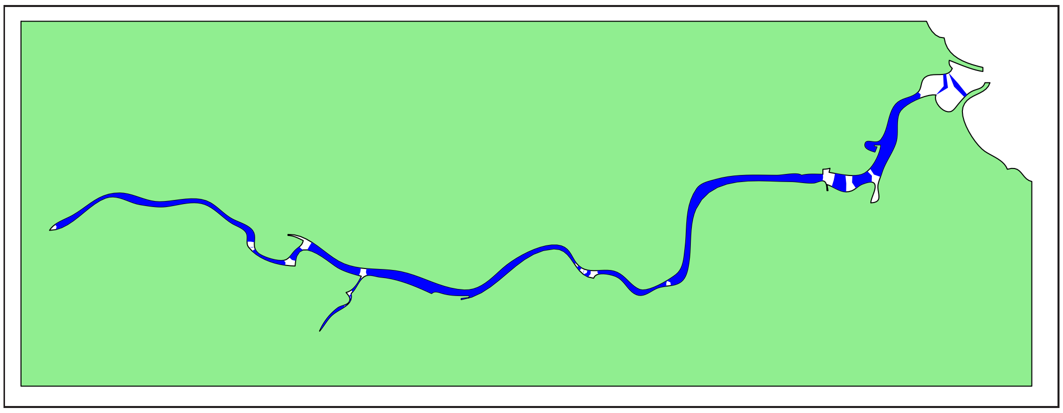


Figure B.3: The results of performing the linearity segmentation on the River Tyne using a threshold of 1.4.

B.2 River Tyne - Linearity with Respect to Edges

Figures B.4 and B.5 show the results of running the linearity with respect to the edges query at different thresholds. Linear segments are marked in blue.

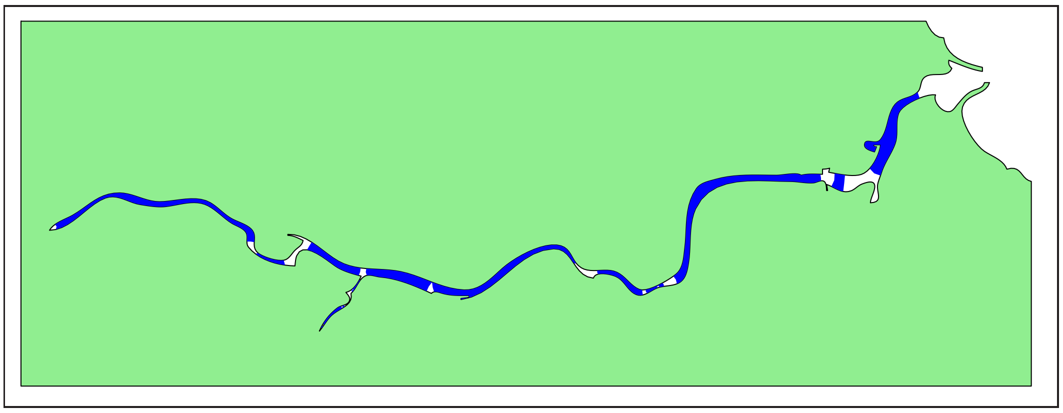


Figure B.4: The results of performing the linearity with respect to edges segmentation on the River Tyne using a threshold of 1.2.

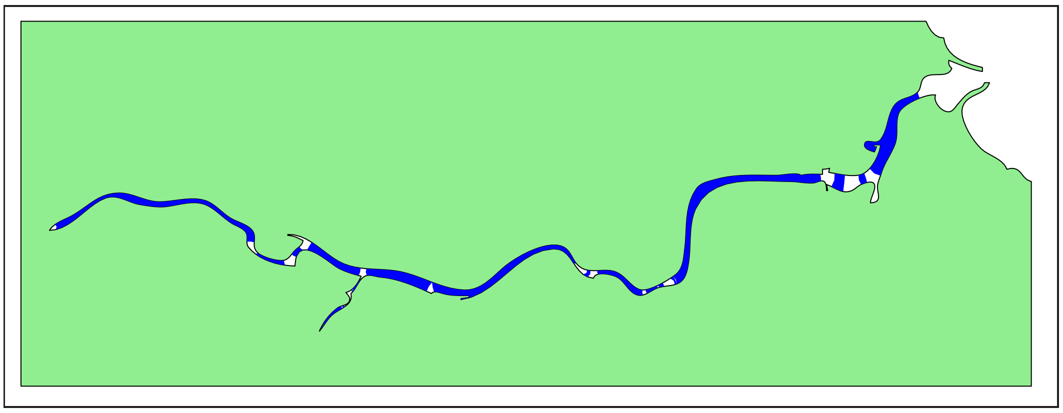


Figure B.5: The results of performing the linearity with respect to edges segmentation on the River Tyne using thresholds of 1.3 and 1.4 (no difference in the results from the two thresholds).

B.3 River Tyne - Interstretches

Figure B.6 show the results of running the *interstretch* query at all thresholds. Linear segments are marked in blue, *interstretches* are marked in red.

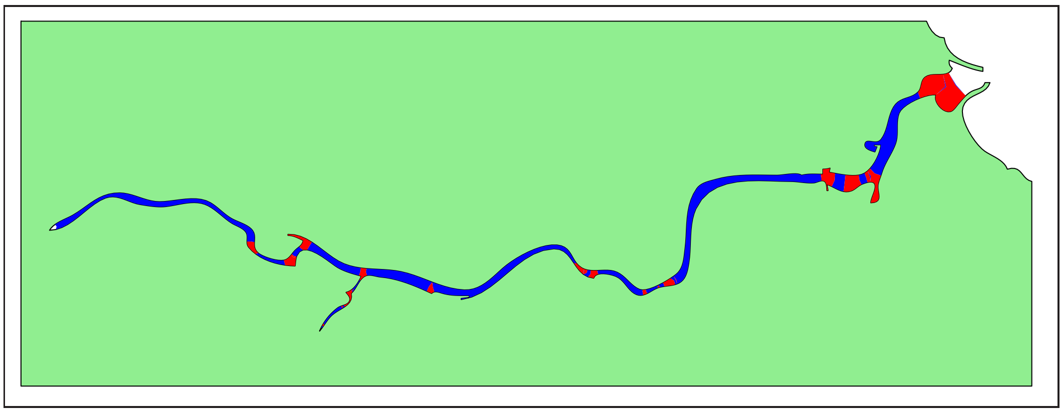


Figure B.6: The results of marking interstretches on the River Tyne using threshold values 1.0 - 5.0.

B.4 River Tyne - Confluence

Figures A.12a - A.12c show the results of running the confluence query at different thresholds. The confluence regions is marked in blue.

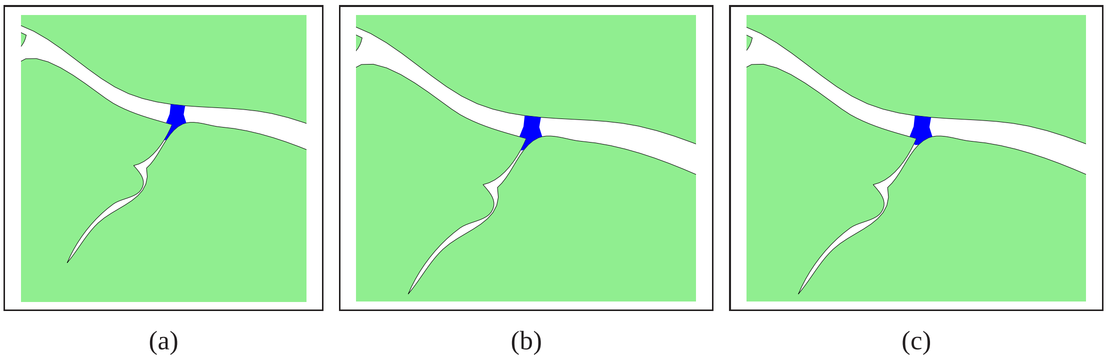


Figure B.7: The results of marking confluences on the River Tyne using varying linearity thresholds.

B.5 River Tyne - Major Stretches

Figures B.8 - B.10 show the results of running the major stretches query at different thresholds. Major stretches are marked in blue.

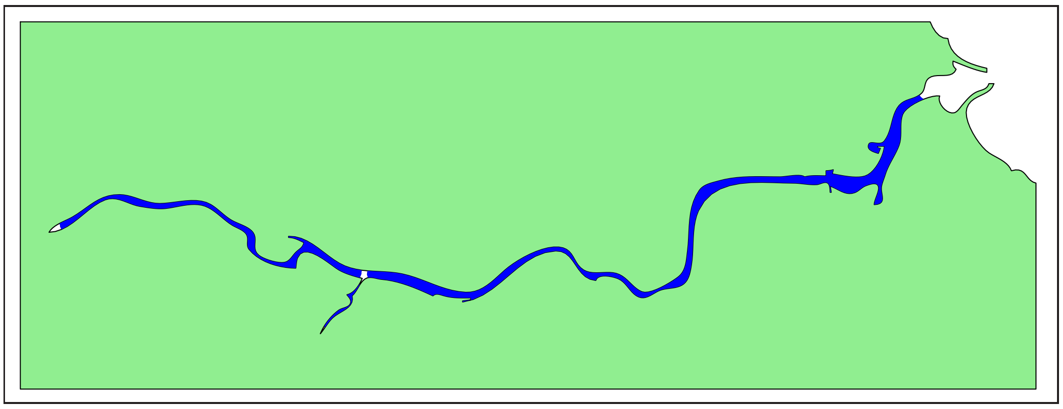


Figure B.8: The results of determining the major stretches on the River Tyne using a threshold of 1.2.

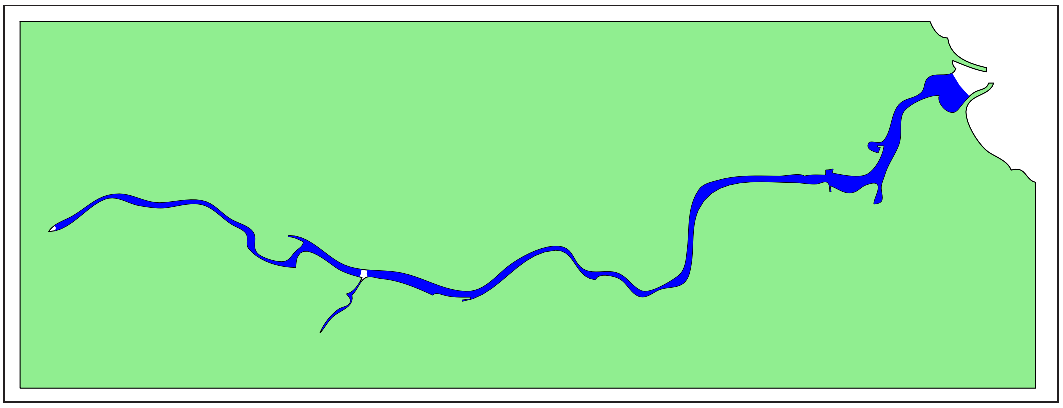


Figure B.9: The results of determining the major stretches on the River Tyne using a threshold of 1.3.

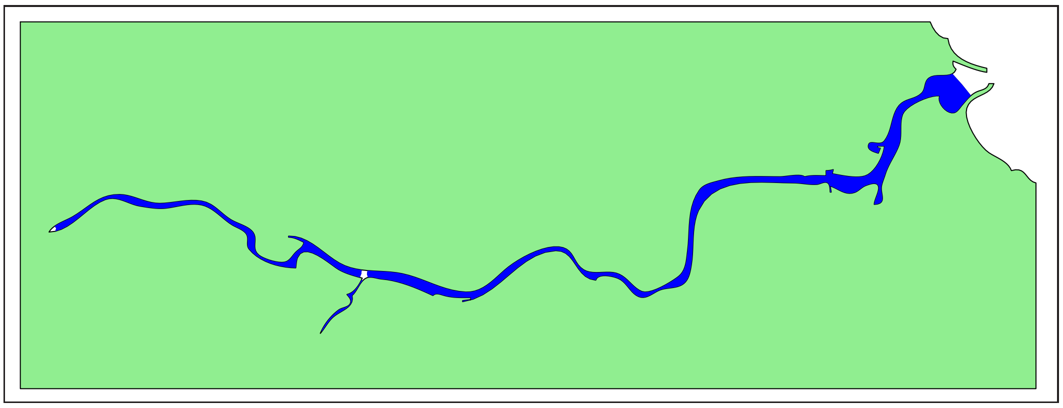


Figure B.10: The results of determining the major stretches on the River Tyne using a threshold of 1.4.

Appendix C

Results using Stour-Orwell Estuary Data-Set

This Appendix shows the results of using the Stour-Orwell Estuary Data-Set with GEOLOG. The queries used are discussed in Chapter 7, as well as the different threshold values used for vague predicates in each query.

C.1 Stour-Orwell Estuary - Linearity With Respect to Centre

Figures C.1 - C.3 show the results of running the linearity query at different thresholds. Linear segments are marked in blue.

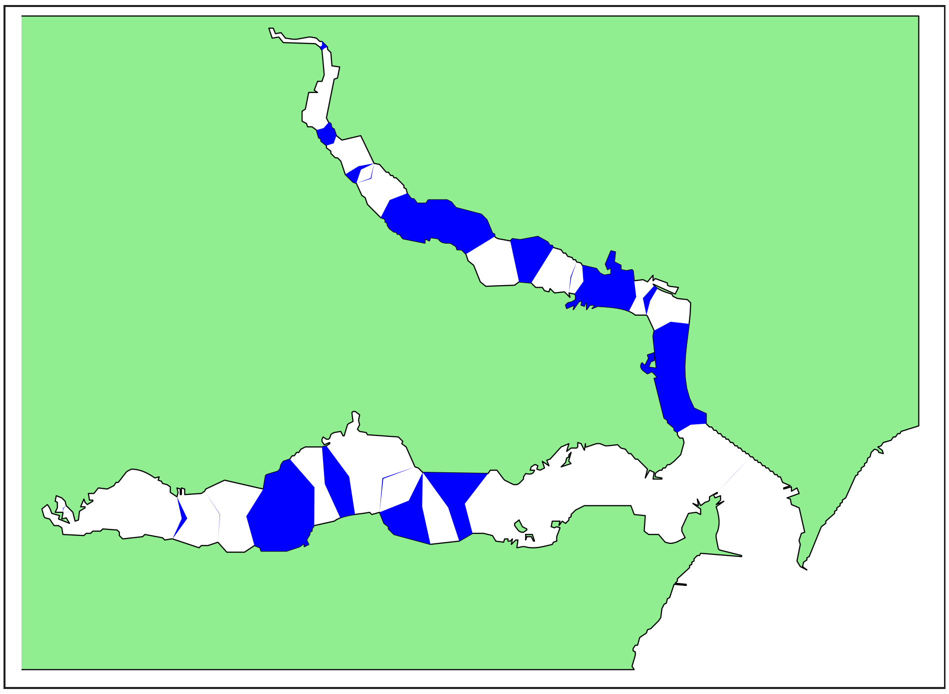


Figure C.1: The results of performing the linearity segmentation on the Stour-Orwell Estuary using a threshold of 1.2.

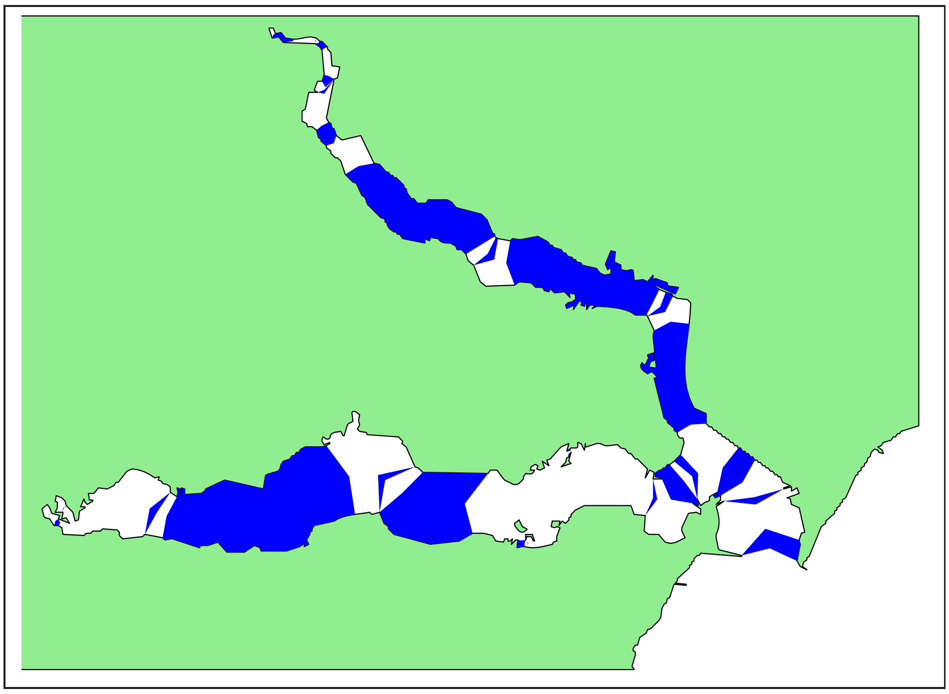


Figure C.2: The results of performing the linearity segmentation on the Stour-Orwell Estuary using a threshold of 1.3.

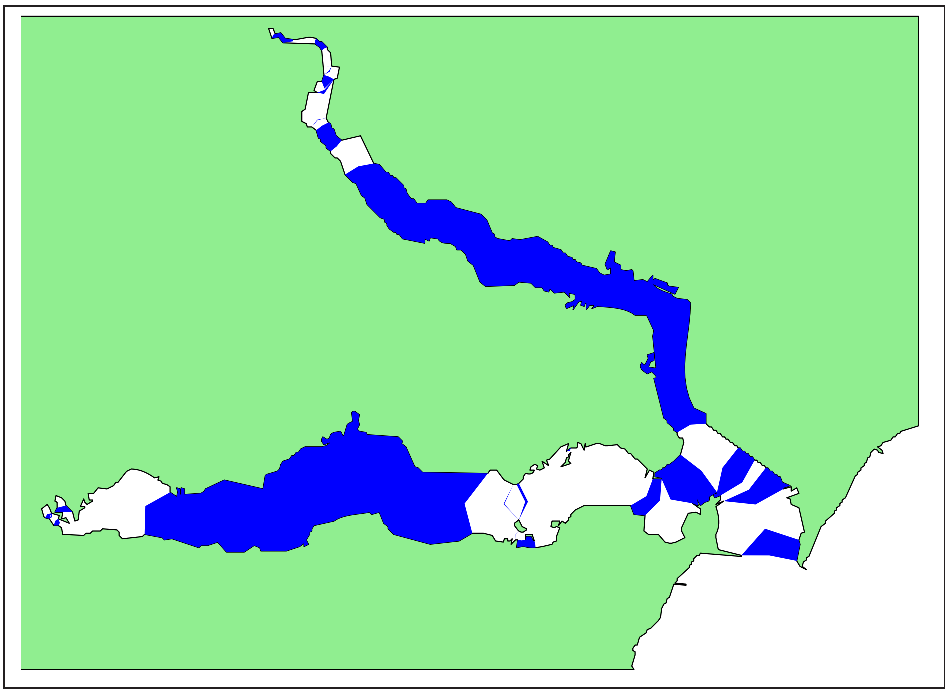


Figure C.3: The results of performing the linearity segmentation on the Stour-Orwell Estuary using a threshold of 1.4.

C.2 Stour-Orwell Estuary - Linearity With Respect to Edges

Figure C.4 shows the results of running the linearity with respect to the edges query at different thresholds. Segments that are both linear with respect to centre and edges are marked in blue.

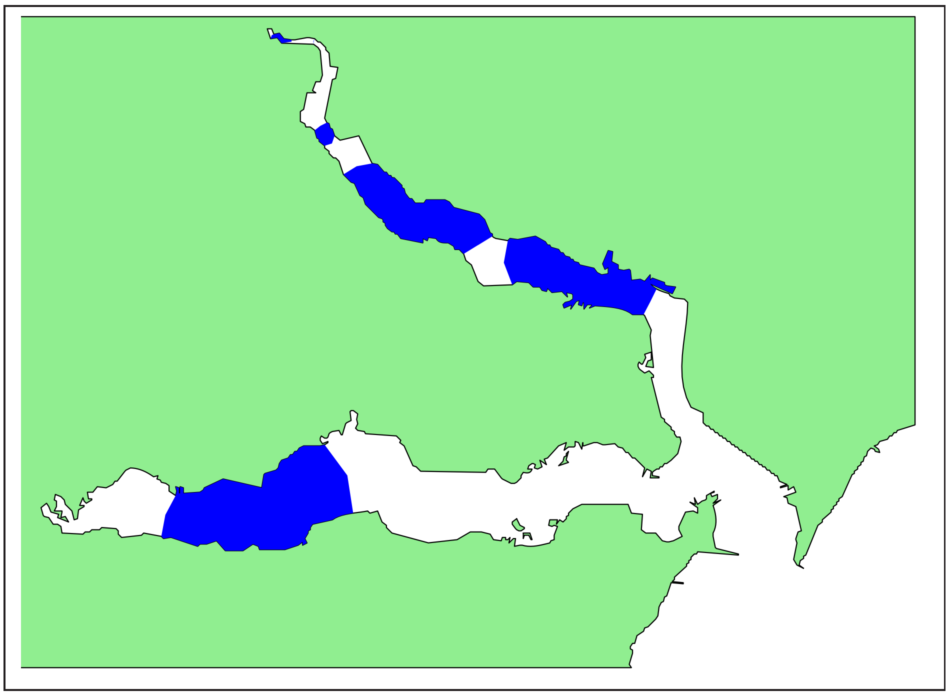


Figure C.4: The results of performing the linearity segmentation with respect to edges on the Stour-Orwell Estuary using thresholds of 1.2 - 1.4 (the same results were generated at all thresholds.

C.3 Stour-Orwell Estuary - Interstretches

Figures C.5 - C.6 show the results of running the interstretch query at different thresholds. Linear segments are marked in blue, interstretches are marked in red.

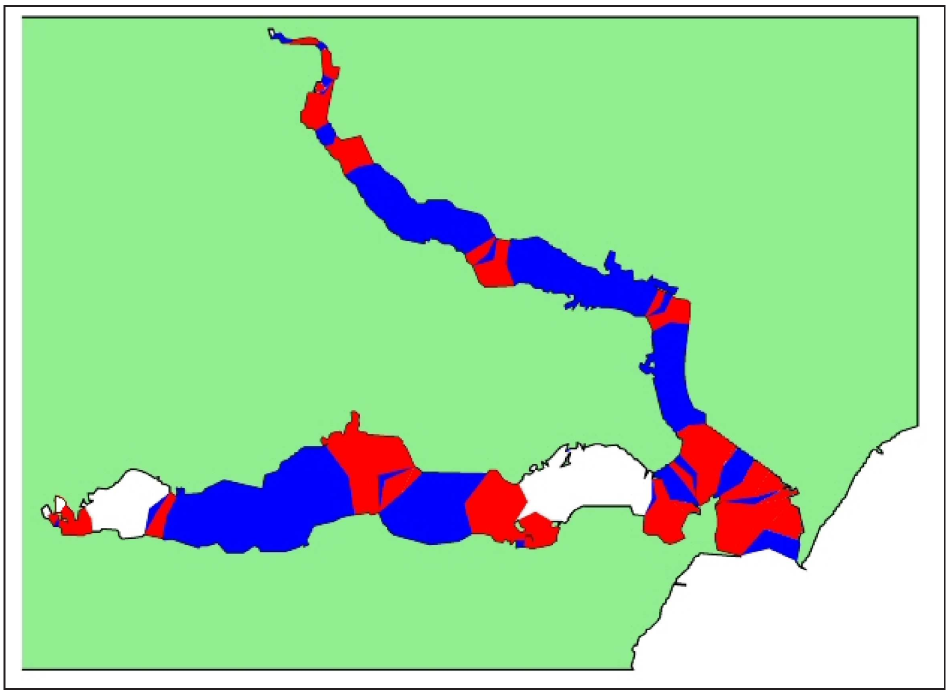


Figure C.5: The results of marking interstretches on the Stour-Orwell Estuary using a threshold of 1.0.

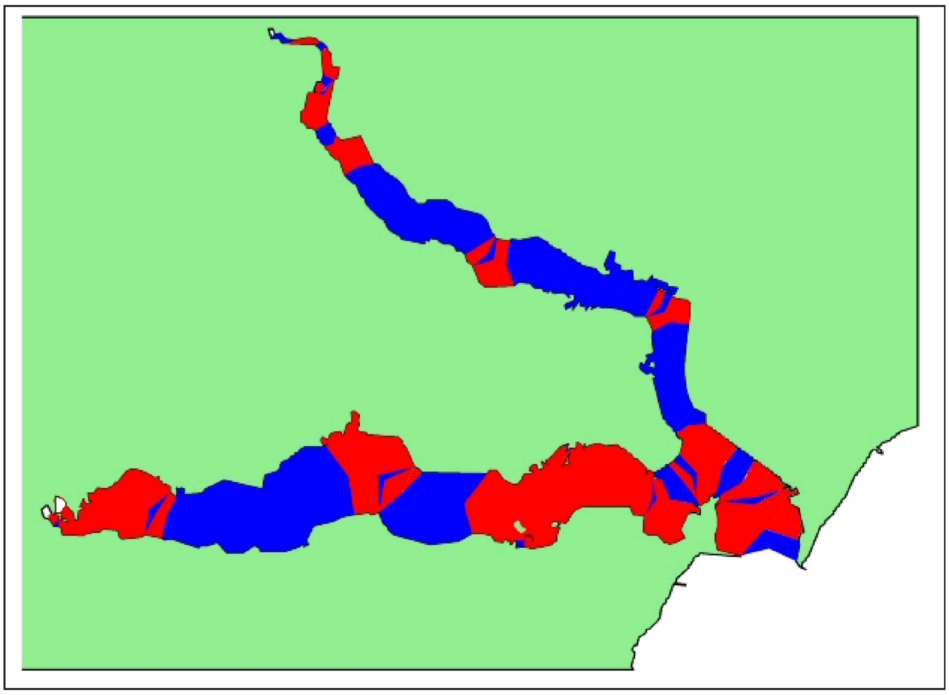


Figure C.6: The results of marking interstretches on the Stour-Orwell Estuary using thresholds of 3.0 and 5.0 (the same results are produced in each case).

C.4 Stour-Orwell Estuary - Island and Island-Water

Figures C.7 - C.9 show the results of running the island and island-water queries at different thresholds. Islands are marked in red, island-water regions are marked in blue.

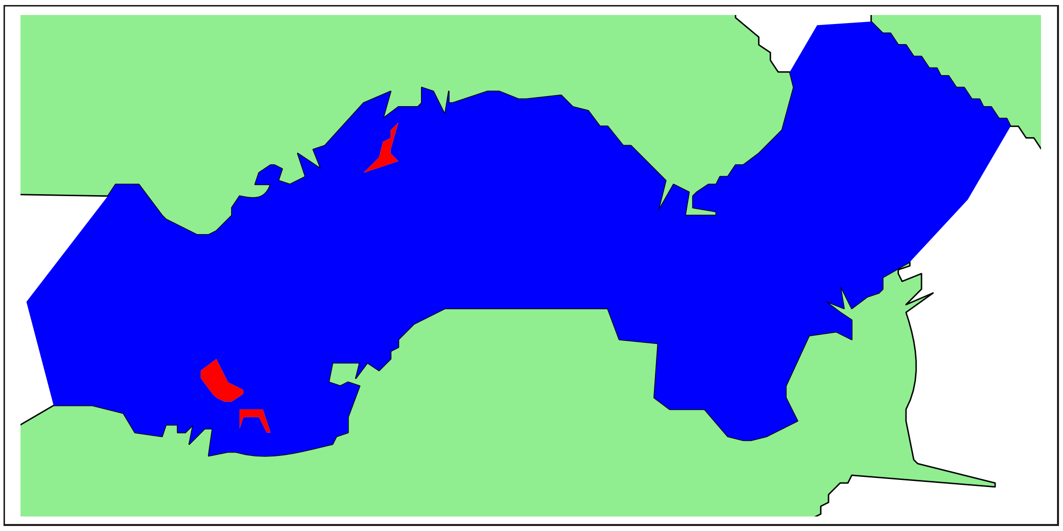


Figure C.7: The results of marking islands and island-water regions on the Stour-Orwell Estuary using a linearity threshold of 1.2.

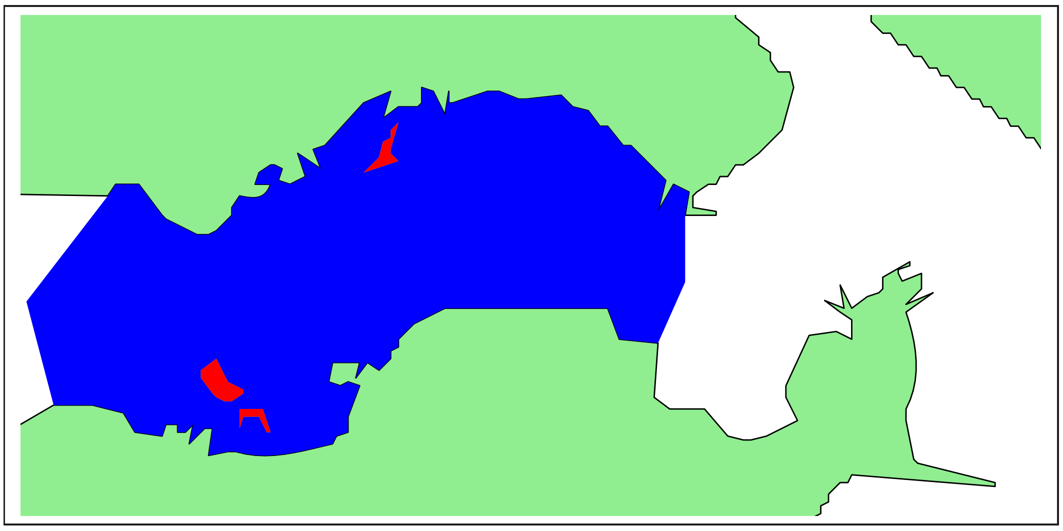


Figure C.8: The results of marking islands and island-water regions on the Stour-Orwell Estuary using a linearity threshold of 1.3.

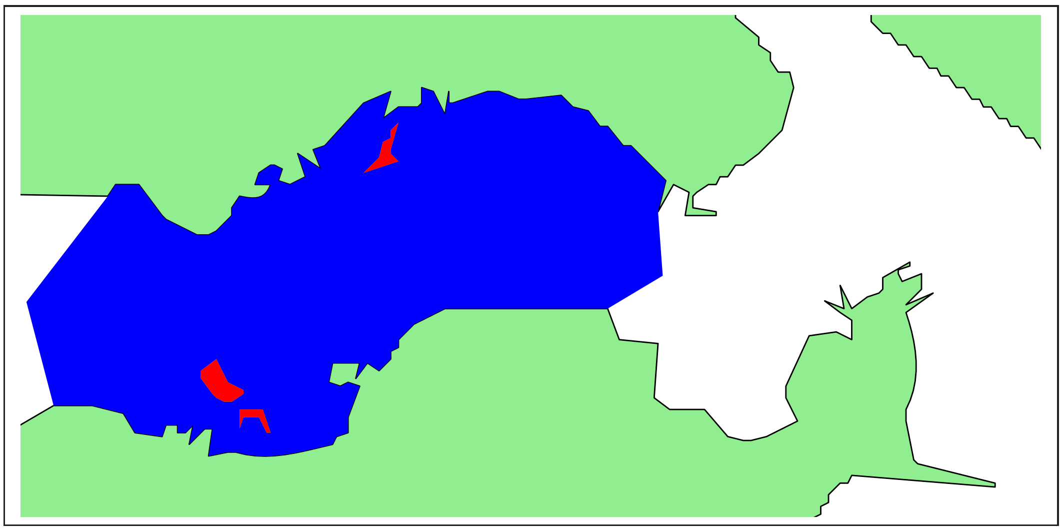


Figure C.9: The results of marking islands and island-water regions on the Stour-Orwell Estuary using a linearity threshold of 1.4.

C.5 Stour-Orwell Estuary - Confluence

Figures C.10 - C.11 show the results of running the confluence query at different thresholds. Confluence regions are marked in blue.

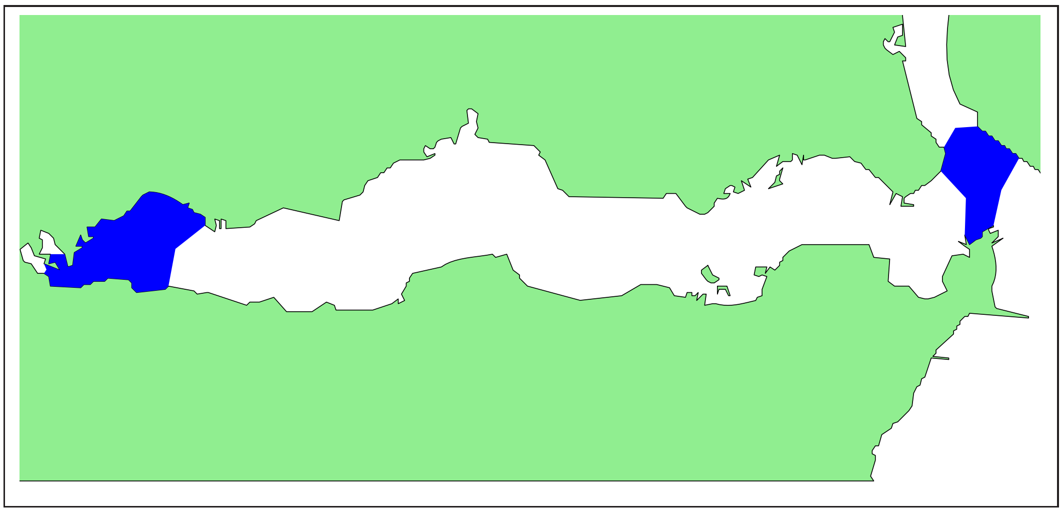


Figure C.10: The results of marking confluence regions on the Stour-Orwell Estuary using a linearity threshold of 1.3.

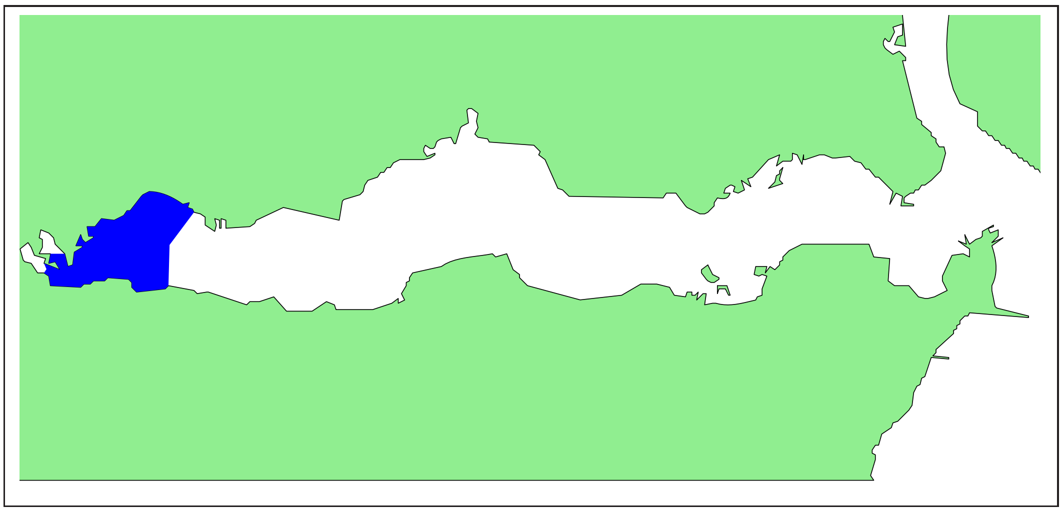


Figure C.11: The results of marking confluence regions on the Stour-Orwell Estuary using a linearity threshold of 1.4.

C.6 Stour-Orwell Estuary - Major Stretch

Figures C.12 - C.14 show the results of running the confluence query at different thresholds. Major stretches are marked in blue.

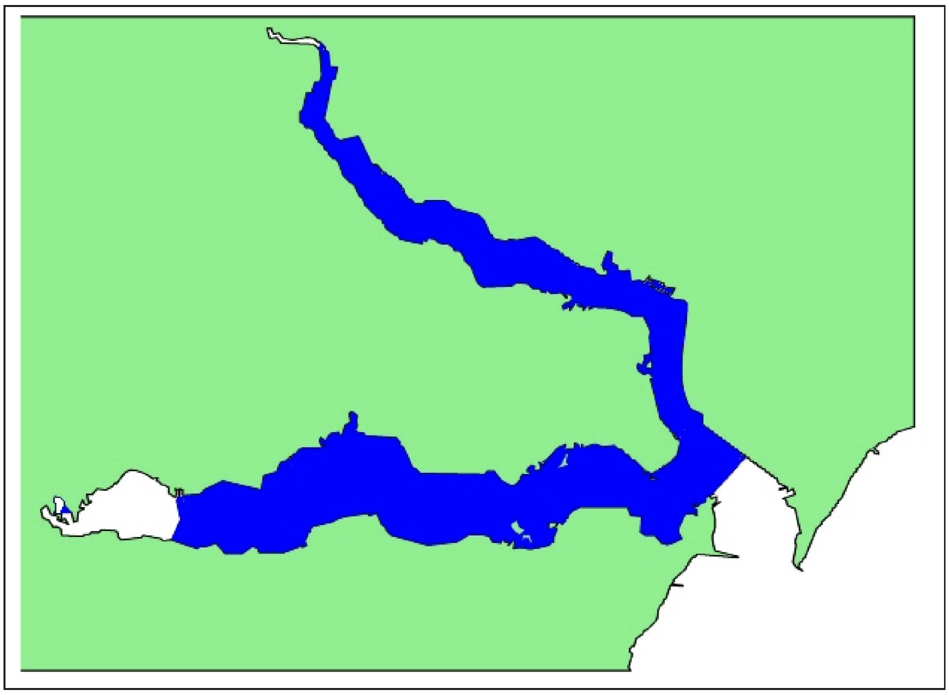


Figure C.12: The results of marking major stretches on the Stour-Orwell Estuary using a linearity threshold of 1.2.

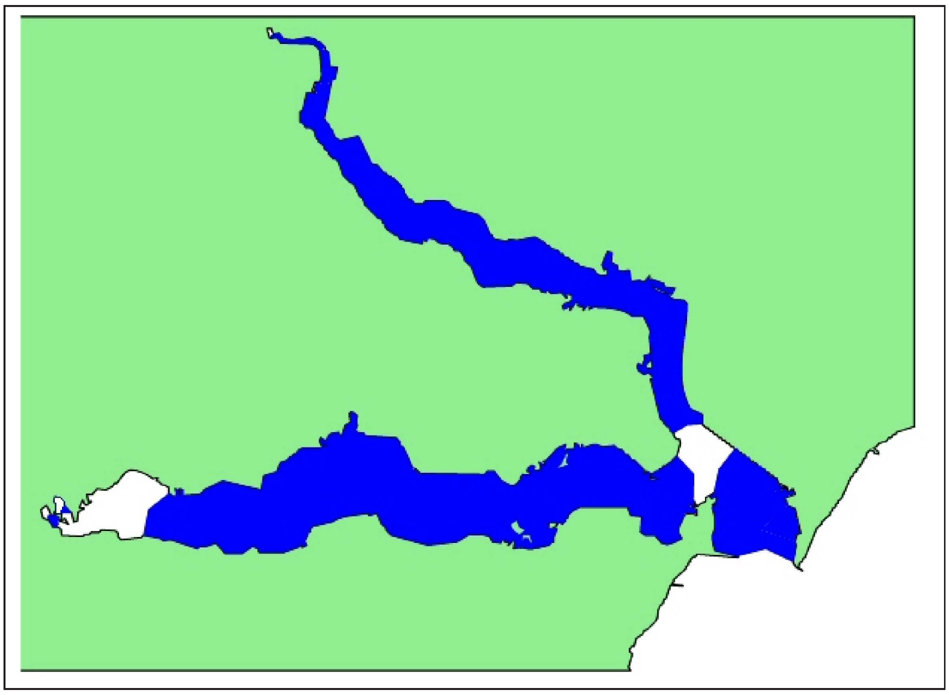


Figure C.13: The results of marking major stretches on the Stour-Orwell Estuary using a linearity threshold of 1.3.

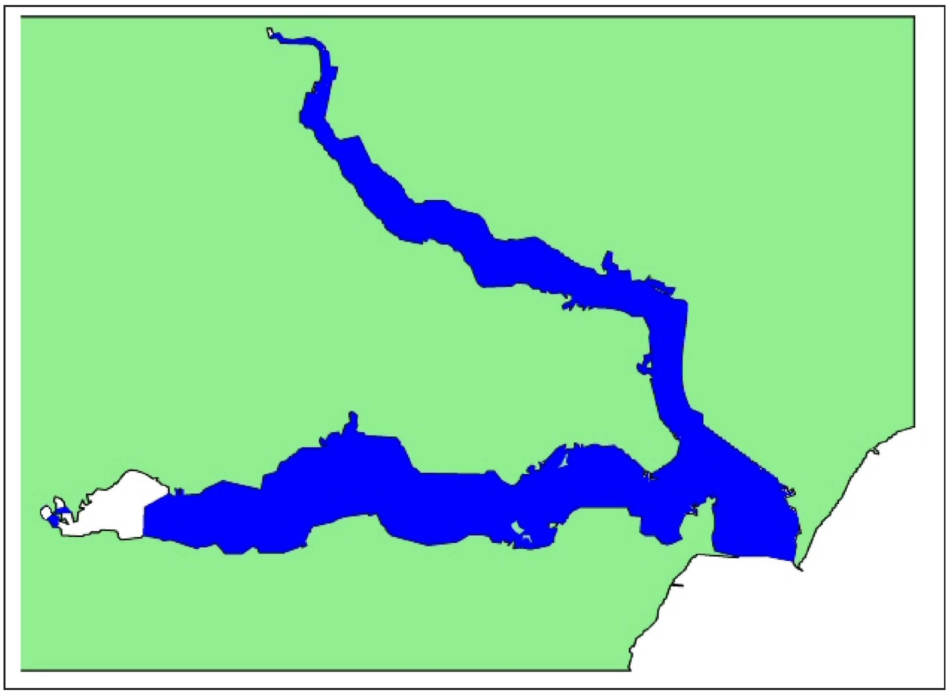


Figure C.14: The results of marking major stretches on the Stour-Orwell Estuary using a linearity threshold of 1.4.

Appendix D

Publications

The following publications, produced from the work in the thesis, are included.

Mallenby, D., “Grounding a Geographic Ontology on Geographic Data”, *8th International Symposium on Logical Formalizations of Commonssense Reasoning (Commonsense 2007)*, Stanford, USA (2007) 101–105.

Third, A. and Bennett, B. and Mallenby, D., “Architecture for a Grounded Ontology of Geographic Information”, *Second International Conference on GeoSpatial Semantics (GeoS 2007)*, Mexico City, Mexico, (2007) 36 – 50.

Mallenby, D. and Bennett, B., “Applying spatial reasoning to topographical data with a grounded geographical ontology”, *Second International Conference on GeoSpatial Semantics (GeoS 2007)*, Mexico City, Mexico, (2007) 210 – 227.

Bennett, B. and Mallenby, D. and Third, A., “An Ontology for Grounding Vague Geographic Terms”, *Fifth International Conference on Formal Ontology in Information Systems(FOIS 2008)*, Saarbrücken, Germany (2008) to appear.

Grounding a Geographic Ontology on Geographic Data

David Mallenby

School of Computing
University of Leeds
davidm@comp.leeds.ac.uk

Abstract

Vagueness is prevalent within the geographical domain, yet it is handled poorly in existing ontology approaches. A proposed way to rectify this is to ground the ontology upon the data. By grounding the ontology, we make an explicit link between the ontology and the data, and thus allow reasoning to be made within the context of the particular data. In order to ground the ontology upon the data, we must first decide how to represent the data and how to handle the vagueness with reasoning. This paper illustrates the stages required to prepare geographical data for an ontology to be grounded upon, including considering how to reason about the vagueness, how to represent the data in a more efficient manner and how to reason about relations within the data to extract attributes that would be used within an ontology.

Introduction

There is a huge amount of geographical data available today, in a variety of formats from classical cartographic maps to satellite imagery. This data can be analysed, combined and reasoned with in Geographic Information Systems (GIS). In order to reason about geographical features we need a method of representing the data and the meanings attached in a logical manner. The use of ontologies has become a popular method of representing such data [9, 33, 11].

The use of ontologies in GIS has been proposed in [9, 27] amongst others. Existing methodologies do not adequately handle vagueness, which is inherent to the geographical domain. Features are often dependant on the context in which they are made, with local knowledge affecting definitions. Geographical objects are often not a clearly demarcated entity but part of another object [9, 27]. The individuation of entities is therefore more important to geographical domains than to others.

One approach proposed to improve the handling of vagueness is to ground the ontology upon the data [17]. By grounding the ontology, we make an explicit link between the ontology and the data, thus allowing reasoning to be made within the context of the particular data. Grounding the ontology upon the data requires the data to be represented in a manner that will allow the link between data and ontology. We require an approach that allows the ontology to segment the data accordingly, based on user specifications.

In this paper we will examine the stages that are required in order to convert geographical data into a suitable form upon which terms in the ontology can be grounded. The data to be looked at is of The Hull Estuary, with the aim being to obtain a method of reasoning about the hydrological features which are implicit in the data. It is important to note that the particular formats and segmentation processes applied here may not necessarily apply to other features within the geographical domain. Rather, the aim is to show the process of preparing such data for an ontology.

Motivation

One of the key considerations for geographical ontologies is the handling of vagueness [31]. Vagueness is inherent to the geographical domain, with many features defined without precise definitions and boundaries. Such definitions are dependant on the context in which they are made.

Vagueness is handled inadequately in present GIS; some approaches such as [9, 27] choose to ignore the size quantifier and categorise a river simply as a waterbody, whilst others have sets of quantifiers [31]. Both approaches base the size quantifier on a predefined perspective that may not be agreed upon or may be based on a particular context that isn't applicable in all situations.

Vagueness is not a defect of our language but rather a useful and integral part. Rather than attempting to remove vagueness, it is better to develop an approach that allows the user to decide what makes up a vague feature. By improving the handling of vagueness, we improve the functionality of GIS, allowing vague features to be reasoned about in an effective manner.

Vagueness in Geography

As discussed by Bennett [2], vagueness is ubiquitous in geographical concepts. Both the boundaries and definitions of geographical concepts are usually vague, as well as resistant to attempts to give more precise definitions. For example, the definition of a river as given by the Oxford English Dictionary [1] is:

> A large natural flow of water travelling along a channel to the sea, a lake, or another river.

This is clearly vague, with the most obvious example being the use of 'large', although there are other parts of the definition that are vague also.

The sorites paradox can be easily adapted to illustrate vagueness in geography, as shown in [32, 33]. So, whilst there are some things that are definitely rivers and some that are definitely not, there does not exist an explicit boundary between the two sets, thus classical reasoning can not state if something is or isn't a river.

Geographical definitions are dependant on the context in which they are made. For example, in the UK rivers are defined usually as permanent flows, but in Australia they may not contain water all year round, thus there is a temporal requirement to the definition [29].

The principal approaches for handling vagueness at present are fuzzy logic and supervaluation theory. Both approaches offer a method of reasoning over vague features. It is usually the case that the two are presented as opposing theories. However, this in part assumes that vagueness can only take one form, which as discussed in Dubois [7] is not true. Rather, there are instances where it is more appropriate to use fuzzy logic and instances where supervaluation theory is better.

Fuzzy logic is the popular approach to handling vagueness, and has been used in a variety of applications since its conception by Lotfi Zadeh [37, 35, 36]. The underlying concept is to allow a method of processing data by allowing partial set membership rather than strict set membership or non-membership. Fuzzy logic is especially adept at handling situations where we do not want to generate an explicit boundary between two sets, but rather represent a gradual transition between the two.

Initially proposed by Fine [8], supervaluation theory proposes that there exist many interpretations of the language. Statements could therefore be true in some interpretations and false in others. In supervaluation semantics, 'precisifications' are used to determine the boundary points at which statements are considered true or false in a given interpretation. Supervaluation theory is suited to situations where we wish to generate a boundary between sets that we know exists but are not able to permanently mark as such.

In our proposed system, we wish to segment, individuate and label hydrological features. We therefore require a method of reasoning that marks explicit boundaries depending on user preferences.

If we were wishing to mark features with transitional boundaries, then fuzzy logic would be suitable, as we would have fuzzy boundaries between features. However, an attempt to return crisp boundaries would not be suited to fuzzy logic due to logical rules used in reasoning.

Supervaluation theory on the other hand, is suited to return a crisp boundary for given preferences. With fuzzy logic we take the stance that there is not a boundary between features so we show a gradual range, whereas with supervaluation theory we assume that there is a boundary, we just don't know for certain (or agree upon) where it is. The user preferences therefore become the precisifications. Supervaluation theory is therefore preferable for this problem.

Ontology Grounding

The ontology level is usually seen as separate to the data level; we reason within the ontology, and return the data that matches our queries. Thus the ontology is devoid of the data context, despite any impact this may have. This has a clear impact upon handling vagueness, where attributes are based heavily upon the context in which they are made.

A proposed improvement to this is to ground the ontology upon the data [17]. By grounding the ontology, we make an explicit link between the ontology and the data, thus allowing reasoning to be made within the context of the particular data.

The symbol grounding problem as proposed by Harnad [13] suggests that computers do not actually understand knowledge they are provided, as meanings are merely symbols we attach to objects. There have been no adequate solutions to this problem as yet and it remains an open problem [28]. Ontology grounding does not solve the problem. Rather, it allows the user to decide the meaning of concepts to some extent.

Grounding the ontology upon the data allows reasoning with the data in particular context. Thus in a particular context a river could be a channel that contains water for a particular period of time as opposed to a permanent flow.

To ground the ontology upon the data, we need to work at both the data level and the ontology level. At the ontology level, we need to consider what attributes we require in order to identify or reason about a feature, whilst at the data level we need to consider how we will obtain such attributes. For example, linearity is an important concept when analysing geographical domains, as the way a feature's shape changes is often used to classify that feature.

So by identifying linear stretches within data, we have an attribute that can be passed to a grounded ontology to facilitate reasoning about that feature. Because linearity is dependant on the data and the context it is used, we must ground the ontology upon the data to collect such an attribute.

Data representation

In order to ground the ontology upon the data, we need to represent the data in an appropriate manner. We need to consider what attributes we require and how these may be collected from the data provided. This is crucial to geographical objects, as often a feature is part of a larger feature, as opposed to being a unique object. Individuation is therefore more important in the geographical domain than in other domains.

The case study looked at here is for inland water networks. Previous work on an ontology for water networks was done in [3]. Here, formal concept analysis was used to determine the attributes required to reason about water networks. The key attributes included flow, size and linearity, with flow and linearity are closely linked. So, we require a method of extracting linear stretches that could be passed to an ontology. We start with our initial polygon that represents the water network, and need to analyse the geometry to determine linear stretches. Linearity is a vague concept, so we will use techniques based upon supervaluation to determine when exactly a particular part of a polygon is considered linear. Thus the user sets the precisification for linearity.

The initial polygon of the water network is insufficient to reason about aspects such as flow or linearity effectively, so we require a better representation of the polygon. The medial axis of a polygon as first proposed by Blum [4] is defined as the locus of the centre of all the maximal inscribed circles of the polygon. Here, a maximal inscribed circle is a circle that cannot be completely contained within any other inscribed circle in the polygon [10].

The benefits of using the medial axis in relation to river networks is discussed in [22], and was suggested in [3] as a way of determining the linearity of stretches of river. The medial axis (or skeleton) has also been used in similar problems to determining river junctions, such as road networks [16].

There are numerous methods for calculating the medial axis, such as extraction from the Voronoi diagrams [5, 14, 19], fast marching methods [30], the divergence of flux [6], and use of the Euclidean distance transform [10, 15, 23, 26].

A Voronoi diagram based approach offers a relatively simple and efficient method of obtaining the medial axis, as the medial axis is a sub graph of the Voronoi diagram for a simple polygon, and so we need only delete the unnecessary Voronoi edges. The VRONI approach and program developed by Held [14] produces Voronoi diagrams and associated derivations such as the medial axis.

The Voronoi/medial axis approach could also be suitable for other areas of the geographical domain. For example, the density of buildings within a village could be analysed using a voronoi diagram, whereby the size of the cells represents the density of buildings.

Figure 1 shows the result of calculating the medial axis of our input file of the Hull Estuary. Because we are only interested in inland water features, the medial axis of the sea was removed, leaving only the medial axis corresponding to the inland water network and a small extension beyond the river mouth.

Attribute collection

At an abstract level, the medial axis provides us with a useful and meaningful representation of the original shapes. For example, in Figure 1 the centre line of the river

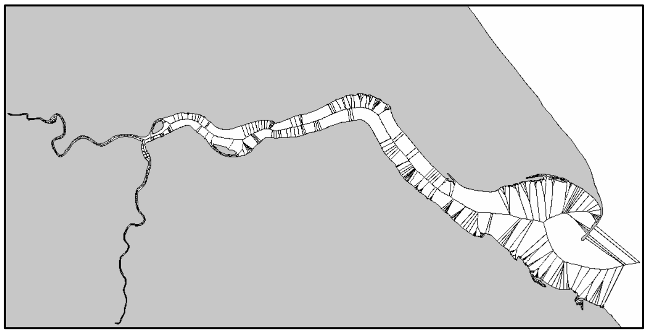


Figure 1: Medial Axis of The Hull Estuary

is easy to locate, and the number of lines in certain sections gives us an idea of the variation in shape in that area.

However, in order to extract any meaningful attributes to pass to an ontology, we must consider the relations between the data and determine the attributes to be extracted. The aim is to collect all the attributes required by an ontology grounded upon the data to reason about the features.

The medial axis is easily translated into a graph. The output from VRONI is a series of arcs, where the radius at one end of the arc is the smallest of the maximal discs on the arc, and the largest radius at the other end. The radii in between are therefore a transition between the two. By recording a point each side of the arc that these min-max radius touch the original polygon sides, we can construct a polygon from an arc or series of connected arcs. We can therefore translate the VRONI arcs into a graph, with the ends of the arcs being the nodes.

We also want to consider series' of arcs, by joining arcs considered to be part of the same channel. One method of determining what arcs to join together is to use approaches used to determine flow through the river network [12, 24, 25]. There are limitations to such approaches, as they assume that lakes and islands do not occur within the network, although Mark [20, 21] suggests that except in rare circumstances lakes do in fact have only one downstream flow. By applying the algorithm to our graph structure, we have an efficient method of determining what arcs to join together into 'superarcs'. We now have an effective method of representing the river network, and a basis from which to collect attributes.

Marking linear stretches

We could calculate whether a stretch is linear in a variety of ways. For our case study, we require the method to be scale invariant, as the size of the channels may vary dramatically.

To determine if a point is linear, we first find all the medial axis points that are on the same superarc within the maximal inscribed circle at that point. We then examine the radius at each of the points, determining the variance between minimum and maximum. If the variation of these widths is below some threshold, then the point is linear.

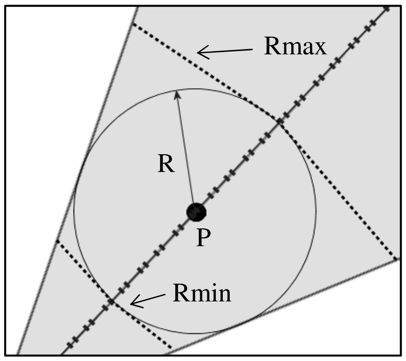


Figure 2: Example of linearity testing

Figure 2 demonstrates this process. Suppose we have the polygon as coloured grey. The medial axis at this section is a simple bisector of the two sides, represented by the dashed line that point P resides on. The maximal inscribed disc is the circle with radius R as shown. To determine if P is linear, we take all points on the medial axis that are within R distance of the point (all points of the medial axis contained by the maximal inscribed disc of P). We then find the radius of the maximal inscribed disc at each of the points, searching for the maximum and minimum values. In our example, these values will clearly be at the points a distance R from P. These are the dotted lines in the figure, marked Rmin and Rmax. If the variation between R-Rmin and R-Rmax is below a certain threshold, then we say the point is linear, as the width of the channel is only varying by a small amount.

So we now have a method of measuring the linearity in relation to the width. The approach is scale invariant, since larger rivers will require more points and smaller rivers will require fewer. This stage can therefore output sets of connected arcs within a superarc that are linear.

Marking gaps to be filled in

Depending on the precisification used, the previous stage may not find all the required stretches. Gaps may occur at sharp bends in a channel or sudden bulges. We could eliminate gaps by changing the degree of linearity required by the program, but in doing so we may end up classifying other sections as linear that we did not want to do so.

It is therefore intuitive to have 'gap' as an attribute that can be collected, whereby if a gap exists between two linear stretches and this gap is small enough (and thus been marked as 'gap'), we can join the stretches together into a major stretch. As with linearity, we require the measurement to be scale invariant.

We first search superarcs for any gaps between linear stretches. Given our graph structure, these are easily found, as each superarc represents a cycle-free path between the start and end nodes of that superarc. We can therefore simply traverse this path searching for gaps between linear stretches.

We calculate the length of the gap by adding the lengths of the arcs within the gap, and calculate the mid-point of the gap, obtaining the radius of the maximal inscribed disc at this point. If this value multiplied by a given threshold is greater than the length of the gap, then the gap is deemed sufficiently small and is marked with the 'gap' attribute. This approach is scale invariant, since larger gaps will require larger radius values at the mid-point in order to be marked as 'gaps'.

Result of marking major stretch

Arcs within the model are now labelled depending on the linearity and gap precisifications, and allow segmented polygons to be generated. We can now classify three simple features; linear stretches, gaps between stretches, and finally major stretches. Here, major stretches are defined as the union of linear stretches and gaps between that are sufficiently small.

The reason these are separate is because principal reasoning of features is to occur at the ontological level. This stage is to collect the attributes that are to be reasoned about at a higher level. The definitions of these attributes is now grounded upon the data, as the attributes 'linear' and 'gap' are unclear unless they are defined within the context of the data. This further ensures that the ontology will be grounded upon the data.

Figure 3 shows the results of these stages, having taken The Hull Estuary as input, with major stretches marked grey and the original medial axis shown in black. The system was developed in Prolog.

Despite only using two attributes, the system is able to mark major stretches stretching along the channels, including around islands. However, there are additional interesting results. First, the polygon generated as major stretches does not always go fully to the edge of the polygon, with occasional inlets missed out.

An example of this is shown in Figure 4, where a small inlet is not part of the major stretch. In some cases, we may want such an inlet to be part of the stretch, but there exist other cases where we would want that to be a separate feature; for example the inlet may in fact stretch out a long

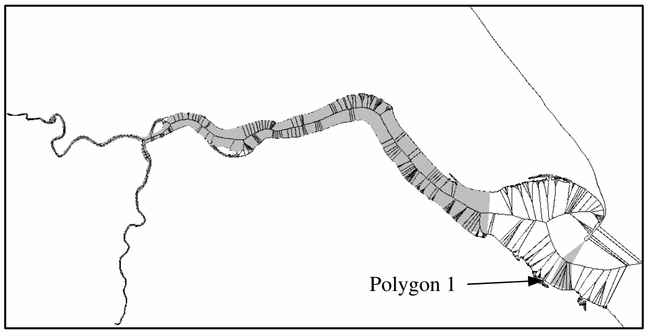


Figure 3: The result of marking major stretches, marked grey, with the original medial axis shown again in black. Polygon 1 represents a surprising result

way or be another channel. Therefore the most appropriate way to deal with this is in a similar fashion to gaps as previously discussed, and design an attribute for such inlets.

The other interesting result is the polygon occurring at the river mouth at Spurn Head, labelled polygon 1. At a first glance this does not seem to be a linear polygon. However, if we imagine travelling in a boat and attempting to remain roughly equidistant from both sides, we would find ourselves travelling in an arc that kept us equidistant from Spurn Head and the south bank of the river, as Spurn Head would be the closest point to the north of us.

To rectify this, we require a reconsideration of our definition of linearity. This particular result suggests that in order for a shape to be linear, we require both the variation in the width to remain small and also the variation in the curvature of the sides.

Future work

The present attributes used would only allow two different features to be considered; linear and non-linear stretches. However, there are many other attributes to be considered (including size, islands or temporal attributes), which in turn will allow us to reason about other features.

A more important stage is to feed the results in an ontology, so reasoning over the features can occur. This grounded ontology will be able to handle the vague entities contained within depending on the user's preferences. The ontology could be built in existing ontology languages such as OWL, as OWL can be inputted into Prolog for the reasoning stage [34, 18].

Conclusion

In this paper we have shown how geographical data can be represented and attributes collected to allow the grounding of an ontology. We have compared fuzzy logic and supervaluation theory, showing why they are suited to different tasks and why supervaluation theory is best suited to our particular problem.

We have also shown how the representation of the data is an important consideration, and that we must find the most effective method of representing the data.

Finally, we used the new representation to collect simple attributes that could then be passed to an ontology to reason about the features. In doing so, we have shown that adding these stages to the design process will allow a manner of reasoning about vague geographical features.

Acknowledgements

I am grateful for comments from Brandon Bennett and Allan Third who greatly helped with the presentation of the paper and my work. I am also grateful to our industrial partner Ordnance Survey, for their contribution towards PhD funding. Finally, I appreciated the comments of the reviewers that helped me prepare the final text.

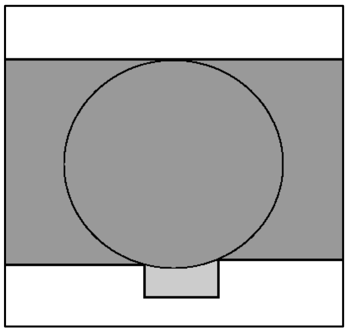


Figure 4 Simple example of how the stretch segmentation doesn't necessarily fill out the whole polygon. Dark grey represents a linear stretch, light grey a part marked as non-linear. The circle represents a maximal inscribed disc to illustrate why this occurs

References

[1] *Concise Oxford English Dictionary*, 2004.

[2] B. Bennett, *What is a forest? On the vagueness of certain geographic concepts*, Topoi-An International Review Of Philosophy, 20 (2001), pp. 189-201.

[3] B. Bennett, G. Sakellariou and P. Santos, *Supervaluation Semantics for an Inland Water Feature Ontology*, (2004).

[4] H. Blum, *Biological Shape And Visual Science.1*, Journal Of Theoretical Biology, 38 (1973), pp. 205-287.

[5] F. Chin, J. Snoeyink and C. A. Wang, *Finding the medial axis of a simple polygon in linear time*, Discrete & Computational Geometry, 21 (1999), pp. 405-420.

[6] P. Dimitrov, C. Phillips and K. Siddiqi, *Robust and efficient skeletal graphs*, *IEEE Conference On Computer Vision And Pattern Recognition, Proceedings, Vol I*, 2000, pp. 417-423.

[7] D. Dubois, F. Esteva, L. Godo and H. Prade, *An information-based discussion of vagueness*, *10th Ieee International Conference On Fuzzy Systems, Vols 1-3 - Meeting The Grand Challenge: Machines That Serve People*, 2001, pp. 781-784.

[8] K. Fine, *Vagueness, Truth and Logic*, Synthese, 30 (1975), pp. 265--300.

[9] F. Fonseca, M. J. Egenhofer, C. Agouris and C. Cmara, *Using Ontologies for Integrated Geographic Information Systems*, Transactions in Geographic Information Systems, 6 (2002).

[10] Y. R. Ge and J. M. Fitzpatrick, *Extraction of maximal inscribed disks from discrete Euclidean distance maps*, *1996 Ieee Computer Society*

Conference On Computer Vision And Pattern Recognition, Proceedings, 1996, pp. 556-561.

[11] N. Guarino, *Understanding, building and using ontologies*, International Journal Of Human-Computer Studies, 46 (1997), pp. 293-310.

[12] R. M. Haralick, S. Wang, L. G. Shapiro and J. B. Campbell, *Extraction Of Drainage Networks By Using The Consistent Labeling Technique*, Remote Sensing Of Environment, 18 (1985), pp. 163-175.

[13] S. Harnad, *The Symbol Grounding Problem*, Physica D, 42 (1990), pp. 335-346.

[14] M. Held, *VRONI: An engineering approach to the reliable and efficient computation of Voronoi diagrams of points and line segments*, Computational Geometry-Theory And Applications, 18 (2001), pp. 95-123.

[15] W. H. Hesselink, M. Visser and J. Roerdink, *Euclidean skeletons of 3D data sets in linear time by the integer medial axis transform*, *Mathematical Morphology: 40 Years On*, 2005, pp. 259-268.

[16] W. Itonaga, I. Matsuda, N. Yoneyama and S. Ito, *Automatic extraction of road networks from map images*, Electronics And Communications In Japan Part Ii-Electronics, 86 (2003), pp. 62-72.

[17] A. Jakulin and D. Mladenić, *Ontology Grounding*, *SiKDD 2005*, Ljubljana, Slovenia, 2005.

[18] L. Laera, V. Tamma, T. Bench-Capon and G. Semeraro, *SweetProlog: A system to integrate ontologies and rules*, *Rules And Rule Markup Languages For The Semantic Web, Proceedings*, Springer-Verlag Berlin, Berlin, 2004, pp. 188-193.

[19] D. T. Lee, *Medial Axis Transformation Of A Planar Shape*, Ieee Transactions On Pattern Analysis And Machine Intelligence, 4 (1982), pp. 363-369.

[20] D. M. Mark, *On The Composition Of Drainage Networks Containing Lakes - Statistical Distribution Of Lake In-Degrees*, Geographical Analysis, 15 (1983), pp. 97-106.

[21] D. M. Mark and M. F. Goodchild, *Topologic Model For Drainage Networks With Lakes*, Water Resources Research, 18 (1982), pp. 275-280.

[22] M. McAllister and J. Snoeyink, *Medial Axis Generalization of River Networks*, CaGIS, 27 (2000), pp. 129 -138.

[23] A. Meijster, J. Roerdink and W. H. Hesselink, *A general algorithm for computing distance transforms in linear time*, *Mathematical Morphology And Its Applications To Image And Signal Processing*, 2000, pp. 331-340.

[24] J. A. C. Paiva and M. J. Egenhofer, *Robust Inference of the Flow Direction in River Networks*, Algorithmica.

[25] J. A. C. Paiva, M. J. Egenhofer and A. U. Frank, *Spatial Reasoning about Flow Directions: Towards an Ontology for River Networks*, XVII International Congress for Photogrammetry and Remote Sensing, 24 (1992), pp. 318-224.

[26] E. Remy and E. Thiel, *Exact medial axis with euclidean distance*, Image And Vision Computing, 23 (2005), pp. 167-175.

[27] B. Smith and D. M. Mark, *Ontology and geographic kinds*, Proceedings, International Symposium on Spatial Data Handling (1998).

[28] M. Taddeo and L. Floridi, *Solving the symbol grounding problem: a critical review of fifteen years of research*, Journal Of Experimental & Theoretical Artificial Intelligence, 17 (2005), pp. 419-445.

[29] M. P. Taylor and R. Stokes, *When is a River not a River? Consideration of the legal definition of a river for geomorphologists practising in New South Wales, Australia*, Australian Geographer, 36 (2005), pp. 183-200.

[30] A. Telea, *An Augmented Fast Marching Method for Computing Skeletons and Centerlines*, in D. Ebert, P. Brunet and I. Navazo, eds., *EG/IEEE TCVG Symposium on Visualization*, ACM Press, 2002.

[31] E. Tomai and M. Kavouras, *From "onto-geoNoesis" to "onto-genesis": The design of geographic ontologies*, Geoinformatica, 8 (2004), pp. 285-302.

[32] A. C. Varzi, *Vagueness in Geography*, Philosophy & Geography, 4 (2001), pp. 49-65.

[33] A. C. Varzi, *Vagueness, Logic, and Ontology*, The Dialogue. Yearbooks for Philosophical Hermeneutics 1 (2001).

[34] J. Wielemaker, *An optimised Semantic Web query language implementation in prolog*, *Logic Programming, Proceedings*, Springer-Verlag Berlin, Berlin, 2005, pp. 128-142.

[35] L. A. Zadeh, *Fuzzy Algorithms*, Information And Control, 12 (1968), pp. 94-&.

[36] L. A. Zadeh, *Fuzzy Sets*, Information And Control, 8 (1965), pp. 338-&.

[37] L. A. Zadeh, *Information And Fuzzy Sets*, Proceedings Of The American Society For Information Science, 13 (1976), pp. 83-83.

Architecture for a Grounded Ontology of Geographic Information

Allan Third, Brandon Bennett, and David Mallenby[⋆]

School of Computing, University of Leeds, Leeds, LS2 9JT, UK
{thirda,brandon,davidm}@comp.leeds.ac.uk

Abstract. A major problem with encoding an ontology of geographic information in a formal language is how to cope with the issues of vagueness, ambiguity and multiple, possibly conflicting, perspectives on the same concepts. We present a means of structuring such an ontology which allows these issues to be handled in a controlled and principled manner, with reference to an example ontology of the domain of naive hydrography, and discuss some of the issues which arise when grounding such a theory in real data — that is to say, when relating qualitative geographic description to quantitative geographic data.

1 Introduction

A major problem with encoding an ontology of geographic information in a formal language is how to cope with the issues of vagueness, ambiguity and multiple, possibly conflicting, perspectives on the same concepts. We present a means of structuring such an ontology which allows these issues to be handled in a controlled and principled manner, with reference to an example ontology of the domain of naive hydrography, and discuss some of the issues which arise when grounding such a theory in real data — that is to say, when relating qualitative geographic description to quantitative geographic data.

We take an encoding of the "ontology" of a particular domain to be a collection of sentences in some formal language defining the terms of that domain and constraining their interpretation by means of axioms. We refer to such a collection as an *ontology* of that domain. One of the purposes of encoding an ontology is to assist the integration of heterogenous data sources and to enable the automatic handling of queries and reasoning tasks with regard to the natural high-level concepts associated with the domain in question. Such tasks may involve the relationships between the concepts themselves, or the application of those concepts to actual data gathered by domain experts.

In order to integrate different data sources, it is necessary to relate the terms defined in an ontology to data objects and their attributes. In terms of an ontology in a formal language such as first-order logic, a specific data set ideally provides a *model* for that ontology — that is to say, the formulae in the ontology

⋆ The authors gratefully acknowledge the support of EPSRC grant no. EP/D002834/1.

should all be *true* in the data set. The process of computing the relationship between terms and data — that is, providing concrete interpretations of predicates in terms of (sets of) data objects — we refer to as *grounding*.

However, as we noted above, geographic information is not straightforward. In particular, many natural geographic terms are *vague* (what is the difference between a hill and a mountain?) and ambiguous ("stream" can refer either to any channel containing flowing water, or to a small such channel such as a brook). The problem of ambiguity is exacerbated by the wide range of both the physical phenomena relevant to geography, and the variety of different human activities to which geographic information is relevant. A hydrographer may define a term such as "estuary" in terms of the relative salinity of different regions of water ([1]), whereas the cartographer, or the navigator of a boat, may each have quite different definitions. Such agents may disagree over which regions are considered "estuary", even if there exists a general commonly-understood meaning of the term to which all agree, and of which the particular meaning used by each is a specialisation. Similarly, two hydrographers, or the same hydrographer on different occasions, may vary in their interpretations of a single term, depending on the context. Thus different perspectives on reality can lead to ambiguity in the interpretation of common terms. In the context of information systems, different perspectives such as these can be reflected in the different kinds of information recorded in data sets: the hydrographer may collect data of no interest to the cartographer or navigator, and vice versa, and yet the same high-level geographic terms can be interpreted over the data gathered by each. Such ambiguity cannot be idealised away, nor, we believe, is it desirable to try to do so.

A further difficulty, which we believe is likely to apply to many situations in which abstract qualitative descriptions are related to quantitative data, is that humans tend to ignore "insignificant" deviations in reality from the abstract description. For example, small tree-less regions on the edge of a wooded area may nonetheless be included as part of a forest, and, as we discuss later, a river can still be classed as being vaguely linear overall, even if there are sections of it which are definitely non-linear, provided those sections are small enough. We show by example a way of handling such irregularities as part of the grounding process.

In light of these issues, we believe it is more useful to try to handle the ontology of geography in such a way as to accommodate vagueness, ambiguity and the existence of different perspectives, rather than attempt to anticipate and accommodate every possibilty. In this matter, we are in agreement with [2], who outlines a semantic framework incorporating an explicit notion of *context* which allows contextual variation for vague and ambiguous terms. The work we present here concerns the internal structure of an ontology, and its relation to data; we believe that any ontology of the kind we discuss could be slotted quite straightforwardly into the framework of [2].

We argue here for the use of a layered architecture for an ontology of geographic information which allows the vagueness and ambiguity of the general terms of that domain to be handled in a straightforward way. The structure we propose allows a principled approach to the problem of grounding the same ontology in different kinds of data. We illustrate this architecture by means of a simple ontology of inland water features, grounded in this case in two-dimensional "map" data.

2 The Semantics of Vagueness

A predicate p is said to be *vague* if there are elements of the relevant real-world domain which are neither clearly p nor clearly not p. The natural language term "river" is vague, for example, because there exist flowing water features about which it is unclear whether they are small rivers or large streams (or even elongated lakes). It is important to remember that the phenomenon of vagueness is distinct from that of ambiguity. A word can have more than one meaning, each of which is perfectly precise, and a word with a single meaning can have unclear boundaries of application. Many words, of course, exhibit both phenomena.

There are a variety of approaches in the philosophical and knowledge representation literature to the semantics of vague terms, from fuzzy logic [3], in which statements about borderline cases of vague terms are treated as partially true, to epistemic models [4], in which the lack of clarity about borderline cases is treated as a kind of ignorance, to supervaluation semantics [5]. In [6] and [7], it is argued that many vague geographic terms are such that, given a partial denotation of a term — for example, the set of clear-cut cases of river — there remain many "acceptable" ways of making that term precise. That is to say, one interpretation may include certain borderline cases of river as genuine rivers, and another may not, without either interpretation contradicting our intuitive understanding of the term. This argument suggests, then, that vague geographic terms can be interpreted using *supervaluation semantics*.

According to the standard account of supervaluationism, vague terms are interpreted relative to a set of *admissible* interpretations, each of which is a classical interpretation of those terms. A single admissible interpretation corresponds to one way of making all vague terms precise. A sentence containing a vague term is *supertrue* (*superfalse*) if it is true (false) on all admissible interpretations, and is neither true nor false otherwise.

To illustrate this idea, consider Figure 1, in which a range of different sources have shaded the region each considers to have some particular (vague) property p. The property p might be, for example, the property of being an estuary. All of our different sources have agreed that region A does *not* lie within the p-region, and all agree that region B *does* lie within it. We can thus identify a *core* region, considered to be p by *every* source, and a *fringe* region, the largest region which *any* source considers to be p. In Figure 1, the core region is that shaded by source 1, which is a subregion of those shaded by the other sources, and the fringe region is that shaded by source 3 — the regions shaded by both other sources are subregions of it. Core and fringe regions can be identified even if some sources provide fuzzy boundaries, by considering which regions are definitely (non-fuzzily) p and definitely not p.

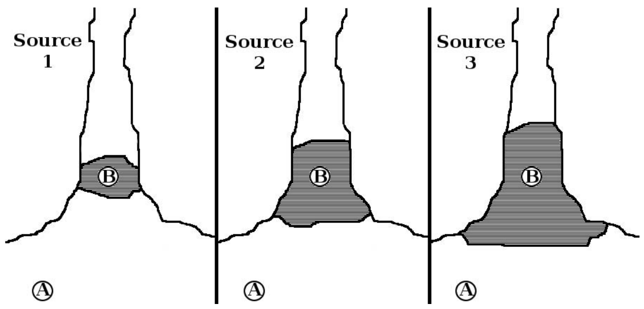


Fig. 1. Alternative interpretations of p

In terms of supervaluation semantics, the different sources correspond to different ways of making the term p precise, and the core and fringe regions identify the range of admissible interpretations. Every admissible interpretation must include the core region as p, and no admissible interpretation includes any region outside the fringe as being p. Supertrue sentences — those which are true in *all* admissible interpretations — turn out to be those to which every source would agree, and superfalse sentences those to which no source would agree. It is possible, of course, for some sentences to be true in some interpretations and not in others: these are the sentences to which some sources would agree, and some would disagree. Supervaluation semantics, then, models the situation where multiple agents, who can each have their own internally-consistent theory governing the use of a term nonetheless share a common understanding of it.

So a supervaluationist semantics of, say, vague hydrographic terms would contain both admissible interpretations in which a borderline stream/river would be classified as a stream, and other interpretations in which the same object would be classified as a river. A clear case of a river — the Amazon, say — would be interpreted as a river on every admissible interpretation, and so sentences referring to it as a river would be supertrue.

We believe that a supervaluationist approach is more appropriate than the use of a multi-valued or fuzzy logic for a variety of reasons. It is not clear, given a set of terms representing data objects, and a vague predicate, how exactly to assign truth values to formulae involving them. The notion of entailment in fuzzy logic is also not as strong as classical or supervaluationist entailment, and can weaken logical relationships between concepts.

In much of the philosophical literature on supervaluationism, the idea of the "admissibility" of an interpretation is often left worryingly undefined. We analyse admissibility, by means of the following observation. The applicability of the vague terms in which we are interested turns out to be dependent on certain precise properties which can take a range of values. For example, it seems clear that a river should be wider than a stream, even if it is unclear at precisely which specific value of width the boundary between them lies. We can model this phenomenon in terms of *threshold values* on the relevant properties — for example, a threshold on the average width of a channel of flowing water determining the boundary between river and stream. A specification of values for each threshold corresponds to a classical interpretation of the vague terms, making each precise. Specifying that the values of a threshold must lie in a given *range* therefore fixes a *set* of ways in which the relevant terms can be made precise. By building such thresholds into the formal definitions of vague terms, we acquire a straightforward means of controlling the set of admissible interpretations of those definitions, and, when grounding such definitions in real data, we can experiment with appropriate ranges of values for those thresholds, and quantify over them to be able to carry out reasoning and draw supertrue conclusions. In this paper, we leave the choice of constraints on threshold values open; however, we show that we can still give a semantically rich logical representation which works modulo the setting of thresholds. Detailed discussion of this particular approach to the logic of vagueness can be found in [8].

Throughout this paper, we represent an n-ary vague predicate p whose interpretation depends on m thresholds $t_1, \ldots, t_m$ by the notation

$$\mathsf{p}[t_1, \ldots, t_m](x_1, \ldots, x_n)$$

Some vague predicates, such as "small", for example, depend for their interpretation on a *comparison class*: what counts as small for a small man, say, is different to what counts as small for a small mouse. We indicate that c is the relevant comparison class for a predicate p (always unary in this paper) with the notation

$$\mathsf{p{:}c}[t_1, \ldots, t_m](x)$$

Any such p thus in fact represents a *family* of predicates, not dependent on a comparison class, one for each c.

3 Ontological Architecture

We divide an ontology into three separate layers, or modules: the *general*, *grounding* and *data* layers.

The general layer is a high-level theory of the structure of the domain, defining symbols corresponding to natural language terms. Where these terms are vague, we model the vagueness by means of parameters, in accordance with the preceding discussion. So, for example, in an ontology of geographic information, this layer defines basic notions such as types and classes of matter – water, oxygen, solid, fluid, and so on – basic spatial predicates, such as the languages of the Region Connection Calculus [9] or Region-Based Geometry [10] and temporal structure [11]. Such basic notions can then be used to define and axiomatise the high-level terms of the domain, such as planet, latitude, longitude, two-dimensional projections, and so on. It can also define the general, commonly-understood meanings of vague or ambiguous terms such as "river" and "lake".

The grounding layer takes predicates which are treated as primitive in the general layer, and provides definitions for those predicates in terms of precise predicates of the kind found in collections of data. For example, a grounding layer for a geographic ontology may take the high-level, vague definition of a river as, say, a large narrow stretch of water, and flesh out the idea of *stretch* with reference to the "linear" features of the two-dimensional geometry of a water network viewed from above. Different grounding layers can be given for the same general layer, depending on the kind of data one has in mind. Clearly, the detailed definition of a stretch of water in terms of two-dimensional data will not be sufficient to ground the definition of a river in a set of data incorporating three-dimensional topographic and bathymetric information. Similarly, different grounding layers can be used to accommodate different perspectives on terms in the general layer — for example, to enable the grounding of the same high-level concept — that of river estuary, say — in completely different data, relating to geometry and salinity, respectively, for example.

We believe that varying the grounding layer in this way can provide the infrastructure for dealing with some of the ambiguity of natural language terms mentioned in the general layer. Different senses of a given term can be encoded as different ways of grounding that term in reality, while those aspects of meaning which are common to all senses of a term can be encoded in the general layer. A full discussion of issues relating to ambiguity, vagueness and multiple perspectives on meaning can be found in [12].

The particular choice of grounding layer depends very heavily, therefore, on the data in which one wishes to ground the ontology, and is not constrained, as the general layer is, to contain commonly-understood terms. Rather, it provides the means of relating such common terms to the specifics of a particular perspective or set of data. It is thus a good place also to define technical terms which are not necessarily widely shared.

Finally, the data layer provides a concrete ground interpretation of the relevant grounding layer, and, by extension, an interpretation of the general layer. From a data set of, say, two-dimensional spatial regions with attributes such as "water", "land", and so on, it is possible to extract a set of ground atomic formulae in which each region in the data is represented by a constant and each attribute as a predicate. Such a set of formulae containing only predicates which are considered primitive in the grounding and general layers can represent a model of those higher-level layers based on actual data.

It is not necessarily straightforward to map the predicates of a high-level theory onto the attributes and relations found in data-sets such as Geographic Information Systems (GISs). Consider, for example, a high-level concept such as river, and suppose that we stipulate in its definition that a river should be vaguely geometrically linear. Suppose further that the actual data in which we want to ground our theory consists, not unusually, of a set of spatial regions and a flag stating whether each represents an actual region of water or land. The problem remains of identifying which subsets of these data can be identified as linear or not, subject to a vague parametrisation of linear. This problem is distinct from the issues of giving both context-independent, and specific context-dependent, definitions of high-level terms, and depends very much on specific data. This dependence is an advantage: in a discussion of rivers, say, the interpretation of terms such as long is very heavily context-dependent. We thus locate such segmentation in the data layer, which therefore consists of a set of data which has been analysed and marked up with the denotations of derived, but low-level, predicates such as linear, long or deep. Hence, the data layer is the most specific yet.

To summarise, then, in order to handle issues such as vagueness, ambiguity, and the grounding problem, we divide an ontology of a particular domain, such as geographic information, into three layers: the general layer, consisting of context-independent definitions of high-level predicates and including, for example, a general description of the structure of the planet, among other prerequisites for any geographical discussion; the data layer, corresponding to a specific data-set and consisting of a set of individual objects and the denotations of a range of "basic" predicates over that domain, often of an observational, quantitative nature, such as land, water, and so on, and more complex, but still low-level predicates which can be derived from the data, such as linear. Between these layers, we have what we call the grounding layer, which varies with context and relates the high-level terms of the general layer to the low-level terms of the data layer.

Figure 2 illustrates this structure, showing how a sample general layer for an ontology of geography can be related to two different grounding layers, one intended to ground high-level general predicates to two-dimensional topographic data, and one intended to ground those same predicates in three-dimensional topographic and bathymetric data. The two-dimensional grounding layer is then related to two different data layers, which share a definition of linear, but have different definitions of the highly context-sensitive term small. The formulae in Figure 2 are intended solely to be illustrative: clearly, a genuine attempt at an appropriate ontology requires much more detail. Note, however, how the threshold parameters for vague predicates are passed down through the layers.

This division into three layers is, we claim, a natural one. As we noted above, the applicability of certain high-level concepts may depend on the context or perspective in which they are interpreted, and it may be possible, or common, to interpret the same concepts as applying to different kinds of data. The separation between the general and the grounding layers is thus motivated. The role of the data layer we take to be more evident still: there is no general interest in a theory of any domain which applies only to one specific set of data, or to no data at all.

4 Example Ontology: Naive Hydrography

In order to illustrate the architecture proposed above, we present an example ontology of common water features, and ground its general layer in two-dimensional, map-like data. We define terms such as "river" and "lake" in a way which we believe to represent a formal encoding of the intuitions of the average native speaker of English confronted with an unlabelled map; no doubt a trained hydrographer would take issue with some or all of our definitions, but,

General: spatial and temporal logic global structure high-level predicates $\mathsf{river}[l](x)$,$\mathsf{stream}[s,l](x)$		
Grounding 1: $\mathsf{river}[l](x) \leftrightarrow \mathsf{linear}[l](x) \wedge \mathsf{water}(x)$ $\neg\mathsf{small}[s](x)$ $\mathsf{stream}[s,l](x) \leftrightarrow \mathsf{linear}[l](x) \wedge \mathsf{water}(x)$ $\mathsf{small}[s](x)$		**Grounding 2:** $\mathsf{river}[l](x) \leftrightarrow \mathsf{2d\text{-}linear}[l](x) \wedge$ $\exists y(\mathsf{3d\text{-}bed}(x,y) \wedge$ $\mathsf{3d\text{-}channel}(y))$
Data 1: 2-d topographic data Human-scale $\mathsf{small}[s](x)$	**Data 2:** 2-d topographic data Boat-scale $\mathsf{small}[s](x)$	**Data 3:** 3-d topographic/ bathymetric data $\mathsf{2d\text{-}linear}[l](x)$ $\mathsf{3d\text{-}bed}(x,y)$ $\mathsf{3d\text{-}channel}(y)$
$\mathsf{linear}[l](x)$		

Fig. 2. Example of a layered structure relating the same general layer to multiple grounding and data layers

after all, the main aim of our proposed structure is precisely to accommodate such disagreement.

4.1 General Layer

We work in a first-order language with equality interpreted over regions of space, and assume an axiomatisation of a suitable set of spatial relations — for example, the binary relations of RCC-8 [9]. We assume henceforth that the reader is familiar with RCC-8.

We also assume a metric function d such that for any pair p_1, p_2 of points, $d(p_1, p_2)$ is the (shortest) distance between p_1 and p_2.

Although a fully general theory of geography must of course be able to represent time, for the moment we ignore issues regarding time, and possible changes in the nature of geographic entities over time, for simplicity. We anticipate that extension to include time in our theory can be carried out along the lines of the proposal in [13]. This paper is also the source of our interpretation of matter types. Briefly, mass nouns such as water are interpreted as referring to the sum of all spatial regions which contain only water. Thus, the interpretation WATER of the term water is itself a region, and so to express that a given region r contains only water, we simply need to write $\mathrm{P}(r, \textsc{water})$, where P is the RCC part relation. Terms referring to families of matter types (classes), such as solid, can be interpreted as the sum of the interpretations of all matter types in that class, and so again are interpreted as regions.

A full ontology of the geographic domain would also have to define terms relating to planetary structure, and the various ways of projecting from three dimensions onto two. Since everything henceforth is concerned with two-dimensional regions on the surface of the Earth, we assume, solely for reasons of space, that such an ontology can satisfactorily be constructed, and issues such as precisely what is meant by "surface of the Earth" can be dealt with.

We concentrate for the moment on the the distinction between river-like and lake-like water features. Obviously, there are many more issues to be considered in the hydrographic domain, such as the precise definition of sea. We intend to return to issues such as this in future work.

What, then, are the distinctions between river-like and lake-like regions? In the absence of data regarding water flow, temporal change, and so on, and with the intuition that many water features can be classified simply by considering shapes on a map, the obvious geometric condition is linearity. A river or stream has a course which approximates a line, albeit often one with a high degree of curvature, whereas a lake or a pond exhibits a non-linear, more disc-like, shape. Let us, then, take *linear-channel* and *expanse* to denote vaguely linear or vaguely circular regions of water, respectively, and l (for "linear") to be a parameter controlling the degree of deviation from true linearity allowed to a region before it is considered definitely not to be linear. The conditions of application of these predicates will be supplied by the grounding layer (as described below).

The more natural terms can then be defined in terms of *linear-channel* and *expanse*.

$$\mathsf{river}[l,n](x) \leftrightarrow \mathsf{linear\text{-}channel}[l](x) \wedge \neg\mathsf{narrow{:}linear\text{-}channel}[n](x)$$

$$\mathsf{stream}[l,n](x) \leftrightarrow \mathsf{linear\text{-}channel}[l](x) \wedge \mathsf{narrow{:}linear\text{-}channel}[n](x)$$

$$\mathsf{lake}[l,s](x) \leftrightarrow \mathsf{expanse}[l](x) \wedge \neg\mathsf{small{:}expanse}[s](x)$$

$$\mathsf{pond}[l,s](x) \leftrightarrow \mathsf{expanse}[l](x) \wedge \mathsf{small{:}expanse}[s](x)$$

where *narrow:linear-channel*$[n](x)$ and *small:expanse*$[s](x)$ are dependent on the width and size of x and the comparison classes of *linear-channel* and *expanse*, respectively, and parameters n, s (for "narrow", "small", respectively) modelling the vagueness of *narrow* and *small*.

Obviously, in general there are other issues relevant to the lake/river distinction such as speed of flow, geometry of the lake/river bed, and so on, but these involve temporal considerations and three-dimensional properties of space, which may not always be recorded in the data. Another temporal issue is that of how to characterise a "hydrographic feature" which does not always contain water, but, for example, only flows seasonally. These are issues to be dealt with in future work.

4.2 Grounding Layer

The preceding discussion outlines the general layer of our case study of an ontology of naive hydrography. We now continue by defining the predicates needed to ground the above definitions in actual data automatically.

In the context of the grounding layer ontology, we assume that the data available consist of a polygonal representation two-dimensional regions of water and land, as might be found in a GIS, and that analysis yields a segmentation of water regions into polygons classified as either linear or non-linear, according to some supplied threshold l of linearity. We also assume that each region in the segmentation is *maximal* with respect to linearity, by which we mean that no (linear/non-linear) region is a proper part of any other (linear/non-linear) region. We have implemented a geometric analysis algorithm [14], so our approach is already applicable to real-world data.

The purpose of the grounding layer is to relate the properties of the information in the data layer to the relevant primitive predicates of the general layer. The requirements of this layer have thus been informed by observations of the output at the data layer. The main issue which has been observed is that regardless of the threshold value for linearity, there are sections of regions of water representing real-world rivers which are not classified as linear. These sections seem to correspond to bends in the river course, junctions in which one river flows into another or irregularities in the shape of the river banks. Each of these phenomena can make a local section of water appear to be closer in shape to a circle than to a straight line, which leads to its classification as non-linear. Figure 3 illustrates this phenomenon, being a graphical representation of our sample data set, the river Humber on the east coast of the UK, and showing, by shading, which regions of that data set are classified as linear by our analysis tool. Unshaded areas in what intuitively one would consider to be linear stretches of water can be observed with any particular linearity threshold. However, casual observation of these "gaps" showed them generally to be insignificant in size compared to the surrounding linear regions. We believe this phenomenon arises from the fact that linear is an abstraction from the actual shape of a river or stream, as indeed any general geometric term applied to these features must be, and there are always likely to be irregularities such as these when describing natural, qualitative features in abstract terms. What is important, however, is to be able to deal with them in a principled way, which motivates the following definitions.

$$\begin{aligned}\mathsf{stretch}[l](x) \quad &\leftrightarrow \quad \mathrm{P}(x, \textsc{water}) \wedge \mathsf{linear}[l](x) \wedge \\ &\forall y(\mathrm{P}(y, \textsc{water}) \wedge \mathsf{linear}[l](y) \wedge \\ &\mathrm{P}(x, y) \rightarrow \mathrm{EQ}(x, y))\end{aligned}$$

$$\begin{aligned}\mathsf{interstretch}[c, l](x) \quad &\leftrightarrow \quad \neg\mathsf{linear}[l](x) \wedge \\ &\forall y(\mathrm{EC}(x, y) \rightarrow (\mathsf{land}(y) \vee \\ &(\mathsf{water}(y) \wedge \mathsf{linear}[l](y) \wedge \mathsf{close\text{-}to}[c](x, y))))\end{aligned}$$

where EQ is RCC equality of regions, and EC is the "external connection" relation. So a *stretch* is a maximal linear region of water, and an *interstretch* is a region of water externally connected only to land, or to regions of water which are *close-to* it, where c, for "close", parametrises the vague predicate *close-to*. Thus an interstretch is an "insignificant" region of water between stretches.

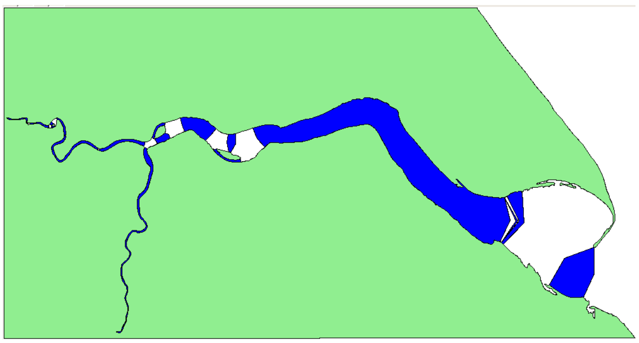


Fig. 3. Linear sections of the Humber estuary

The *linear-channel* predicate required by the definition of *river* and *stream* is then defined to apply to any region that is equal to the sum of a maximal sequence of regions $r_1, \ldots, r_n$ such that for each i, $1 < i < n$, r_i is either a stretch or an interstretch between r_{i-1} and r_{i+1}, and for $i = 1$ or $i = n$, r_i is either a stretch or a *stretch-source*, and

$$\begin{aligned}\mathsf{stretch\text{-}source}[l](x) \quad \leftrightarrow \quad & \mathrm{P}(x, \textsc{water}) \wedge \\ & \exists y(\mathsf{stretch}[l](y) \wedge \mathrm{EC}(x, y) \wedge \\ & \forall z(\mathrm{EC}(x, z) \wedge \neg \mathrm{P}(z, y) \rightarrow \\ & \neg \mathrm{P}(z, \textsc{water})))\end{aligned}$$

That is, a stretch source is a region of water externally connected to a stretch but otherwise entirely surrounded by non-water. Stretch-source is intended to capture the situation where a water channel appears to come out of the ground, at a spring, say. Such a region, it is easy to check, will always be classified as non-linear, but is not an interstretch, being connected to only one stretch.

An *expanse* plays a similar role for lake-like regions as stretch does for river-like regions, and is defined as follows.

$$\begin{aligned}\mathsf{expanse}[c, l](x) \leftrightarrow\ & \mathrm{P}(x, \textsc{water}) \wedge \neg \mathsf{linear}[l](x) \wedge \\ & (\neg \mathsf{interstretch}[c, l](x) \wedge \neg \mathsf{stretch\text{-}source}(x)) \wedge \\ & \forall y(\mathrm{P}(y, \textsc{water}) \wedge \neg \mathsf{linear}[l](y) \wedge \mathrm{P}(x, y) \rightarrow \mathrm{EQ}(x, y))\end{aligned}$$

The discussion so far has ignored the fact that linear channels can be, and often are, connected to one another, and that these connections occur in different kinds. Specifically, it is possible for two rivers, say, to merge to form a larger river (a "confluence"), for one smaller river to flow into a larger river (a "tributary"), and for a single river to divide into two separate channels, which may rejoin each other further downstream, as happens, for example, when an island occurs. We give an outline of a naive way of handling these different kinds of junction in our sample theory. We consider only the case of junctions between two channels, for simplicity. It is hoped that more complex junctions can be decomposed in terms of simpler cases.

Suppose, then, that we have two linear channels, c_1 and c_2, each of which is composed of a connected sequence of stretches, interstretches, stretch sources and stretch inlets according to the constraints given above. Let c_1 consist of the sequence $r_1, \ldots, r_n$ of such regions, and let c_2 consist of $s_1, \ldots, s_m$, such that r_i is connected to r_{i+1} and s_j is connected to s_{j+1} for all i, j, $1 \leq i < n, 1 \leq j < m$. There is a junction between c_1 and c_2 if $r_1, \ldots, r_n$ and $s_1, \ldots, s_n$ have either a common initial subsequence, a common final subsequence, or both (provided in this final case that both channels also contain distinct subsequences of regions, otherwise $c_1 = c_2$). That is to say, either, for some i, $1 \leq i < n, i < m$, $r_j = s_j$ for all $j \leq i$ and $r_{i+1} \neq s_{i+1}$, or, for some i, $1 < i \leq n, i \leq m$, $r_j = s_j$ for all j, $i \leq j \leq n, j \leq m$ and $r_{i-1} \neq s_{i-1}$, or both of these hold simultaneously. For ease of exposition, let us assume that c_1 and c_2 have a common *final* subsequence, representing the case where two linear channels merge to form a new channel. We refer to that common final sequence as c_3, and let c_1', c_2' be the initial sequences of c_1 and c_2, respectively, so that c_1 is the concatenation of c_1' and c_3 and c_2 is the concatenation of c_2' and c_3. We identify the junction of c_1 and c_2 with the triple (r, s, t) consisting of the final regions r, s from each of the sequences c_1', c_2', respectively, and the first region t in the sequence c_3.

The idea of junction is thus precise: given a segmentation of water regions into linear channels as described above, the interpretation of junction is fixed. One source of vagueness, however, lies in the notions of tributary and confluence. We assume that there are two possible ways to characterise the merging of two linear channels: either two similarly sized channels flow together to form a new, “large” channel, or one “small” channel flows into a “large” channel. We refer to these cases as confluence and tributary, respectively.

In order to interpret these terms in our theory, we need a vague notion of “similar size”. Let us say, then, than for any two members r_i, s_j of the sequences of regions making up linear channels c_1 and c_2, similar-size-to$[c](r_i, s_j)$ holds if the difference between the average widths w_i, w_j of r_i, s_j are less than c, using the same vagueness parameter c we used for close-to to represent a “small” distance in the relevant context.

We can now define confluence and tributary. We say that the junction (r, s, t) of c_1 and c_2 is a confluence if similar-size-to$[c](r, s)$; thus neither c_1' nor c_2' can be identified as the “main” channel into which the other is flowing. We say that c_2' (which, it is easy to check, will be a linear-channel) is a tributary of c_1 if similar-size-to$[c](r, t)$, and the average width of s is less than, and *not* close-to the widths of r, t. We believe that physical constraints rule out the possibility that none of r, s, t are a similar-size-to either of the others, and the possibility that the widths of any of r, s, t are not appropriately representative of the “widths” of c_1', c_2' and c_3'. Note that since confluence and tributary depend on similar-size-to, both of these predicates depend on the vagueness parameter c of close-to.

Naively, we interpret similar-size-to to refer to the physical size of the channels at the junction. A more sophisticated approach can always, of course, interpret these with respect to more hydrographically relevant considerations, such as size of catchment for each channel, supposing that such information is available in, or can be deduced from, the data.

The case where two channels diverge, and then rejoin further downstream, is more straightforward, and can be used, relatively easily, to identify regions of land which can be described as *islands*.

These definitions provide enough information to ground the high-level terms given earlier in actual data consisting of regions of ground and water classified as linear or otherwise.

4.3 Data Layer

The grounding layer we have outlined above is intended to relate predicates of the general layer to actual data in the form of two-dimensional regions of both water and ground. We assume that such data is relatively common, and that it is relatively straightforward to compute which spatial (RCC-8) relations hold between regions. What remains is to compute the denotations of the remaining predicates of the grounding layer. In the theory we have given above, those predicates are close-to, small:expanse, narrow:linear-channel and linear.

All of the predicates that we must interpret are parametrised in order to model their vagueness, and it is at the level of the data layer that values, or ranges of values, for those parameters must be set. The predicate close-to, which relates two regions, is relatively easy to deal with, by computing the largest distance between any two points in those regions, and stipulating that close-to holds when that distance is less than the value of the parameter c. Such a definition of close-to corresponds to intuition and reflects, through a suitable choice of value for c, the highly context-sensitive aspect of its meaning. The related terms small:expanse and narrow:linear-channel can be handled similarly, with the relevant threshold parameters being compared to, say, area and average width, respectively.

The more difficult spatial predicate to interpret is, as might be expected, linear, where by “linear region”, we mean, loosely speaking, a region whose width is small, and relatively constant, with respect to its length. A detailed discussion of how we compute linearity can be found in [14]. Roughly speaking, though, we wish to classify a region as *linear* if it does not exhibit “too much” variation in width along its length, with the notion of “too much” being controlled by the parameter l.

Figure 3 earlier shows which regions are considered linear by the algorithm of [14] in our sample set of input data, which, as stated above, represents the Humber estuary on the east coast of the UK. This classification depends on a particular choice of linearity threshold l. The shaded regions are linear. Note the presence of unshaded regions which lie within the area one might intuitively wish to classify as river. These are locally non-linear regions, which with the interpretation of close-to, can be classified in terms of the higher-level theory as either interstretch or stretch-source.

5 Conclusions

We have presented a layered architecture for ontology concerning vague, ambiguous and context-dependent terms, and illustrated this architecture with reference to a simple ontology of water features. We have discussed the grounding of this ontology in two-dimensional data. Our architecture is designed to assist the grounding of high-level definitions in actual data, without having to sacrifice the vagueness and ambiguity inherent to many geographic terms, for example.

We have implemented a system which is able to take appropriate sets of data, and an encoding of a high-level geographic ontology, and by means of suitable grounding definitions, carry out various tasks relating the high-level terms to the data, such as identifying to which data objects those terms apply, and evaluating complex formulae over those objects.

This system enables us to test different definitions and explore the effect on the resulting classification of the data; it was such testing which identified the need for the grounding term interstretch, the gap between the abstract description of rivers in two dimensions as vaguely linear water features, and the irregularities and "insignificant" deviations from this abstraction which occurs in actual data. This resulting accommodation is carried out in a principled manner which we believe reflects the approach humans take to the interpretation of such terms.

It should be noted that the specific ontology we have presented is by no means intended to be prescriptive, but merely to demonstrate the features both of our theoretical framework and its practical applications. An obvious application is to use a system such as ours automatically to label low-level geographic data in terms of high-level concepts, particularly in cases where the specific data was not originally collected with those particular high-level concepts in mind. It is also possible to use such a system to *test* different proposed definitions of terms, and compare the results of grounding to the expectations of domain experts. Other directions for future work include the extension of the classification to a larger and more finely discriminating set of hydrographic features, and the incorporation of temporal aspects into the theoretical framework.

Although we have focused here on the domain of geography, and more specifically still, on hydrographic features, the approach we have taken can, we believe, extend to much more general domains. The ability of our framework to accommodate vague and ambiguous natural language terms in a flexible fashion makes it suitable for application to a wide range of fields which have so far been difficult to handle using standard modelling techniques.

References

1. Cameron, W.M., Pritchard, D.W.: Estuaries. In: Hill, M.N. (ed.) The Sea: The Composition of Sea-water, vol. 2, Harvard University Press (1963)
2. Cai, G.: Contextualization of geospatial database semantics for mediating human-GIS dialogues. Geoinformatica 11(2), 217–237 (2007)
3. Zadeh, L.A.: Fuzzy logic and approximate reasoning. Synthese 30, 407–428 (1975)
4. Williamson, T.: Vagueness. Routledge, London (1994)

5. Fine, K.: Vagueness, truth and logic. Synthése 30, 263–300 (1975)
6. Bennett, B.: Application of supervaluation semantics to vaguely defined spatial concepts. In: Montello, D.R. (ed.) COSIT 2001. LNCS, vol. 2205, pp. 108–123. Springer, Heidelberg (2001)
7. Bennett, B.: What is a forest? on the vagueness of certain geographic concepts. Topoi 20(2), 189–201 (2001)
8. Bennett, B.: A theory of vague adjectives grounded in relevant observables. In: Doherty, P., Mylopoulos, J., Welty, C.A. (eds.) Proceedings of the Tenth International Conference on Principles of Knowledge Representation and Reasoning, pp. 36–45. AAAI Press (2006)
9. Randell, D.A., Cui, Z., Cohn, A.G.: A spatial logic based on regions and connection. In: Proc. 3rd Int. Conf. on Knowledge Representation and Reasoning, San Mateo, pp. 165–176. Morgan Kaufmann, San Francisco (1992)
10. Bennett, B.: A categorical axiomatisation of region-based geometry. Fundamenta Informaticae 46(1–2), 145–158 (2001)
11. Allen, J.F.: Towards a general theory of action and time. Artificial Intelligence 23(2), 123–154 (1984)
12. Bennett, B.: Modes of concept definition and varieties of vagueness. Applied Ontology 1(1) (2005)
13. Bennett, B.: Space, time, matter and things. In: Welty, C., Smith, B. (eds.) FOIS 2001. Proceedings of the 2nd international conference on Formal Ontology in Information Systems, Ogunquit, pp. 105–116. ACM, New York (2001)
14. Mallenby, D.: Grounding a geographic ontology on geographic data. In: Logical Formalizations of Commonsense Reasoning: papers from the AAAI Spring Symposium. AAAI Technical Report SS-07-05 (2007)
15. Santos, P., Bennett, B., Sakellariou, G.: Supervaluation semantics for an inland water feature ontology. In: Kaelbling, L.P., Saffiotti, A. (eds.) Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI-05), Edinburgh, pp. 564–569. Professional Book Center (2005)
16. Kokla, M., Kavouras, M.: Fusion of top-level and geographical domain ontologies based on context formation and complementarity. International Journal of Geographical Information Science 15(7), 679–687 (2001)
17. Fonseca, F., Egenhofer, M., Davis, C., Câmara, G.: Semantic granularity in ontology-driven geographic information systems. Annals of Mathematics and Artificial Intelligence 36, 121–151 (2002)
18. Smith, B., Mark, D.M.: Ontology and geographic kinds. In: Proceedings of the International Symposium on Spatial Data Handling (1998)
19. Worboys, M., Duckham, M.: Integrating spatio-thematic information. In: Egenhofer, M.J., Mark, D.M. (eds.) GIScience 2002. LNCS, vol. 2478, pp. 346–361. Springer, Heidelberg (2002)

Applying Spatial Reasoning to Topographical Data with a Grounded Geographical Ontology

David Mallenby and Brandon Bennett

School of Computing
University of Leeds, Leeds
LS2 9JT, UK
{davidm,brandon}@comp.leeds.ac.uk

Abstract. Grounding an ontology upon geographical data has been proposed as a method of handling the vagueness in the domain more effectively. In order to do this, we require methods of reasoning about the spatial relations between the regions within the data. This stage can be computationally expensive, as we require information on the location of points in relation to each other. This paper illustrates how using knowledge about regions allows us to reduce the computation required in an efficient and easy to understand manner. Further, we show how this system can be implemented in co-ordination with segmented data to reason about features within the data.

1 Introduction

Geographic Information Systems are becoming increasingly popular methods of representing and reasoning with geographical data. In order to do this, we require methods of reasoning logically about geographical features and the relations that hold between them, including spatially. Ontologies have been cited as a method to perform this reasoning [1,2,3], but existing methodologies do not handle the inherent vagueness adequately. Features are often dependant on the context in which they are made, with local knowledge affecting definitions. Geographical objects are often not a clearly demarcated entity but part of another object [1,4]. The individuation of entities is therefore more important to geographical domains than to others.

One approach proposed to improve the handling of vagueness is to ground the ontology upon the data [5,6],making an explicit link between the ontology and the data, thus allowing reasoning to be made within the context of the particular data. So we require approaches that will allow spatial reasoning such as Region Connection Calculus (RCC) [7] to be used. RCC is a powerful representation of the principal relations between regions, but it can also be computationally expensive.

In this paper we examine developing the system introduced in [6], which takes topographical data as input and segments into polygons with attached attributes.

The data to be looked at is of the Hull Estuary[1], with the aim being to obtain a method of reasoning about the hydrological features implicit in the data. We examine how this segmented data can be stored effectively, and what is required in order to reason about the RCC relations between given polygons. Finally, we look at how these can be expanded to allow first order logic definitions of inland water features to be entered, with the appropriate regions returned. We do this by applying an example definition to see what results are returned.

2 Motivation

Vagueness is inherent to the geographical domain, with many features being context dependant, as well as lacking precise definitions and boundaries. Vagueness is not a defect of our language but rather a useful and integral part. It is a key research area of the Ordnance Survey[2], where it has been noted that GIS does not handle multiple possible interpretations well. Rather than attempting to remove vagueness, we should allow the user to make decisions about vague features. So rather than segmenting or labelling the image in advance, we require a mechanism for entering logical queries that may incorporate vagueness and can segment accordingly.

With GIS, we need to deal with several layers. We have our initial data level, which represents the points and polygons that make up a topographical map for example. An additional layer is the ontology level, whereby we define features and relations between the data. The ontology level is usually seen as separate to the data level; we reason within the ontology, and return the data that matches our queries. Thus the ontology is devoid of the data context. This has a clear impact upon handling vagueness, where context is important. A proposed improvement to this is to ground the ontology upon the data [5]. By grounding the ontology, we make an explicit link between the ontology and the data, thus allowing reasoning to be made within the context of the particular data.

The symbol grounding problem as proposed in [8] suggests that computers do not actually understand knowledge they are provided, as meanings are merely symbols we attach to objects. There have been no adequate solutions to this problem as yet and it remains an open problem [9] . Ontology grounding does not solve the problem. Rather, it allows the user to decide the meaning of concepts to some extent.

Grounding the ontology upon the data allows reasoning with the data in particular context, thus achieving our previously mentioned requirement of allowing the user control over the features generated. To ground the ontology upon the data, we need to work at both the data level and the ontology level. In [6]

[1] Landsat ETM+ imagery. Downloaded from the Global Landcover Facility (GLCF). Image segmented into water and land then vectorised.
`http://glcfapp.umiacs.umd.ed:8080/esdi/index.jsp`

[2] Ordnance Survey Research Labs: Modelling fuzzy and uncertain features
`http://www.ordnancesurvey.co.uk/oswebsite/partnerships/research/research/data_fuzzy.html`

linearity was shown as an example of such an attribute, and it was shown the work required on both levels to use such an attribute. To expand the system, we are required to implement approaches to generate polygons based upon the spatial relations between regions, such as if they are connected or disconnected.

Spatial reasoning can be computationally expensive, as we require information on the location of all points in relation to a given region. Previous work has looked at the problem at an abstract level [10]. By looking at how the relations are calculated, we can determine methods of reducing the calculations required based upon simpler observations. So instead of explicitly requiring every point location be determined, we could use other information to infer what relations are possible and reduce down our scope until we have our solution.

By implementing an RCC based system, we allow quantitative data to be reasoned with qualitatively. This significantly improves the expressiveness of GIS. This also allows for the individuation of features.

3 The Region Connection Calculus

The Region Connection Calculus (RCC) was introduced in [7]. RCC assumes an initial primitive relation C(x,y), which is true if x and y share a common point). From this initial connected relation, we can derive other relations that hold between two regions. A list of the basic key relations as listed in [11] follows:

$$DC(x,y) \equiv_{df} \neg C(x,y) \tag{1}$$
$$P(x,y) \equiv_{df} \forall z[C(z,x) \rightarrow C(z,y)] \tag{2}$$
$$PP(x,y) \equiv_{df} [P(x,y) \wedge \neg P(y,x)] \tag{3}$$
$$EQ(x,y) \equiv_{df} [P(x,y) \wedge P(y,x)] \tag{4}$$
$$O(x,y) \equiv_{df} \exists z[P(z,x) \wedge P(z,y)] \tag{5}$$
$$DR(x,y) \equiv_{df} \neg O(x,y) \tag{6}$$
$$PO(x,y) \equiv_{df} [O(x,y) \wedge \neg P(x,y) \wedge \neg P(y,x)] \tag{7}$$
$$EC(x,y) \equiv_{df} [C(x,y) \wedge \neg O(x,y)] \tag{8}$$
$$TPP(x,y) \equiv_{df} PP(x,y) \wedge \exists z[EC(x,z) \wedge EC(y,z)] \tag{9}$$
$$NTPP(x,y) \equiv_{df} PP(x,y) \wedge \neg\exists z[EC(x,z) \wedge EC(y,z)] \tag{10}$$

RCC-8 consists of eight of these relations: DC, EQ, PO, EC, TPP, TPPi, NTPP, NTPPi, where TPPi and NTPPi are the inverses of TPP and NTPP respectively. Fig. 1 shows graphically the RCC-8 set. This set is both jointly exhaustive and a pairwise disjoint set of base relations, such that only one can ever hold between two given regions [7]. RCC has previously been proposed as a method of spatial reasoning that could be applicable to GIS, for example in [12] where it was noted that the same set of relations have independently been identified as significant for GIS [13,14].

An additional property that we would like to express is the notion of self-connected regions, such that a region is self-connected if it is not divided into a

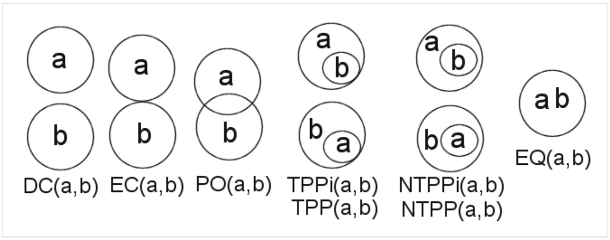


Fig. 1. The RCC-8 relations

number of DC parts. This is important, as in our system we will start with an initial set of segmented polygons, and wish to connect them to form larger regions that satisfy given properties. To do this, we first define a formula sum(x,y) which represents the spatial sum or union of two regions. From this we can define self-connectedness to be equal to the sum of a set of connected regions [11]:

$$\forall xyz[C(z, sum(x, y)) \leftrightarrow [C(z, x) \vee C(z, y)]] \tag{11}$$

$$CON(x) \equiv_{df} \forall yz[EQ(x, sum(y, z)) \rightarrow C(y, z)] \tag{12}$$

So equation 11 states that z represents the spatial sum of regions x and y f all parts of z are either connected to either x or y. This spatial sum is then used in 12 to define self-connectedness (CON); if x is self-connected, any two regions whose spatial sum is equal to x must be connected to each other. Thus x is a single connected region; if we imagine standing in any part of x it would be possible to travel to any other part of x without actually leaving the region.

4 Vagueness in Geography

Vagueness is ubiquitous in geographical concepts [15]. Both the boundaries and definitions of geographical concepts are usually vague, as well as resistant to attempts to give more precise definitions. For example, the definition of a river as given by the Oxford English Dictionary [16] is:

> A large natural flow of water travelling along a channel to the sea, a lake, or another river.

The most obvious example of vagueness is 'large', though other aspects may also be vague such as the boundary between respective channels. But this isn't the only definition for a river; some may differ entirely, others may be more or less restrictive. In comparison, OpenCyc[3] is the open source version of Cyc, which is intended to be the largest and most complete general knowledge base in the world. The definitions of river and stream in OpenCyc are:

> A *River* is a specialisation of *Stream*. Each instance of River is a natural stream of water, normally of a large volume.

[3] OpenCyc `http://www.opencyc.org/`

> A *Stream* is a specialisation of *BodyOfWater*, *InanimateObject-Natural*, and *FlowPath*. Each instance of *Stream* is a natural body of water that flows when it is not frozen.

Again, these are vague and also do not include the restrictions of the water flowing into a particular feature. Yet at the same time, both definitions are perfectly valid within a given context to describe rivers.

The sorites paradox can be easily adapted to illustrate vagueness in geography [3,17], showing that an explicit boundary may not always exist between definitions, such as between rivers and streams. Geographical definitions are also dependant on the context in which they are made. For example, whilst UK rivers are defined usually as permanent flows, in Australia this is not necessarily the case, and thus temporal aspects enter the definition [18]. The application of UK based definitions in Australia could therefore fail to classify some rivers due to their temporal nature, whilst Australian based definitions may overly classify things as rivers when applied in the UK.

The principal approaches for handling vagueness at present are fuzzy logic and supervaluation theory. It is usually the case that the two are presented as opposing theories. However, this in part assumes that vagueness can only take one form, which as discussed in [19] is not true. Rather, there are instances where it is more appropriate to use fuzzy logic and instances where supervaluation theory is better.

The suitability of the two approaches to the proposed system were discussed in [6], where it was noted that supervaluation theory was more applicable as crisp boundaries were produced. This means that we use *precisifications* to represent user decisions and to set contexts.

5 Data Segmentation

In [6], an initial polygon representing the inland water network extending from the Hull estuary was segmented based upon linearity thresholds. This was done by first finding the medial axis of the polygon using a voronoi diagram based approach VRONI [20]. The medial axis of a polygon as first proposed in [21] is defined as the locus of the centre of all the maximal inscribed circles of the polygon. Here, a maximal inscribed circle is a circle that cannot be completely contained within any other inscribed circle in the polygon [22].

However, the medial axis is extremely sensitive to noise and variation along the edge of the input polygon. We want to be able to prune off arcs such that the remaining arcs still represent the topology of the polygon effectively, without disconnecting parts or removing arcs we wish to keep. The approach used to prune the medial axis skeleton here was contour portioning [23,24], which satisfies these requirements. The contours used here are manually defined; whilst an automatic approach is desirable (and work has been done in this area), it is beyond the scope of this project.

The results of using contour partitioning are shown in Fig. 2, where we see the remaining skeleton retains the topology whilst removing unnecessary arcs. This skeleton easily translates into a graph.

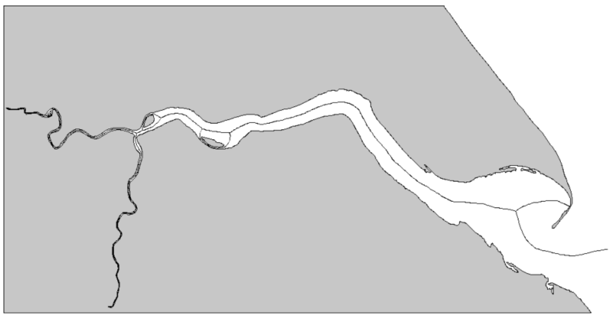


Fig. 2. The results of contour partitioning to reduce the medial axis of the Hull Esturary to a simplified skeleton whilst retaining topology of the shape

The next stage was to use this skeleton to determine linearity. In [6] this was done using a scale invariant approach looking at the variation in widths across stretches of the skeleton. From this we can determine linear lines in the skeleton.

To generate a polygon from this, we determine a left and right side for the skeleton, and combine this with the end points to create a simple polygon. For each side, we use the two boundary points closest to both end nodes, which we already know as these are the points that the maximal inscribed circle at each point touches the boundary.

Once these two points are selected, we find the shortest path between the two along the boundary. This is done by representing the boundary as a graph, and thus a path between is easy to calculate. If no path exists or the length of the path is too great in relation to the distance between the points, then we simply use a single edge with the points as end nodes. An example of this is shown in Fig. 3. This approach guarantees a unique polygon for each line that is simple. We can also use the technique on sets of lines to generate larger polygons.

The initial results of this segmentation stage is a series of segmented polygons, with a label attached representing whether the polygon is linear or non-linear. Further attributes could be used to generate further polygons and labels, such as different linearity measures or size and distance measurements. Some may require further segmentation of the data, whilst others can be performed without segmenting.

The initial results of marking all linear polygons is shown in Fig. 4. Although most parts we would like marked as linear are marked as such, there are some cases that are not. These may be rectified with alternative or refined definitions of linearity. For example, the mouth of the river does not seem linear, because although the width is not varying, the difference in the two banks is significant. Therefore a refined linearity definition may be that a polygon is required to also

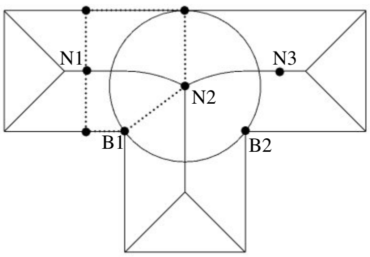


Fig. 3. An example of how a line is translated into a corresponding polygon. For this shape, we have taken the line *N1-N2*. For *N1* we just use both tangent points. For *N2* we choose *B1* as it is closer to *N1*. We then trace along the boundary using the shortest path, with the dotted line representing the resultant polygon. Had the line been *N1-N2-N3*, depending on the length around the boundary between *B1* and *B2* we may replace the path with edge *B1-B2* instead.

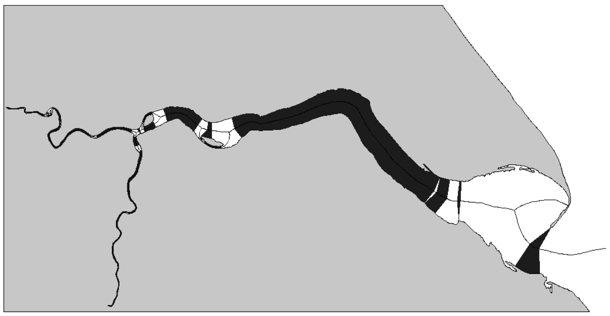


Fig. 4. The results of marking linear stretches, with the original skeleton once again shown. Black sections represent polygons marked as linear with respect to the width.

be linear in relation to its edges, in that the length of the edges should not vary too great from the length of the stretch.

However, there are always likely to be discrepancies in our data, because of variations in actual data in comparison to our abstract notion of a river as a constant line. So we would like a mechanism that can flag up such small discrepancies so that they can be filled in. A method for this was discussed in [6], where the discrepancies were referred to as gaps. To avoid confusion we have introduced the term *interstretch* to represent these features. So using a closeness threshold we can determine which polygons could be 'filled in' at a higher level to generate connected stretches.

6 Data Storage and Querying

Our initial system allows us to segment our data into a series of linear or non-linear polygons, as well as identify *interstretches*. However, we would like to reduce the amount of pre-computed features used, as the aim is to allow a user to generate their own definitions. For example, rather than explicitly calculating *interstretch*, we would like to be able to identify these parts based upon first order logic. So example definitions of stretch and *interstretch* are:

$$\begin{aligned} stretch[l](x) \leftrightarrow\ & CON(x) \wedge P(x, WATER) \wedge linear[l](x) && (13)\\ \wedge\ & \forall y(P(y, WATER) \wedge linear[l](y) \\ \wedge\ & P(x, y) \rightarrow EQ(x, y)) \\ interstretch[c, l](x, y, z) \leftrightarrow\ & stretch[l](x) \wedge stretch[l](y) && (14)\\ \wedge\ & P(z, WATER) \wedge EC(x, z) \wedge EC(y, z) \\ \wedge\ & CON(z) \wedge \forall w(PP(w, z) \\ \rightarrow\ & (close - to[c](w, x) \wedge close - to[c](w, y))) \end{aligned}$$

Here, we use the form *p[v](x)*, where the predicate p is true for a given variable or *precisification* v for a given variable x, So, our previous definition of something being linear if the variation in widths is low translates to linear$[l](x)$, or x is linear for a given *precisification l*. Equation 13 defines stretch as a maximal self-connected region that is water and linear for a given $precisification$. Equation 14 defines an *interstretch* as a self-connected region of water that is connected to two stretches, such that all parts of the *interstretch* are close to the two stretches.

Now, instead of having *interstretch* as a primitive, we have a primitive *close-to*, representing the notion that they are close if the distance between is insignificant. As with linearity, this can be treated as a *precisification*. From these definitions, we wish to define water-channels to be maximal self-connected regions that are made up of stretches or *interstretches*. An initial attempt at representing this logically is:

$$\begin{aligned} waterchannel[c, l](x) \quad \leftrightarrow \quad & CON(x) \wedge P(x, WATER) \wedge \forall w(PP(w, x) && (15)\\ \rightarrow \quad & \exists s(stretch[l](s) \wedge P(w, s) \wedge TPP(s, x)) \vee \\ \exists d, e, f \quad & (interstretch[c, l](d, e, f) \wedge P(w, f) \\ \wedge \quad & TPP(d, x) \wedge TPP(e, x) \wedge TPP(f, x))) \end{aligned}$$

So a water-channel is a self-connected region of water such that all proper parts of the region are either part of a stretch or an *interstretch*, that is also proper parts of the channel.

This is not the only way such a query could be formed, and there may be further refinements required in order to capture exactly our intended definition. In order to represent a query such as water-channel, we require several stages of work. First, we need a data representation that allows more effective querying. We then need to consider how we can test for RCC relations. Finally, we then need a method of producing the union of resultant polygon sets to produce the final water-channel result.

Our aim at each stage is to find a balance between simplicity and computational complexity. The system is intended to use logic definitions that may not be known at this stage. So to accommodate for this, our design should be reasonably easy to understand and adapt, whilst remaining reasonably efficient.

6.1 Data Storage

The winged edge structure [25] and variations such as the half-edge winged structure [26] offer a more effective representation of polygons as opposed to simply storing the corner points. Our initial polygon data can be easily translated into such a structure. We can easily gain a list of all polygons, edges and points in Prolog.

6.2 Calculating the RCC Relations

We now move on to encoding RCC relations such that we can query the system to find the relations between polygons, thus allowing qualitative and quantitative queries. However, this move from an abstract level to the data level is computationally expensive. We therefore wish to reduce the calculations required at each stage in order to speed up the reasoning process. Previous work on an abstract level was done in [10], which illustrated the process could be broken down into a hierarchical tree, reducing the calculations required.

We first reduce down the potential relations that can occur between two polygons. A first approximation is to compare the bounding boxes of each polygon, hereby defined as the smallest rectangle that can entirely contain its polygon. This significantly reduces the initial calculations, and allows us to eliminate relations that are not possible. We do this using an approach similar to Allen's interval Algebra [27]. The algebra represents 13 different relations (hereby referred to as Allen relations) that can occur between two time intervals, as shown in Fig. 5.

If we treat the X-Axis and Y-Axis as separate dimensions, we can determine the Allen relations between two polygons in each axis. We then compare the resulting pair of Allen relations and determine what possible RCC relations these allow. Determining the Allen relations is straightforward; for a given axis we find

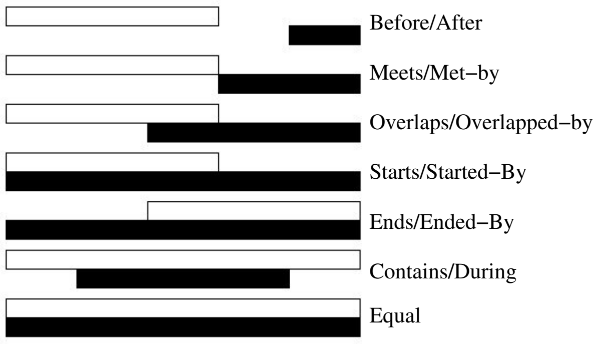


Fig. 5. A graphical representation of the 13 different Allen relations. With the exception of the final relation Equals, the other 12 are in fact 6 pairs of duals. So the first relation represents both white before black and black after white.

the minimum and maximum values of the two polygons and represent as two lines. We can then sort these numbers and determine what Allen relation they correspond to. We repeat for the other axis, so each operation is only working in a single dimension.

This results in a pair of Allen relations, which in turn represent a set of possible RCC relations, as shown in 6. In these examples, we have quickly determined that the first example shows two disconnected polygons, thus no further computation is required. With the other two examples we are left with a set of possible relations. However, we can use these reduced sets to determine the most effective approaches to take next, thus tailoring our deductions to each pair of polygons.

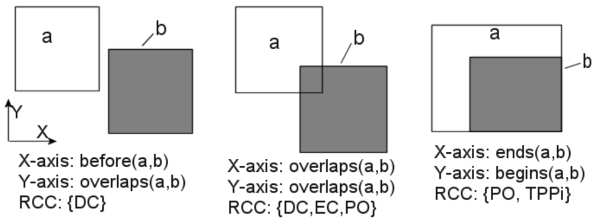


Fig. 6. Examples of how the bounding boxes of two polygons *a* and *b* may be related spatially, and what possible RCC relations these represent. We obtained the Allen relations for the X- and Y-axis', then compared these to see what the set of possible RCC relations are for the polygons.

Theoretically there are 169 different combinations, but in fact there are only 14 different possible combinations, listed in table 1. So we now have a method of reducing the possible RCC relations quickly; we can for example quickly determine polygons which are definitely disconnected.

Table 1. The possible relations as a result of comparing the Allen relations between the X- and Y-axis. Starred relations also have versions replacing TPP/NTPP with TPPi/NTTPi.

Possible RCC combinations from previous stage
DC
DC, EC
DC, EC, PO
DC, EC, PO, TPP *
EC, PO
EC, PO, TPP, NTPP *
EC, PO, TPP, TPPi, EQ
PO
PO, TPP *
PO, TPP, NTPP *

Table 2. The definitions of the RCC-8 relations used in the system. Only 6 are shown here, as TPPi and NTPPi are merely the inverses of TPP and NTPP, respectively.

RCC	Definition in system
DC(X,Y)	There is no intersection between X and Y, and no point of X is inside or on the boundary of Y (and vice versa)
EC(X,Y)	There exists a point that is on the boundary of both X and Y, but there are no points of X inside Y (or vice versa)
PO(X,Y)	There exists either at least one intersection between X and Y, and there are points that are inside one polygon but outside the other
TPP(X,Y)	All points of X are either inside or on the boundary of Y
NTPP(X,Y)	All points of X are inside Y
EQ(X,Y)	All points of X and Y lie on the boundary of each other

Our calculations for RCC relations are based upon the locations of the corners of the polygons to be compared, and whether they are inside, outside or on the boundary of the other polygon in question. In addition, we add to our set all points of intersection between the two polygons that are not already a corner point of a polygon. Table 2 shows the RCC-8 relations defined in terms of the tests that are required in order to make decisions as to the RCC relation between two regions. This is similar to [28], where the spatial domain was also restricted to polygons as opposed to arbitrary points due to the existence of efficient algorithms to handle polygons.

6.3 Intersections

The intersections of two polygons has been studied extensively, in an attempt to improve upon the brute force approach of comparing all lines against all others. More efficient methods are based upon the sweepline approach [29]. The aim of such algorithms is to reduce the comparisons between lines. For our approach, we use our Allen relations based approach to reduce the number of intersection tests.

Using the brute force algorithm as our basis, we order the polygons into two sets of lines. We then take a line in our first set and compare against the bounding box second set, since if an intersection exists the line must touch, intersect or be inside this bounding box. So using our Allen relations approach we can quickly test if the line falls inside the box, and thus eliminate lines that could not intersect the second polygon. For a line that satisfies this criteria, we wish to improve upon simply then comparing the line against all others. We once again use Allen relations, as lines can't intersect if they occur before/after each other in either axis. This once again eliminates many lines, leaving only a small set to be tested.

One final consideration is which of the polygons to use as our first set, as this choice can further speed up the process. Looking at the possible relations, we first see if the relation PP is possible. If so, we use the outer polygon, as our bounding box test would remove no lines if the inner polygon was used. If the relation PP is not possible, we use whichever polygon has fewer lines, as this will remove more intersections in the first part of the test.

So our intersection algorithm uses information previously calculated to speed up the calculation of all intersects whilst remaining simple to understand. Although further work is required to determine the actual efficiency of this approach in comparison to others, it has so far been successfully implemented within Prolog, where it has proven fast enough for the requirements of the project. The result of this stage is a set of points of intersection, which may include existing corner points of a polygon. These can be separated into existing and new points using set operations.

6.4 Points Inside

As with our intersection tests, we wish to reduce down the number of points we test to keep computation time down. Using the bounding boxes generated previously, we can again reduce down the possible points to be all those that are inside or touching the bounding box. This subset is then tested to find which points are inside using a standard test of extending a line horizontally from the point and then counting the number of intersections with the polygon; if the number of intersections is odd the point is inside and if it is even the point is outside. We can reduce the number of intersection tests by using Allen relations to eliminate lines that could not intersect the projected line. How to handle points that lie on the boundary is often an issue for such algorithms. However, we have previously found this set of boundary points in our intersection tests and so can use this set to remove points on the boundary, leaving only points explicitly inside.

6.5 Using the Results with RCC

The results of the previous stages give us a series of sets of points. We can therefore test for RCC relations using set operations on these points, as our previous definitions easily translate into set operations.

First, we find all the potential RCC relations using the Allen relations based approach mentioned earlier. From this stage we can make decisions based on which tests to do; for example if the results of this stage is the set [DC,EC], we know we only need to test for at least one intersection to determine if the answer is DC or EC. So for each of these sets of possible relations we can order the queries to be asked so that they are optimal. We can also find other relations that are implied; if DC is not a member of the list then we know that the two regions are connected, whereas if DC and EC have both been removed we know that at least some part of one region is part of the other.

By using Prolog, we are able to allow for variations of the query. So instead of simply being able to return the relation between two polygons, we can also ask such queries as ”find all polygons that are connected to X” and ”find all pairs of polygons that are externally connected”.

6.6 Building New Regions Based on Queries

The aim of our system is to return regions that match particular queries from an ontology, so we require the system to be able to return sums of regions. For our water-channel example, we need to find all linear polygons, as well as all *interstretch* polygons that connect linear polygons together, and return the results as single connected regions.

We have previously defined self-connected as being the sum of connected regions. This is also applicable to our skeleton and the associated graph, whereby any subset of this graph can be considered self-connected if there is a path between all nodes in the subgraph generated from the subset. As illustrated in Fig. 3, the polygon generated for any given line is simple and self-connected. Thus using this technique on sets of lines is the equivalent of taking the union of all the polygons generated from all connected subsets of the set of lines. We can thus infer that the resulting polygon is self-connected if the skeleton used to generate it is a connected graph. To produce all linear polygons, we simply find the set of all linear lines and convert into a graph, then generate polygons for each maximal self-connected component, where maximal means that there does not exist an edge that is connected to our component that is not part of the component. For *interstretch*, we find the set of lines used to produce the polygons that satisfy our definition and repeat the process above (thus some non-linear polygons may have more than one *interstretch* proper part).

To generate our maximal self-connected polygons, we an approach similiar to a breadth-first search, marking neighbours of polygons as we find them. An example of this process is shown in Fig. 7.

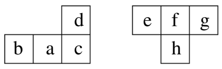


Fig. 7. An example of how self-connected regions are marked. Starting at a our breadth-first search returns the set a,b,c,d, and then finds polygons remain, so repeats to get e,f,g,h. These sets can then be spatially summed to return maximal self-connected regions.

So for our water-channel example, our criteria is that all polygons are either linear or an *interstretch* between linear polygons. We find this set, then using our breadth-first search type approach, travel through all connections until all have been visited. The result is sets of maximal self-connected regions.

We now wish to generate the sum of these polygons, to create our new polygon representing a water-channel. For this, we can use our winged edge structure coupled with our polygon generation approach. Firstly, if we have a set of polygons that are only ever EC, we can find the union by removing all edges that are incident to two or more polygons and traveling along the remaining edges, returning a polygon when we reach our start point (if further edges remain, these are holes and we simply repeat the process until there remains no unvisited edges). If we have overlapping polygons, then we can use our polygon generation approach to create a union by combining the sets of lines that make up our polygons and generating a new polygon that represents their union. We could simply use this approach for the union of all polygons, but this is slower than the union of existing polygons.

So our sum operation combines the previous operations; we find our set of candidate polygons, find maximal self-connected sets via our breadth-first search and then form the union either through union or additional polygon generation. Further operations such as spatial difference or intersection could also be developed, but are beyond the scope of this work. The results of running our water-channel query are shown in Fig. 8, where we see stretches have successfully been joined to form larger regions.

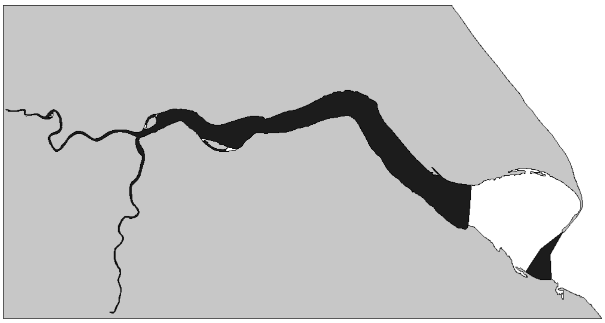


Fig. 8. The results of running the water-channel query. The linear stretches were segmented first, then a query representing interstretch was run. Finally, a water-channel was defined to be the self-connected sum of these two features, such that we find the set of polygons that are either linear or an interstretch, then used our traveling algorithm to find maximal self-connected sets.

7 Future Work

The next key stage of the research is into further logical definitions that can be used to represent inland water features, and thus construct an ontology that represents such features. This may require further primitive functions to be implemented in addition to the linearity and closeness tests present in the system. However, the aim is to keep such primitives to a minimum, as the system is intended to be as general as possible. Thus new features should be defined in logical definitions at the ontology level.

The system has been developed in Prolog and at present is designed to use first order logic based queries. However, a possible extension would be to integrate more closely with a language such as OWL, which can be inputted into Prolog [30,31]. By creating the ontology in OWL, we allow interaction with the semantic web, whilst retaining the segmentation level in Prolog allows us to reason with vague features and ground the ontology upon the data effectively. This is also proposed in [32], where it is shown that OWL cannot effectively handle RCC without modifying the rules of the language. However, such revisions may remove other favourable features of OWL, hence a hybrid system is more appropriate.

8 Related Work

The problem of combining qualitative and quantitative data has previously been discussed in [33]. Here, the combination of different levels of information are discussed, such that the intention is to bridge the gap between the primitive level of points, lines and polygons, and the object level describing the spatial relations and definitions of features.

Like the Allen relation based approach used in this paper, transitivity tables are formed representing the possible relations between different primitives. Thus, spatial relations can be calculated by deductive processes as opposed to computational geometric algorithms (or at least a reduced usage of such algorithms). In this paper, we have expanded this to show how intersection and point locations can be determined using similar approaches to reduce the computational geometry requirements.

As previously mentioned, [10] discussed a hierachical approach to determining RCC relations. Moreover, the calculations were converted to boolean terms, such that the problem becomes one of the closure of half-planes. On the other hand, in this paper decisions are made based on both the intersection and location of points with respect to the regions. Thus a richer level of detail is deductible.

Another approach to deducing the spatial relationships is to use constraint logic programming [34], as discussed particularly in [35]. Such an approach offers an interesting alternative, but is reliant on the efficiency of the constraint logic solver used, and as discussed in [35] further work is required to improve such an approach for effective implementation.

9 Conclusions

In this paper we have demonstrated a method of calculating and using RCC relations on segmented topographical data, thus allowing integration with an ontology grounded upon the data. This improves the handling of vagueness within geographical features, as we can make decisions on features based upon the context in which they are made, as opposed to using predefined regions.

We have shown that although the calculation of RCC relations is computationally expensive, we can still implement the relations effectively by using other knowledge representation approaches such as Allen's interval algebra. Further, Allen's relations were adapted to provide simple but effective methods of calculating the intersections and locations of points of polygons in relation to each other, although more efficient algorithms may exist. Further work is therefore required to determine the efficiency of the approaches discussed here, or whether a hybrid approach using deductive methods in conjunction with other computational geometric algorithms, thus providing the most efficient environment overall.

We have shown how previous queries used in the system could be written in first order logic instead of being specified in the code. Although these may require further clarification, this does highlight the possibility of defining features in first order logic. We have also shown how maximal self-connected regions satisfying such queries can be generated. Finally, we have shown where the work is intended to progress and how this will improve the handling of vagueness within geographical features.

Acknowledgments. I am grateful to our industrial partner Ordnance Survey, for their contribution towards PhD funding. I am also grateful to Brandon Bennett and Allan Third for their support. Finally, I am extremely grateful to the reviewers for their comments that helped me prepare the final text.

References

1. Fonseca, F., Egenhofer, M., Agouris, C., Cmara, C.: Using Ontologies for Integrated Geographic Information Systems. Transactions in Geographic Information Systems 6(3), 231–257 (2002)
2. Guarino, N., Welty, C.: Evaluating Ontological Decisions with Ontoclean. Communications Of The Acm 45(2), 61–65 (2002)
3. Varzi, A.C.: Vagueness in Geography. Philosophy & Geography 4(1), 49–65 (2001)
4. Smith, B., Mark, D.M.: Ontology and Geographic Kinds. In: Poiker, T., Chrisman, N. (eds.) Proceedings of the Tenth International Symposium on Spatial Data Handling, Burnaby, BC, pp. 308–320. Simon Fraser University (1998)
5. Jakulin, A., Mladenić, D.: Ontology Grounding. In: Proceedings of the 8th International multi-conference Information Society IS-2005, Ljubljana, Slovenia, pp. 170–173 (2005)
6. Mallenby, D.: Grounding a Geographic Ontology on Geographic Data. In: Amir, E., Lifschitz, V., Miller, R. (eds.) 8th International Symposium on Logical Formalizations of Commonsense Reasoning (Commonsense 2007), Stanford, USA, pp. 101–105 (2007)
7. Randell, D.A., Cui, Z., Cohn, A.G.: A Spatial Logic-Based on Regions and Connection. In: Principles Of Knowledge Representation And Reasoning: Proceedings Of The Third International Conference (Kr 92), pp. 165–176. Morgan Kaufmann Pub Inc., San Mateo (1992)
8. Harnad, S.: The Symbol Grounding Problem. Physica D 42(1-3), 335–346 (1990)
9. Taddeo, M., Floridi, L.: Solving the Symbol Grounding Problem: a Critical Review of Fifteen Years of Research. Journal Of Experimental & Theoretical Artificial Intelligence 17(4), 419–445 (2005)
10. Bennett, B., Isli, A., Cohn, A.G.: A System Handling RCC-8 Queries on 2D Regions Representable in the Closure Algebra Half -Planes. In: Moonis, A., Mira, J.M., de Pobil, A.P. (eds.) Methodology and Tools in Knowledge-Based Systems. LNCS, vol. 1415, pp. 281–290. Springer, Heidelberg (1998)
11. Giritli, M.: Who Can Connect in RCC? In: Günter, A., Kruse, R., Neumann, B. (eds.) KI 2003. LNCS (LNAI), vol. 2821, pp. 565–579. Springer, Heidelberg (2003)

12. Bennett, B.: The Application of Qualitative Spatial Reasoning to GIS. In: Abrahart, R. (ed.) Proceedings of the First International Conference on GeoComputation, vol. I, pp. 44–47 (1996)
13. Egenhofer, M.J.: Reasoning about Binary Topological Relations. In: Günther, O., Schek, H.-J. (eds.) SSD 1991. LNCS, vol. 525, pp. 144–160. Springer, Heidelberg (1991)
14. Egenhofer, M.J., Franzosa, R.D.: Point-Set Topological Spatial Relations. International Journal Of Geographical Information Systems 5(2), 161–174 (1991)
15. Bennett, B.: What is a Forest? on the Vagueness of Certain Geographic Concepts. Topoi-An International Review Of Philosophy 20(2), 189–201 (2001)
16. Simpson, J., Weiner, E. (eds.): Concise Oxford English Dictionary, 11th edn. Oxford University Press, Oxford (2004)
17. Varzi, A.C.: Vagueness, Logic, and Ontology. The Dialogue. Yearbooks for Philosophical Hermeneutics 1, 135–154 (2001)
18. Taylor, M.P., Stokes, R.: When is a River not a River? Consideration of the Legal Definition of a River for Geomorphologists Practising in New South Wales, Australia. Australian Geographer 36(2), 183–200 (2005)
19. Dubois, D., Esteva, F., Godo, L., Prade, H.: An information-Based Discussion of Vagueness. In: 10th IEEE International Conference On Fuzzy Systems, Vols 1-3 - Meeting The Grand Challenge: Machines That Serve People, pp. 781–784. IEEE PRESS, New York (2001)
20. Held, M.: Voroni: An Engineering Approach to the Reliable and Efficient Computation of Voronoi Diagrams of Points and Line Segments. Computational Geometry-Theory And Applications 18(2), 95–123 (2001)
21. Blum, H.: Biological Shape and Visual Science. Journal Of Theoretical Biology 38(2), 205–287 (1973)
22. Ge, Y.R., Fitzpatrick, J.M.: Extraction of Maximal Inscribed Disks from Discrete Euclidean Distance Maps. In: 1996 Ieee Computer Society Conference On Computer Vision And Pattern Recognition, Proceedings. Proceedings / Cvpr, Ieee Computer Society Conference On Computer Vision And Pattern Recognition, pp. 556–561. IEEE PRESS, New York (1996)
23. Bai, X., Latecki, L.J., Liu, W.Y.: Skeleton Pruning by Contour Partitioning. In: Kuba, A., Nyúl, L.G., Palágyi, K. (eds.) DGCI 2006. LNCS, vol. 4245, pp. 567–579. Springer, Heidelberg (2006)
24. Bai, X., Latecki, L.J., Liu, W.Y.: Skeleton Pruning by Contour Partitioning with Discrete Curve Evolution. IEEE Transactions On Pattern Analysis And Machine Intelligence 29(3), 449–462 (2007)
25. Baumgart, B.G.: Winged Edge Polyhedron Representation. Technical report, Stanford University, 891970 (1972)
26. Mantyla, M.: Introduction to Solid Modeling, vol. 60949. W. H. Freeman & Co, New York (1988)
27. Allen, J.F.: Maintaining Knowledge about Temporal Intervals. Communications Of The Acm 26(11), 832–843 (1983)
28. Haarslev, V., Lutz, C., Möller, R.: Foundations of Spatioterminological Teasoning with Description Logics. Principles of Knowledge Representation and Reasoning, 112–123 (1998)
29. Bentley, J.L., Ottmann, T.A.: Algorithms for Reporting and Counting Geometric Intersections. Ieee Transactions On Computers 28(9), 643–647 (1979)
30. Laera, L., Tamma, V., Bench-Capon, T., Semeraro, G.: Sweetprolog: A system to Integrate Ontologies and Rules. In: Antoniou, G., Boley, H. (eds.) RuleML 2004. LNCS, vol. 3323, pp. 188–193. Springer, Heidelberg (2004)

31. Wielemaker, J.: An optimised Semantic Web Query Language Implementation in Prolog. In: Gabbrielli, M., Gupta, G. (eds.) ICLP 2005. LNCS, vol. 3668, pp. 128–142. Springer, Heidelberg (2005)
32. Grütter, R., Bauer-Messmer, B.: Towards Spatial Reasoning in the Semantic Web: A Hybrid Knowledge Representation System Architecture. In: Fabrikant, S., Wachowicz, M. (eds.) The European Information Society: Leading the Way With Geo-information. Lecture Notes in Geoinformation and Cartography, vol. XVII, p. 486 (2007)
33. Abdelmoty, A., Williams, M., Paton, N.: Deduction and Deductive Database for Geographic Data Handling. In: Symposium on Large Spatial Databases, pp. 443–464 (1993)
34. Jaffar, J., Maher, M.J.: Constraint Logic Programming: A Survey. Journal of Logic Programming 19/20, 503–581 (1994)
35. Almendros-Jimenez, J.: Constraint Logic Programming Over Sets of Spatial Objects. In: Proceedings of the 2005 ACM SIGPLAN workshop on Curry and functional logic programming, Tallinn, Estonia, pp. 32–42. ACM Press, New York (2005)

An Ontology for Grounding Vague Geographic Terms

Brandon BENNETT, David MALLENBYand Allan THIRD
School of Computing,
University of Leeds, UK
e-mail: {davidm,brandon,thirda}@comp.leeds.ac.uk

Abstract. Many geographic terms, such as "river" and "lake", are vague, with no clear boundaries of application. In particular, the spatial extent of such features is often vaguely carved out of a continuously varying observable domain. We present a means of defining vague terms using *standpoint semantics*, a refinement of the philosophical idea of supervaluation semantics. Such definitions can be grounded in actual data by geometric analysis and segmentation of the data set. The issues raised by this process with regard to the nature of boundaries and domains of logical quantification are discussed. We describe a prototype implementation of a system capable of segmenting attributed polygon data into geographically significant regions and evaluating queries involving vague geographic feature terms.

Keywords. Vagueness, Geographic Entities, Query Answering

1. Introduction

In recent years increasing attention has been paid to the ontology of geographic entities. A major motivation for this has been the recognition that the implementation of computational Geographic Information Systems (GIS) which can support functionality for sophisticated data manipulation, querying and display requires robust and detailed specification of the semantics of geographic entities and relationships. A second, more philosophical, motivation for attention to this domain is that it presents a concrete manifestation of many ontological subtleties. For instance issues of individuation, identity and vagueness arise in abundance, when one tries to give precise specifications of the meanings implicit in geographic terminology [1,2,3,4].

Our concerns in this paper will relate to both these motivations. On the one hand, we will examine the particular ontological issues associated with interpretation of vague geographic feature terms (especially hydrological terms such as 'lake' and 'river) and will outline how the general semantic framework of *standpoint semantics* can be applied to provide a framework within which such vagueness can be represented explicitly. We shall also see that when deployed in conjunction with a geometry-based theory of feature segmentation, this semantics gives an account of how vague features are individuated with respect to the material structure of the world. On the other hand, we shall also be very much concerned with the implementation of certain GIS functionality for which a coherent theory of vagueness and its relation to individuation is a necessary pre-requisite.

We look specifically at the problem of interpreting logical queries involving vague predicates with respect to a geographic dataset. We shall assume that such data takes a typical form consisting of a set of 2-dimensional polygons, each of which is associated with one or more labels describing the type of region that the polygon represents. This is a simplification of geographic data in general, which will often include other types of information such as point or line entities, altitudes, additional cartographic entities such as icons or textual strings and meta-annotations relating to the provenance or accuracy of data items. Moreover, the data will not normally consist simply of a set of entities but a complex data structure supporting indexing and various kinds of computational manipulation of data elements. Nevertheless, labelled 2-dimensional polygons form the core of most real geographic information systems.

The structure of the rest of the paper is as follows. In section 2 we present an overview of the basic theory of standpoint semantics, which is a refinement of supervaluation semantics. Section 3 considers the ontological principles that govern the ways in which one can divide up the geographic realm into distinct regions corresponding to geographic features. In section 4 we consider the implementation of a geographic query interpretation system and see that severe difficulties arise regarding finding an appropriate computationally tractable domain of quantification. We shall see that finding a solution to this problem requires a theory of individuation (such as was developed in section 3). Section 5 then looks in detail at the implementation of our prototype system, which provides a limited proof of concept of our theoretical analysis. Finally, concluding remarks and discussion of future work are given in section 6.

2. Standpoint Semantics

In making an assertion or a coherent series of assertions, one is taking a *standpoint* regarding the applicability of linguistic expressions to describing the world. Such a standpoint depends partly on one's beliefs about the world. This epistemic component will *not* be considered in the current paper: we shall assume for present purposes that one has correct knowledge of the world — albeit at a certain level of granularity (which in the context of geographic information is likely to be rather coarse). The other main ingredient of a standpoint, which we *will* be concerned with here, is that it involves a linguistic judgement about the criteria of applicability of words to a particular situation. This is especially so when some of the words involved are vague. For instance, one might take the standpoint that a certain body of water should be described as a 'lake', whereas another smaller water-body should be described as a 'pond'.

The notion of 'standpoint' is central to our analysis of vagueness. Vagueness is sometimes discussed in terms of different people having conflicting opinions about the use of a term. This is somewhat misleading since even a person thinking privately may be aware that an attribution is not clear cut. Hence a person may change their standpoint. Moreover this is not necessarily because they think they were mistaken. It can just be that they come to the view that a different standpoint might be more useful for communication purposes. Different standpoints may be appropriate in different circumstances. The core of standpoint semantics does not explain why a person may hold a particular standpoint or the reasons for differences or changes of standpoint, although a more elaborate theory dealing with these issues could be built upon the basic formalism.

In taking a standpoint, one is making somewhat arbitrary choices relating to the limits of applicability of natural language terminology. But a key feature of the theory is that all assertions made in the context of a given standpoint must be mutually consistent in their use of terminology. Hence, if I take a standpoint in which I consider Tom to be tall, then if Jim is greater in height than Tom then (under the assumption that height is the only attribute relevant to tallness) I must also agree with the claim that Jim is tall.

Our *standpoint semantics* is both a refinement and an extension of the *supervaluation* theory of vagueness that has received considerable attention in the philosophical literature (originating with [5]). Supervaluation semantics enables a vague language to be logically interpreted by a set of possible precise interpretations (*precisifications*). This provides a very general framework within which vagueness can be analysed within a formal representation. Here we do not have space to give a full account of supervaluation semantics. Detailed expositions can be found in the philosophical literature (e.g. [6]).

By itself, supervaluation semantics simply models vagueness in terms of an abstract set of possible interpretations, but gives no analysis of the particular modes of semantic variability that occur in the meanings of natural language vocabulary. A key idea of standpoint semantics is that the range of possible precisifications of a vague language can be described by a (finite) number of relevant *parameters* relating to objectively observable properties; and the limitations on applicability of vocabulary according to a particular standpoint can be modelled by a set of *threshold values*, that are assigned to these parameters. To take a simple example, if the language contains a predicate Tall (as applicable to humans), then a relevant observable is 'height'. And to determine a precisification of Tall we would have to assign a particular threshold value to a parameter, which could be called tall_human_min_height.[1] In general a predicate can be dependent on threshold valuations of several different parameters (e.g. Lake might depend on both its area and some parameter constraining its shape.) Thus, rather than trying to identify a single measure by which the applicability of a predicate may be judged, we allow multiple vague criteria to be considered independently.

In the current paper (as in several previous papers on this topic [4,8,9]) we shall assume that standpoints can be given a model theoretic semantics by associating each standpoint with a threshold valuation. In so far as standpoints may be identified with an aspect of a cognitive state, this idea is perhaps simplistic. It is implausible that an agent would ever be committed to any completely precise value for a threshold demarcating the range of applicability of a vague predicate. Cognitive standpoints are more plausibly associated with constraints on a range of possible threshold values (e.g. if I call someone tall then my claim implies an upper bound on what I consider to be a suitable threshold for tallness — the threshold cannot be higher than the height of that person) rather than exact valuations of thresholds.[2] But in the context of cartographic displays, we may more plausibly propose that any useful depiction of geographic entities corresponding to geographic terms should be determined by application of precise criteria associated

[1]Vague adjectives tend to be context sensitive in that an appropriate threshold value depends on the category of things to which the adjective is applied. This is an important aspect of the semantics of vague terminology but is a side issue in relation to our main concerns in the current paper. Here we shall assume that vague properties are applied uniformly over the set of things to which they can be applied. To make this explicit we could always use separate properties such as Tall-Human and Tall-Giraffe, although we won't actually need to do this for present purposes. A formal treatment of category dependent vague adjectives is given in [7].

[2]This elaboration of the status of standpoints in relation to thresholds is being developed in a separate strand of research.

with the term, and that such criteria require a definite value to be associated with every threshold parameter.

A *threshold valuation* appropriate for specifying a standpoint in relation to the domain of hydrographic geography might be represented by:

$V = [\mathsf{pond\_vs\_lake\_area\_threshold} = 200m^2,\ \mathsf{river\_min\_linearity\_ratio} = 3,\ ...]$

Here one parameter determines a cut-off between ponds and lakes in terms of their surface area and another fixes a parameter indicating a linearity[3] requirement used to characterise rivers.

3. Geographic Entities and their Boundaries

As noted by Smith and Mark in [3], the geographic domain is distinctive in that typical geographic objects are attached to the world and are not easily demarcated in the way that physically detached objects such as organisms and artifacts can be. Thus the individuation of geographic features is ontologically problematic. Previously, Smith [10,11] had drawn attention to a distinction between of *bona fide* and *fiat* boundaries:

> *Fiat boundaries are boundaries which owe their existence to acts of human decision or fiat, to laws or political decrees, or to related human cognitive phenomena. Fiat boundaries are ontologically dependent upon human fiat. Bona fide boundaries are all other boundaries.* [11]

A paradigm case of a fiat boundary is the border of a country whose location does not depend on any physical boundary in the world.[4] In [3] it is argued that, in so far as they may be said to exist at all, the boundaries of mountains must be fiat because they rely on human judgement for their demarcation. Whilst we have no objection to this use of terminology, we believe that there is a significant difference between the national border and mountain cases. Although any particular demarcation of a border around a mountain is certainly dependent on human judgement, the range of reasonable boundaries is also to a large extent determined by the lie of the land.

In order to understand this distinction more clearly, it will be instructive first to consider another kind of boundary, which we call an *implicit geometrical boundary*. Such

[3]Note that we use the term ‘linearity’ to refer to elongation of form rather than straightness. Thus we would describe a river as linear, even though it may bend and wiggle. A geometric characterisation of linearity of this form has been presented in previous papers [8].

[4]Of course particular national boundaries may be aligned to physical boundaries such as the banks of rivers but this is a contingent circumstance.

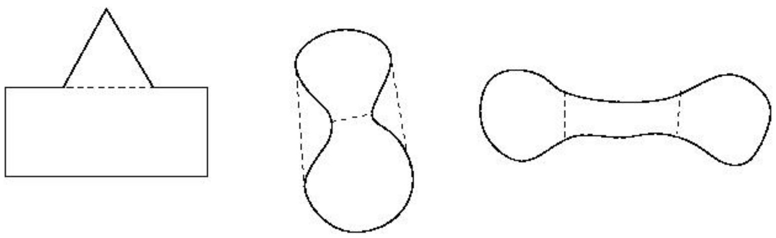


Figure 1. Implicit geometric boundaries.

a boundary does not lie upon an actual discontinuity in the fabric of the world but follows a line that is determined by the spatial configuration of other boundaries, which may be either bona fide or fiat (or a combination of both). Such boundaries are depicted in Figure 1. On the left we see a region within which there is an implicit boundary between a rectangular portion and a triangular projection. The middle region involves a 'neck' flanked by concavities, and these features also imply certain geometric boundaries.

In the region on the right, implicit boundaries are not so clear cut. In describing the region one may be inclined to mention two bulbous parts joined by an elongated section. This suggests the existence of implied boundaries between these three portions. These are examples of *vague boundaries* whose course is hinted at, but not completely determined, by the geometric form of a concrete boundary.

This analysis suggests a four-fold classification of kinds of boundary:

- *Bona fide* boundaries between matter or terrain types.
- *Fiat* boundaries imposed on the world by conscious agents
- *Implicit Geometrical* boundaries determined geometrically in relation to bona fide and/or fiat boundaries.
- *Vague* boundaries, which can be made precise in relation to some standpoint taken on an appropriate precisification of vague properties or relations. The resulting precise properties/relations will then determine a geometrical boundary (which will be demarcated in relation to bona fide and/or fiat boundaries).

The latter two types could be regarded as special cases of *bona fide* or *fiat* boundaries. However, it is not completely clear to which camp they should be assigned. Whether implicit geometric boundaries are considered *bona fide* or *fiat* depends upon whether one takes a Platonist or constructivist view of the existence of geometrical entities. It may be argued that vague boundaries must involve an element of human judgement and hence must be *fiat*. However, if one takes a Platonist view of implied geometric boundaries, then vague boundaries also have a *bona fide* underpinning.

Meta-terminological confusion notwithstanding, it is clear that many kinds of natural geographic feature have vague boundaries and that the demarcation of these is determined by a combination of physical properties of the world and human judgement. We believe that the way that this occurs can be explained by standpoint semantics.

This is well illustrated by consideration of the division of a water system into lakes and rivers. As described in [8,9], such a segmentation can be achieved by specifying geometric predicates that can identify linear/elongated *stretches* of a water system (as represented by polygons) and distinguish these from from expansive (lake-like) regions of the system. Indeed these have been implemented in prototype GIS software (GEOLOG). A feature of these predicates is that they depend on a small number[5] of parameters, for which specific values must be chosen to obtain a segmentation into lakes and rivers. This parameterised variability of geometry-based predicates can be directly described within the framework of standpoint semantics. Each choice of parameters given to the computational segmentation procedure corresponds to a standpoint taken with respect to the interpretation of the terms 'river' and 'lake'.

Of course more factors are relevant to the meanings of these natural language terms; so this shape-based characterisation is only part of a full explanation of the usage of

[5]In our simplest implementation there is just one such parameter, but better results have been obtained by adding a second parameter.

hydrographic terms. For instance, water flow is such an essential part of our concept of river that it might appear that no satisfactory characterisation of rivers could omit this aspect. But, GIS and other cartographic data rarely includes flow information (such information is hard to obtain and to depict); and yet, it seems that humans usually have little difficulty in identifying rivers represented in a 2-dimensional map display. One explanation for this is that, although flow is an important criterion in its own right, the dynamic behaviour of water distributed over an uneven but approximately horizontal surface is closely correlated (due to physical laws) with the geometry of the projection of the water system onto the horizontal plane. Thus, given our knowledge of the way the world works, we can infer a lot about flow just from a 2-dimensional representation of a water system.

Having said that, we would in future like to incorporate flow into our hydrographic ontology and believe that can be done within the general framework that we propose. A simple approach would be to take a field of flow vectors (this would have to be interpolated from some set of data points) and segment the water system according to a threshold on flow magnitude, so that we would obtain polygons labelled as either flowing or (comparatively) still. We could then define types of hydrographic feature in terms of a combination of both shape-based and flow-based characteristics. (We could also investigate correlations between the two types of characteristic.)

In many cases there is ambiguity with regard to which objective properties are relevant to a particular natural language term (e.g. is salinity relevant to lake-hood). Such controversy may be modelled by allowing standpoints to vary not-only in respect of threshold parameter values but also in the assignment of definitions to terms. Although this is clearly an important issue, it will not be considered in the present paper.

3.1. Land Cover Types and their Extensions

As well as by referring to geographic features, the geographic domain is very often described in terms of its terrain or land cover. A region may be wooded, ice covered, rocky etc.. In some cases the boundaries of such regions may be clearly *bona fide*, whereas in others, especially where there is a transitional region between terrain types (e.g. jungle $\leftrightarrow$ scrub-land $\leftrightarrow$ desert), the boundary may be vague. In either case there is certainly a physical basis to land cover demarcations; and in the case where the boundary is vague, the range of reasonable demarcations can be modelled within standpoint semantics in terms of thresholds on appropriate parameters relating to properties of the Earth's surface.

However, apart from such vagueness, there is another characteristic of land cover that is potentially problematic for computational manipulation of geographic data. Land cover types are *downward inherited*, meaning that, if a region is covered by a given type of terrain, then all sub-regions are also covered by this terrain type.[6,7] It is also clear that, if we have a set of regions all covered by the same terrain type, then the mereological sum of these regions is also covered by that type. Both these conditions are entailed by

[6]This kind of inheritance of properties among spatial regions is discussed in detail in [12].

[7]In fact downward inheritance will not normally apply beyond a certain fineness of granularity, but for present purposes we shall ignore this complication and assume that we do not have to worry about fine grained dissections of the world.

the following equivalence, which applies to properties that may be said to be manifest homogeneously over extended regions of space:[8]

TT-hom) $\mathsf{HasTerrainType}(r, t) \leftrightarrow \forall r'[\mathsf{P}(r', r) \rightarrow \mathsf{HasTerrainType}(r', t)]$

With regard to computational manipulation of land cover information this homogeneity property has both positive and negative implications. On the negative side it suggests that if a GIS ontology includes land cover terms that can be predicated of arbitrary regions of geographic space, then the set of regions that can instantiate such predicates, must include arbitrary sub-regions of its base polygons. For example, if the ontology includes a predicate $\mathsf{Water}(r)$, meaning that r is completely covered with water then this will be satisfied by arbitrary dissections (and unions) of those data polygons labelled with the 'water' attribute.[9]

But on the positive side it is clear that one would never want to actually exhibit all water-covered polygons. Once we give the total extent of a given terrain type we can simply exhibit this, and the fact that all its sub-regions also have that type is implicit. It is obvious to a GIS user that an extended region of blue represents water and moreover that every sub-region of the blue area is also wet. (By contrast it is also obvious that, where regions corresponding to countries are indicated on a map, their sub-regions are not themselves countries.) Hence, although a geo-ontology must certainly take account of the downward inheritance of land cover types, it seems that it should be possible to do this without requiring an explicit representation of arbitrary subdivisions of the Earth's surface.

4. Handling Geographic Data: Queries, Definitions and Domains of Quantification

In order to construct an ontology-based GIS capable of answering queries expressed in terms of formally defined geographic concepts and evaluated with respect to geographic data represented by labelled polygons, the following rather challenging problems must be addressed:

P1) The ontology must define all terms in a way that enables their extensions to be somehow computable from the spatial properties and attributes of polygon data.

P2) The formalism must enable the characterisation of features with vague boundaries.

P3) The implementation must be able to deal with regions with implicit geometrical boundaries that are determined by but not explicitly present in the base polygons, without explicitly modelling potentially infinite geometrical dissections of space.

P4) The implementation must be able to take account of the fact that predicates relating to spatially homogeneous properties (such as terrain types) are downwardly inherited (without explicitly modelling arbitrary dissections of space).

P5) An effective method of ontology-based geographic query evaluation must be implemented.

[8]In natural language, such properties are typically associated with mass nouns.

[9]The situation here can be contrasted with the case of a non-downward-inherited feature type predicate such as $\mathsf{Lake}(r)$. In this case, even if we consider geographic space to include arbitrary polygons, only a finite number of these could satisfy this predicate. Hence, it is plausible that instances of $\mathsf{Lake}(r)$ can be obtained by some finitary computation over the base water polygons. Indeed, we have implemented such a computation.

4.1. Spatial Regions and Relations

In order to address **P1**, we need a method of determining the spatial relations that hold between two regions. We use the Region Connection Calculus (RCC) [13], which allows us to express topological relations between regions and to use these to define features involving complex configurations of spatial parts.

However, the standard models of RCC are infinite domains — typically, the sets of all regular closed (or regular open) subsets of Cartesian space (either two or three dimensional). Relating such models to actual data is problematic, because in a computational implementation, one can only refer explicitly to a finite set of entities. Real spatial data usually consists of finite sets of polygons, but the domain of quantification in the standard RCC would include not only these polygons but also all possible ways of carving these up into further polygons.

Our approach to solving this problem is to find a way of working with a finite set of regions, which is adequate to characterise the domain in so far as is relevant to any given spatial query. As discussed in [14], the full set of regions contains many regions we are not interested in, such as tiny regions or obscure shapes with convoluted boundaries, thus we would prefer to work only with the set of regions that we can derive useful or interesting features from. For example, if we are interested only in inland water features, we are only interested in segmenting up the inland water regions, and it may be sufficient to represent the land as a single polygon. We thus choose to restrict our domain of regions to polygons, as previously proposed in [15,16]. To expand upon this, our domain consists of polygons which are initially generated from the data, with further polygons derived from this polygonal information through predicates using standpoints. In [8], we showed how the calculation of the RCC relations between a set of polygons can be performed efficiently.

A problem that arises with such an approach is the generation of this domain. Ideally we would generate all possible polygons to begin with, but this would be too large a set to work with when answering queries. Instead, we start with an initial set of polygons designed to represent the basic separation of *matter types* [17], thus each initial polygon is filled by some specified matter type. These polygons may be further segmented during the query interpretation process.

Such further segmentation will normally arise from shape related or metrical predicates being used in a query (or occurring in the definition of a predicate used in a query). Moreover, since shape and measurement predicates will often be vague, these can correspond to different geometrical conditions, and thus different ways of carving up the initial polygons, according to the standpoint relative to which the query is evaluated.

4.2. Demarcating Vague Regions

Our approach to demarcating vague regions is of course based upon standpoint semantics. This has been explained above and also in several previous papers [8,9] and some further details will be given below in describing our prototype implementation. Here we just give a brief overview. Our procedure first determines a medial axis skeletonisation of the region occupied by a given land cover type. This is then used to segment the region into linear and expansive sub-regions based on threshold values of certain parameters determined by a given standpoint. Vague regions corresponding to different types of geographic feature can then be specified definitionally, in terms of the distribution of land cover types over topological configurations of the regions in this segmentation and over regions derived by further geometrical dissection of these regions.

4.3. Controlled Quantification over Geometrically Derived Regions

We now turn to problem **P3**. One method of constructing an ontology that is computationally tractable over a concrete domain, is to constrain quantification in such a way that all entities (in our case spatial regions) that are relevant to the evaluation of a given formula are either present in an initial finite set of entities, or are members of further finite sets that can be effectively computed from the initial entity set. We now sketch a relatively limited modification of first order logic in which this can be achieved.

Let Base be the finite set of entities (e.g. polygons) present in our data-set. Restricting quantification to range just over entities in Base is clearly tractable, so we can certainly allow quantification of the form:

QB) $(\forall x \in \mathsf{Base})[\phi(x)]$

Many domains have a natural Boolean structure which may be useful for defining properties and relations. Thus in the spatial domain we are often concerned with sums, intersections and complements of regions. Let Base^* be the elements of a Boolean Algebra over Base. We may then allow quantification of the form:

QB*) $(\forall x \in \mathsf{Base}^*)[\phi(x)]$

If Base is finite then so is Base^*. So quantification can still be evaluated by iterating over the domain. But unfortunately Base^* will be exponentially larger than Base, so it would almost certainly be impractical to do this in a real application. However, there is another way of extending the domain of quantification, which is both more controllable and more flexible.

Let $\Gamma(t_1, \ldots, t_m;\ x_1, \ldots, x_n)$ be a relation, such that given any m-tuple of ground terms $\langle t_1, \ldots, t_m\rangle$, one can effectively compute the set of all n-tuples $\langle x_1, \ldots, x_n\rangle$, such that $\Gamma(t_1, \ldots, t_m;\ x_1, \ldots, x_n)$ holds. We may call $\langle t_1, \ldots, t_m\rangle$ an input tuple and $\langle x_1, \ldots, x_n\rangle$ an output tuple. The condition on Γ means that for any given finite set of input tuples there is a finite set of output tuples such that some pair of input and output tuples satisfies Γ. For example, Γ might be a spatial relation $\mathsf{BisectNS}(r; r_1, r_2)$ which is true when r_1 and r_2 are the two parts of r obtained by splitting it into northern and southern parts across the mid-line of its extent in the north-south dimension. Another example is $\mathsf{Concavity}(r, r')$, where given an input polygon r there are a finite number of polygons r' corresponding to concavities of r (i.e. maximal connected regions that are parts of the convex hull of r but do not overlap r).

We shall call relations of this kind *effective generator relations*. They are simply logical representations of a certain kind of algorithm that could be implemented in computer software — and indeed much of the functionality of a GIS depends on algorithms of this kind. Given an effective generator relation Γ, we can now define the following form of controlled quantification:

QEGR) $(\forall x_1, \ldots, x_n : \Gamma(t_1, \ldots, t_m;\ x_1, \ldots, x_n))[\ \phi(x_1, \ldots, x_n)\ ]$

Here, the variables $t_1, \ldots, t_m$ must be already bound to wider scope quantifiers, which can be either quantifications over Base or over domains specified by other effective generators. Hence, the range of each variable is restricted either to Base or to a set of entities that can be computed from Base by applying algorithms corresponding to a series of effective generator relations.

Semantically, **QEGR** is interpreted as equivalent to:

- $(\forall x_1, \ldots, x_n)[\,\Gamma(t_1, \ldots, t_m;\, x_1, \ldots, x_n) \rightarrow \phi(x_1, \ldots, x_n)\,]$

4.4. Spatially Homogeneous Properties and Downward Inheritance

So far we have not implemented any mechanism for handling downward inheritance. Instead we have circumvented the issue by limiting our predicates to those satisfied either by maximal components of uniform land cover, or by regions derived from these by particular geometrical decompositions. For instance, we define 'linear stretches' of water which are geometrically dissected (relative to a given standpoint) from the total region of water. In the future we would like to handle spatially homogeneous properties by representing their logical relationship to base polygons.

4.5. Query Evaluation

We express a query by means of the notations $? : \phi$ representing a test as to whether ϕ is true in relation to a given data-set and $?(x) : \phi(x)$, which means: return a list of all entities e_i such that $\phi(e_i)$ is true as determined by interpreting the symbols of ϕ in relation to the data-set. More generally, $?(x_1, \ldots, x_n) : \phi(x_1, \ldots, x_n)$ would return a list of n-tuples of entities satisfying the given predicate. In our context, the entities returned will normally be polygons. Query variables cannot occur within any of the quantifiers of our representation, however they can be identified with values of these variables by the use of an equality predicate.

Since queries will be interpreted in relation to actual geographic data, it is natural to use a *model-based* approach to query evaluation.[10] General purpose model building systems, such as MACE [19], allow consistency checking of arbitrary first order formulae, by checking all possible assignments to predicates. But in our case we have a single interpretation of basic predicates that can be derived directly from the geographic dataset. Thus, we can compute sets of all tuples satisfying the predicates that occur in a query and then evaluate the query formula over this model.

Boolean connectives can be evaluated in an obvious way, but the treatment of quantifiers is somewhat more complex. Since quantification is restricted to range over either base polygons or derived polygons generated by the **QEGR** quantifiers, this means that the domain of regions that must be considered is finite. By examining the structure of nested **QEGR** quantifiers occurring in a query, we can determine sequences of spatial function applications which, when applied to the base polygons, will generate all polygons that are relevant to that query. Once these polygons have been computed, quantifiers can be evaluated over this extended domain. Our current prototype does not explicitly include the **QEGR** quantification syntax, but implements a simplified version of this

[10]Model-based reasoning has been applied in various areas of AI. For instance, a similar approach to ours has been used in Natural Language Processing [18].

mechanism. It is geared towards evaluating queries containing a limited range of predicates and generates domains of polygons that are sufficient to deal with these. This will be described in the next section.

5. Implementation within a Prototype GIS

We now give some details of our GIS prototype which we call GEOLOG. The system is implemented in Prolog and operates on several hydrographic data-sets covering estuarine river systems in the UK. The system implements geometric shape decomposition algorithms based on a number of parameters. These are linked to an explicit representation of shape predicates using a first order formalisation in which the parameters attached to predicates are interpreted according to standpoint semantics. First order queries can be evaluated and their instantiations depicted on a cartographic display.

5.1. Shaped-Based Properties and Segmentation

Since queries may contain RCC relations describing topological relations between regions, a database of RCC relations over all stored polygons is maintained. This requires a considerable amount of storage but means that these relations do not have to be recomputed whenever a new query is executed, which greatly speeds up query answering times. As described in [20,8] segmentation of regions into linear and expansive parts is computed using a *medial-axis* approach which is supported by use of the VRONI software package [21]. The idea is to measure width variation along the medial axis. Given a medial axis point p of region r which is distance d from the edge of r, we compute the maximum and minimum distances, max, min, to the edge of r of all medial axis points within distance d of p. The value l=$\mathsf{max}/\mathsf{min}$ gives a useful measure of the width variation at p. $l = 1$ means the width is constant, and a value of 1.2, for example, means that there is a 20% width variation in a section of the medial axis centred at p along a length equal to the width at p. Using this value as a standpoint parameter, the predicate $\mathsf{Stretch}(r)$, corresponding to the vague concept of a ‘linear stretch of water’ is defined. This is a maximal connected water region all of whose medial axis points have a value of l less than a given threshold.

5.2. Query Evaluation

In developing an effective implementaton, we wanted to minimise both the number of polygons stored in the system and the time required to construct polygons by geometrical computation. This led us to an approach of ‘just in time’, incremental expansion of the domain. The basic idea is that that when presented with a query, GEOLOG ensures that all polygons relevant to its interpretation are generated before evaluating the query as a whole. But it then stores the generated polygons as they are likely to be required again for subsequent queries.

The initial dataset consists simply of a partition of the geographic space comprising polygons labelled with the basic land cover types: *land*, *sea* and (fresh) *water*. Queries relating to the base polygons themselves can be answered straightforwardly, although they are of little interest as they do not take any account of the semantics of geographic features. However, a number of higher level geometric and hydrographic predicates are also available for use in queries. Each of these predicates is associated with an algorithm for expanding the domain of polygons by geometrical computations, to include additional polygons corresponding to all their possible instances. When a query containing one or more of these non-basic predicates is entered, the domain is first expanded according to the associated algorithms (in general this must be done recursively until a fixed point is reached), and the newly generated polygons are labelled with appropriate attributes. Once this procedure has has been carried out, the dataset contains polygons corresponding to all possible instances of predicates occurring in the query. Quantifiers can now be evaluated by iterating over polygons in this expanded dataset.

For instance, if one enters a query $\mathsf{Stretch}(x)$ GEOLOG would perform a linearity segmentation relative to a given standpoint, so that the required linear and expansive polygons are generated. We can now answer queries involving reference to stretches or to any concepts that have been defined in terms of linear and/or expansive polygons. A user of the system has direct access to the threshold assignment defining the standpoint and can modify the thresholds. When this is done the system must recompute the segmentation, and this in turn will lead to different polygons being returned from queries that depend on the segmentation.

5.3. Results of Querying for Stretches and Rivers

Results of executing the query $\mathsf{Stretch}(x)$ with different input datasets and linearity parameter thresholds are shown in Figure 2. The images on the left correspond to a threshold of 1.2, whereas those on the right are for a threshold of 1.4. Thus, the interpretation on the right takes a more liberal view of what can count as linear than the one on the left.

As is clear from Figure 2, the artificial concept of ‘linear water stretch’ does not correspond directly to the natural concept of ‘river’. Typically a river will consist of many such stretches, interspersed with more expansive areas of water, corresponding to bulges in the watercourse. We experimented with a range of threshold parameters governing how loosely or strictly the predicate ‘linear’ is interpreted; but found that there is no threshold that yields a natural interpretation of ‘river’. If we use a loose definition that allows bulges to be classified as parts of a stretch, we find that very expansive, lake-like water regions become incorporated into stretches. But if we tighten the linearity threshold to rule out obvious lakes, then rivers must be consist of fragmented sequences of stretches.

In order to circumvent this problem, we propose that a river should indeed be modelled as a sequence of stretches interspersed by bulges. To make this precise we have introduced a further artificial concept of *interstretch*. This is a water region that is expansive but such that all its parts are ‘close’ to a water stretch, where closeness is defined by a second threshold applied to a suitable geometric measure. This enables us to incorporate small bulges into rivers without needing to unduly weaken our general criteria for identifying linear water features. As described in [8], this has been found to interpret the concept river in a very plausible way.[11]

The introduction of interstretches might at first sight appear to be an *ad hoc* hack. However, we believe that a plausible general explanation can be given as to why this seems to work. In classifying a vague feature, we suggest that one is looking for criteria

[11] Further complications arise from the branching structure of water systems. These have been only partially solved and are a topic of ongoing work.

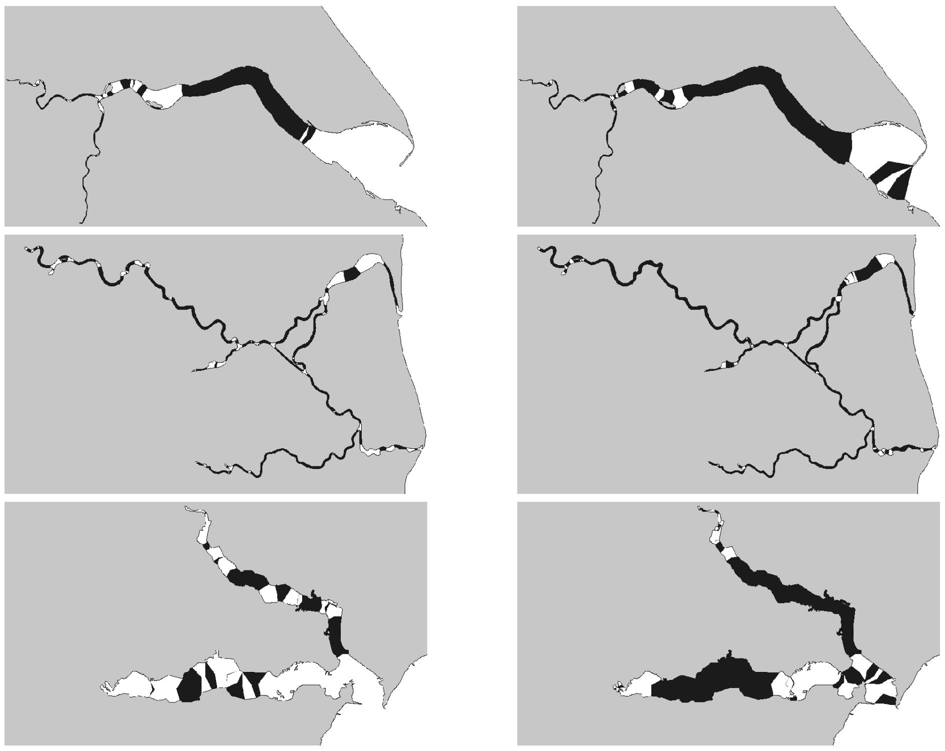


Figure 2. A comparison of marking 'linear water stretches' relative to different The top images are of the Humber Estuary, the middle images are of the Norfolk Broads at Great Yarmouth and Lowestoft. The bottom images are of the Stour And Orwell Estuary.

that are satisfied globally by a region but is also prepared to allow exceptions in regard to small parts of the region that deviate from these criteria. For instance, to classify a surface as approximately planar, one is looking for a global approximation to a plane but will accept small areas where the surface departs considerably from planarity, which are regarded as insignificant bumps on the surface. We thus plan to apply a similar approach to classifying other types of geographic feature.

6. Conclusion

We have described a variety of ontological issues that complicate the issue of defining and individuating geographic regions and features. From theoretical analysis of the semantics of vagueness and of computational manipulation of geometric decompositions of polygonal data, a possible architecture for implementing an ontology-based GIS is taking shape. Our current prototype gives a strong indication that this can lead to a new kind of GIS in which geographic terminology is grounded upon data *via* rigorous definitions rather than *ad hoc* segmentations. However, much work remains to be done, both in terms of specifying a more extensive geographic ontology and also in relation to developing a more flexible and efficient query answering mechanism.

Acknowledgements

We are grateful to the EPSRC for supporting this work *via* grant EP/D002834 and to the Ordnance Survey, for their contribution towards funding a PhD on this topic. Valuable comments from the reviewers are also gratefully acknowledged.

References

[1] Achille C. Varzi. Vagueness in gegraphy. *Philosophy and Geography*, 4:49–65, 2001.

[2] Brandon Bennett. What is a forest? on the vagueness of certain geographic concepts. *Topoi*, 20(2):189–201, 2001.

[3] Barry Smith and David M. Mark. Do mountains exist? towards an ontology of landforms. *Environment and Planning B: Planning and Design*, 30(3):411–427, 2003.

[4] Paulo Santos, Brandon Bennett, and Georgios Sakellariou. Supervaluation semantics for an inland water feature ontology. In L.P. Kaelbling and A. Saffiotti, editors, *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI-05)*, pages 564–569, Edinburgh, 2005.

[5] Kit Fine. Vagueness, truth and logic. *Synthèse*, 30:263–300, 1975.

[6] Timothy Williamson. *Vagueness*. The problems of philosophy. Routledge, London, 1994.

[7] Brandon Bennett. A theory of vague adjectives grounded in relevant observables. In Patrick Doherty, John Mylopoulos, and Christopher A. Welty, editors, *Proceedings of the Tenth International Conference on Principles of Knowledge Representation and Reasoning*, pages 36–45. AAAI Press, 2006.

[8] David Mallenby and Brandon Bennett. Applying spatial reasoning to topographical data with a grounded ontology. In F. Fonseca, M. Andrea Rodrígues, and Sergei Levashkin, editors, *GeoSpatial Semantics, proceedings of the second international conference*, number 4853 in Lecture Notes in Computer Science, pages 210–227, Mexico City, November 2007. Springer.

[9] Allan Third, Brandon Bennett, and David Mallenby. Architecture for a grounded ontology of geographic information. In F. Fonseca, M. Andrea Rodrígues, and Sergei Levashkin, editors, *GeoSpatial Semantics, proceedings of the second international conference*, number 4853 in Lecture Notes in Computer Science, pages 36–50, Mexico City, November 2007. Springer.

[10] Barry Smith. On drawing lines on a map. In Andrew Frank and Werner Kuhn, editors, *Spatial Information Theory — Proceedings of COSIT'95*, number 988 in Lecture Notes in Computer Science, pages 475–484, Berlin, 1995. Springer.

[11] Barry Smith. Fiat objects. *Topoi*, 20(2):131–148, 2001.

[12] Alberto Belussi and Matteo Cristani. Mereological inheritance. *Spatial Cognition and Computation*, 2:467–494, 2000.

[13] D. A. Randell, Z. Cui, and A. G. Cohn. A spatial logic-based on regions and connection. In *Principles Of Knowledge Representation And Reasoning: Proceedings Of The Third International Conference (Kr 92)*, pages 165–176. Morgan Kaufmann Pub Inc, San Mateo, 1992.

[14] Ian Pratt and Dominik Schoop. A complete axiom system for polygonal mereotopology of the real plane. *Journal of Philosophical Logic*, 27(6):621–661, 1998.

[15] Volker Haarslev, Carsten Lutz, and Ralf Möller. Foundations of spatioterminological reasoning with description logics. In *Principles of Knowledge Representation and Reasoning*, pages 112–123, 1998.

[16] Rolf Grütter and Bettina Bauer-Messmer. Towards spatial reasoning in the semantic web: A hybrid knowledge representation system architecture. In S. Fabrikant and M. Wachowicz, editors, *The European Information Society: Leading the Way With Geo-information*, volume XVII of *Lecture Notes in Geoinformation and Cartography*, page 486, 2007.

[17] Brandon Bennett. Space, time, matter and things. In *FOIS '01: Proceedings of the international conference on Formal Ontology in Information Systems*, pages 105–116, New York, NY, USA, 2001. ACM.

[18] Patrick Blackburn and Johan Bos. *Representation and Inference for Natural Language. A First Course in Computational Semantics*. CSLI, 2005.

[19] William McCune. Mace4 reference manual and guide. *CoRR*, cs.SC/0310055, 2003.

[20] David Mallenby. Grounding a geographic ontology on geographic data. In E. Amir, V. Lifschitz, and R. Miller, editors, *Proceedings of the 8th International Symposium on Logical Formalizations of Commonsense Reasoning (Commonsense-07)*. AAAI, 2007.

[21] Martin Held. VRONI: An engineering approach to the reliable and efficient computation of voronoi diagrams of points and line segments. *Computational Geometry: Theory and Applications*, 18(2):95–123, 2001.