THE HONG KONG
POLYTECHNIC UNIVERSITY
香港理工大學

The Hong Kong Polytechnic University

Department of Computing

An Extension of the Crisp Ontology for Uncertain Information Modeling – Fuzzy Ontology Map

Lam Ho Wang, Toby

A Thesis Submitted in Partial Fulfillment

of the Requirements for

the Degree of Doctor of Philosophy

August 2006

Certificate of Originality

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgement has been made in the text.

_______________________(Signed)

__Lam Ho Wang, Toby__(Name of student)

Abstract

In the current World Wide Web (WWW), users find it difficult to locate relevant information using search engines. This may be due to the fact that the current World Wide Web lacks semantic markup. One of the possible solutions for this problem is Semantic Web. In the latest Semantic Web technology, descriptive markup languages, such as Resource Description Framework (RDF) and Web Ontology Language (OWL), were proposed to model the web content in a machine-readable way which assists information gathering and automatic searching by software agents. Since these ontology markup languages deal with 'hard' semantics with the description and manipulation of crisp data, they are not capable to represent uncertain information.

This thesis proposes an extension of the current ontology representation which supports uncertain information modeling. The extension is called Fuzzy Ontology Map (FOM) which is based on the integration of fuzzy theory and graph theory. By considering an ontology as a graph, an ontology graph can be constructed by using a vertex to represent a subject or literal value and an arrow to represent a predicate. Each edge in the ontology graph has a fuzzy membership value. A FOM is a connection matrix which collects the membership values between classes in the ontology graph. Thus, a fuzzy ontology could be created by using the FOM and the ontology document (RDF/OWL). This research also defines a set of algorithms for inferring fuzzy relationships in an FOM. It is possible to use an FOM to develop real-world applications and systems that can deal with imprecise or vague information. To demonstrate how FOM works, two prototype applications were developed, SemTour:HK and iJADE FreeWalker (iJFW).

SemTour:HK is a tourist information portal integrating an ontology with the FOM to allow users to perform fuzzy searches. The average processing time for a fuzzy search is around 1.9s which is 0.4s longer than the exact match searching. The prototype was tested on 20 novice users. 80% of the users felt that the system can help tourists find tourist information in Hong Kong. 75% of the users agreed that the fuzzy search function is useful.

iJFW is an intelligent agent-based tourist guiding system which is used in mobile devices. It was developed based on the integration of intelligent agent technology (IAT) and Global Positioning System (GPS) with FOM. There are two FOMs in the system: i) a preference FOM in the client side and that logs the users' cuisine interests; ii) an accommodation FOM in the server side for fuzzy searching of accommodation choices. iJFW was tested on a 36.6kbit/s wireless connection. The average response time for a fuzzy search is around 25s. Proportionally, the average processing time for gathering the result for 3G wireless connections would be 3.4s. 30 candidates were invited to answer a questionnaire after using the system. 67% of the candidates felt that the fuzzy search function was useful for finding information about a particular cuisine. 17 candidates thought that iJFW could replace tourist guidebooks. 13 candidates though that iJFW would be a subsidiary tool for traveling.

Publications

Papers Accepted/Published

1. Toby H.W. Lam, 'Fuzzy Ontology Map - A Fuzzy Extension of the Hard-Constraint Ontology,' *Proceedings of 2006 IEEE / WIC / ACM International Conference on Web Intelligence (WI 2006), Hong Kong, China, December 2006*, IEEE Computer Society, pp. 506 – 509.
2. Toby H.W. Lam and Dr. Raymond S.T. Lee (2006), 'An Intelligent Ontology-based Agent Framework for Semantic Web Service: Ontia iJADE,' *Knowledge-Based Intelligent Information and Engineering Systems (LNCS 4252)*, pp. 637 – 643, Springer-Verlag, Berlin Heidelberg.
3. Toby H.W. Lam and Dr. Raymond S.T. Lee (2006), 'iJADE FreeWalker - An Ontology-based Tourist Guiding System,' *Knowledge-Based Intelligent Information and Engineering Systems (LNCS 4252)*, pp. 644- 651, Springer-Verlag, Berlin Heidelberg.
4. Janice W.Y. Hui, Toby H.W. Lam and Dr. Raymond S.T. Lee (2005), 'The Design and Implementation of an Intelligent Negotiation Shopping System,' *Multiagent and Grid Systems*, Vol. 1 (3), pp. 131-146

5. Tony W.H. Ao Ieong, Toby H.W. Lam, Alex C.M. Lee and Raymond S.T. Lee (2005), ‘iJADE Tourist Guide : A Mobile Location-Awareness Agent-Based System for Tourist Guiding,’ *Knowledge-Based Intelligent Information and Engineering Systems (LNAI 3681)*, pp. 671-677, Springer-Verlag Berlin Heidelberg

Paper Submitted

6. Toby H.W. Lam and Raymond S.T. Lee, ‘iJADE FreeWalker – An Intelligent Ontology Agent-based Tourist Guiding System,’ submitted *International Journal of Software Engineering and Knowledge Engineering (under second review)*

7. Toby H.W. Lam and Raymond S.T. Lee, ‘iJADE FreeWalker: The Design and Implementation of an Ontology Agent Based Tourist Guiding and Servicing System,’ submitted to *Multiagent and Grid Systems*

Acknowledgements

I would like to take this opportunity to express my sincere gratitude to the following bodies:

1. The Department of Computing of the Hong Kong Polytechnic University for providing excellent facilities and resources for my studies and research;
2. My academic supervisor, Dr. James N.K. Liu, who has provided unreserved, constructive, valuable advice and guidance throughout my studies;
3. My ex-supervisor Dr. Raymond S.T. Lee for the useful discussions on my research;
4. My parents, sister and Hera Wong for their continuous and tremendous support, encouragement, tolerance as well as their patience through the years of my studies; and
5. Martin Kyle for his tremendous support and patience in proof-reading of my papers and thesis.

Without the support and assistance of these people, I would not have been able to continue my study and research, nor complete this thesis.

Table of Contents

List of Figures

List of Tables

Chapter 1. Introduction

In 2001, Tim Berners-Lee, James Hendler and Ora Lassila published an article about the vision of semantic web in Scientific American (Berners-Lee, Hendler and Lassila, 2001). They described Semantic Web as a new form of web content that should be meaningful to computers (such as software agents), as well as people. The web content in the World Wide Web can be applied in order to provide 'truly' useful applications to the end-users. The future direction of WWW should aim at this ultimate goal.

In the current World Wide Web, people always find information from the Internet by using search engines. If a user wants to buy a book from the Internet, he/she may go to Amazon.com, enter the book title to the search engines, browse the information and then buy the book. At this stage, it is still not possible to ask a computer to do the whole process for us automatically. One reason for this is that web pages are designed to display information to users, instead of computers. This limits the computer to analyzing the content of the web. Thus, Berners-Lee and other researchers proposed a new form of web, Semantic Web, which would make the web content accessible and understandable to both machines and people. By using Semantic Web, computers such as software agents can search information and perform actions automatically and intelligently. Besides, lots of works can be done by using software agents. For instance, the software agents can then help the user to find and buy the cheapest book from the Internet, to arrange a meeting with a client to suit their schedule, find the shortest path to reach a destination.

At present, web pages are mainly created by HyperText Markup Language (HTML)

(HTML, 1999). HTML is a simple markup language used for presenting information to the users. Since the main focus of HTML is for presentation, it is difficult to let the software agents process the content or information in a standardized way. Due to this limitation, more descriptive markup languages, such as Resource Description Framework (RDF) (RDF, 2004) and Ontology Web Language (OWL) (OWL, 2004), were proposed. These markup languages are able to model the web content in a machine-readable way which assists information gathering and searching by software agents in an automatic way. The web resources modeled by using RDF and OWL are called ontologies. An ontology is a specification of a conceptualization which is used to describe the objects and the relations between them in a domain (Gruber, 1993). However, there is a structural problem in these markup languages. These ontology markup languages only can deal with hard semantics (either true or false and there is nothing in between) in the description and manipulation of crisp data. Due to this limitation, it is not possible to represent uncertain information by using current ontology representation. The fact is that much of our real-world knowledge consists of uncertain or imprecise information. This thesis shows how to extend the crisp ontology to represent uncertain information.

The following sections briefly introduce the motivations of the research, contributions and organization of this thesis.

1.1 Motivation of Research

The two-valued-based logical methods in Semantic Web are not able to process soft semantics. Fuzzy logic is a good bridge between human ‘soft logic’ and machine ‘hard logic’. This work is motivated by Zadeh’s concept of Web IQ (WIQ) (Zadeh, 2003), which described Web intelligence as:

“In moving further into the age of machine intelligence and automated reasoning, we have

reached a point where we can speak, without exaggeration, of systems which have a high machine IQ (MIQ)… In the context of the Web, MIQ becomes Web IQ, or WIQ, for short"

Zadeh showed a view of evolving search engines into some kind of question-answering systems. Since current Semantic Web and ontology tools are based on bivalent logic, they have intrinsic limitations on addressing problems especially when the information is ill-structured, uncertain and imprecise. To further move towards to WIQ, Zadeh recommended that it is better to abandon the bivalence in current Semantic Web and to adopt fuzzy logic.

The work described in this thesis is about developing an extension of the current ontology representation which supports uncertain information modeling. The ontology markup language was extended so that each path in the ontology graph could include fuzzy information. Such extension is called Fuzzy Ontology Map (FOM) which is based on the integration of fuzzy theory and graph theory. A Fuzzy Ontology Map (FOM) has been developed to provide a solution to enhance the current ontology representation (such as RDF and OWL). It is not necessary to change the syntax of the current ontology representation. Thus, FOM is suitable for Semantic Web applications. FOM provides a feasible solution for fuzzy searching. Besides, a user can define his/her own preference FOM for information filtering. The applications developed based on FOM would give a personalization experience for the user. The main aim of this research is to investigate and to develop an extension for the ontology, which can support uncertain knowledge and information representation.

1.2. Contributions

The goal of this work is: i) to define a way to utilize the structure of ontology and the fuzzy relationships within them, ii) to show how the fuzzy relation inference can be integrated into applications. The ultimate goal is to create a framework that is suitable to model real-world knowledge such that the information presented to the user is more useful.

To achieve this goal, first a connection matrix, Fuzzy Ontology Map (FOM), was defined. FOM collected the membership values between each class in the ontology. In the ontology graph, each path can have a membership value to represent fuzzy information. The membership value is a real value between 0 and 1. *Fuzzy ontology* was defined as an ontology which supports the representation of uncertain information. The fuzzy ontology is created by using RDF/OWL with FOM. Then, two types of fuzzy set were also defined: *symmetric* and *asymmetric*. A symmetric fuzzy set is the one in which the fuzzy relation between two classes is non-directional such as a level of similarity. The degree of similarity is symmetric as there is no difference between 'class A similar to class B' and 'class B similar to class A' (non-directional). It means that there is no difference in the two (opposite) direction (A->B = B->A). An asymmetric fuzzy set is the one in which the fuzzy relation between two classes is directional such as level of preference. Suppose class A is preferred to class B, then there is a difference in the opposite direction. It means that if class A is preferred to class B (A->B), then it is not the same in opposite direction (B->A). By using these definitions and FOM, a set of algorithms was further derived for inferring fuzzy relationships within the FOM. If two classes in the ontology are not directly connected, the fuzzy membership values would be calculated by the paths that connect them in an FOM.

Two applications were developed to demonstrate how the inferred fuzzy relation in FOM can be used. The first one is called SemTour:HK, a web-based tourist information portal which integrates tourist ontology with the accommodation of a FOM. Users are allowed to perform fuzzy search in the accommodation page. When the user is looking for accommodation, he/she can perform a fuzzy search to search for similar accommodation type in the database. The second application is called iJADE FreeWalker (iJFW), a mobile-based tourist guiding system. It was developed based on the integration of ontology, FOM, agent technology and GPS technology. There are two FOMs in the system: i) preference FOM on the client side; ii) accommodation FOM on the server. The preference FOM allows a user to set his/her own cuisine preference. Besides, like the SemTour:HK, it allows users to perform fuzzy searching to search for similar cuisine types.

This thesis illustrates how to extend the current crisp ontology representation. FOM is proposed for extending the crisp ontology. The Author's contribution can benefit research in the semantic web, multi-agent systems and fuzzy search systems. To show the feasibility of FOM, twp prototype systems were implemented to demonstrate various aspects of the FOM architecture. These systems include i) SemTour:HK – a tourist information portal, ii) iJFW – a mobile tourist guiding system. Through these works the Author has shown that using applications developed based on Fuzzy Ontology Map offers fuzzy searching function to the users.

In summary, the first contribution of this thesis is a novel representation in agent technology, Fuzzy Ontology Map (FOM), which extends the current crisp ontology. By using FOM, a fuzzy ontology was developed which supports representations of uncertainty. The second contribution is a set of algorithms for inferring the fuzzy relations in an FOM. The third contribution is the technical analysis of how to create an FOM. Most other research lacks technical information, but this information is significant for implementation. Thus, the technical details on how to create the FOM and state the reason why FOM is created in that way are presented. The fourth contribution is the implementation of two prototypes namely SemTour:HK and iJADE FreeWalker. These prototypes demonstrate how FOM works and the benefits to end-users.

1.3. Organization

An outline of the structure of this thesis is as follows:

- Chapter 1 presents a general introduction to the whole thesis.
- Chapter 2 introduces the Semantic Web. This chapter provides an overview and

motivation of Semantic Web. It also describes the related work about extending crisp ontology.

- Chapter 3 introduces Fuzzy Ontology Map. This chapter shows how to extend the crisp ontology to become a fuzzy ontology by using the proposed Fuzzy Ontology Map (FOM). First, the theoretical details about an FOM and a set of algorithms for inferring fuzzy relationship in an FOM are presented. Then, the technical details on how to construct the fuzzy ontology by using FOM is shown. Finally, the experimental results of the FOM is presented.
- Chapter 4 presents a prototype system – SemTour:HK. SemTour:HK is a tourist information portal which combines the semantic web technology with the FOM. By using FOM, SemTour:HK supports fuzzy searching.
- Chapter 5 shows another prototype system – iJADE FreeWalker (iJFW). iJFW is a mobile tourist guiding system which is developed based on the integration of agent technology, Semantic Web and FOM. This application uses the FOM to provide recommendations to users according to the user's preference.
- Chapter 6 concludes this thesis. It also suggests some future research work.

Chapter 2. Background

2.1 Introduction

In the current World Wide Web, users find it difficult to locate relevant information using search engines. This may be due to the fact that the current World Wide Web lacks semantic markup. The inventor of the World Wide Web, Tim-Berners Lee, foresaw this problem in 2001. He wrote an article with James Hendler and Ora Lassila to propose a solution, Semantic Web (Berners-Lee, Hendler and Lassila, 2001). Semantic Web has a number of components such as agent technology, markup language, and ontology. Ontology is a hierarchical relationship between terms within a domain that specifies defined terms and the relationships between those terms. Ontology is modeled by a number of markup languages such as Resources Description Framework (RDF) (RDF, 2004) and Web Ontology Language (OWL) (OWL, 2004). These markup languages are World Wide Web Consortium (W3C) recommended standards for modeling ontologies. However, there are some limitations in using these markup languages to model ontology. Since the markup language are based on first-order-logic, one of the limitations is that it is not possible to use these languages to model uncertain knowledge. In this chapter, first background information about the World Wide Web and Semantic Web are given. Then, it shows the core limitations of the current ontology and describes related work on extending the current crisp ontology.

2.1.1 Organization of the Chapter

The rest of this chapter describes the background information and related work. Section 2.2 describes background information about the World Wide Web. Section 2.3 illustrates the idea of Semantic Web and its main components such as Resource Description Framework (RDF) and Web Ontology Language (OWL). Section 2.4 states the problem of the hard-constraint of the current ontology markup language. Section 2.5 shows the related work about extension of the crisp ontology. Section 2.6 presents a brief conclusion.

2.2 World Wide Web

In the current World Wide Web, web pages are created by using a markup language called HyperText Markup Language (HTML) (HTML, 1999). Using HTML, one can create web pages with simple hypertext and other information to be displayed in a web browser, since this markup language was originally designed for information presentation, as a result, the current web architectures face the problem that machines are unable to process and to interpret the web information efficiently.

At present, if one wishes to find specific information on the Internet, one can use popular search-engines such as Google[1], Yahoo[2] and AltaVista[3]. These keyword-based search engines are the main tools for today's web. Keyword searching works by using techniques such as frequency-inverse document frequency (tf-idf) (Salton and Buckley, 1988) to ascertain the importance of a word in a document. Such approaches, however, have a number of problems. First, keyword searches locate web pages by using input keywords but without

[1] http://www.google.com
[2] http://www.yahoo.com
[3] http://www.altavista.com

reference to semantics. The use of keyword searches leads to a high recall and low precision rate in the search engines results. In addition, the results are highly sensitive to the input keywords. If we input incorrect keywords into the search engines, it may lead to low or no recall in the retrieved pages (Antoniou and van Harmelen, 2004). The use of semantic information, perhaps in conjunction with frequency-based search methods, could be expected to produce meaning-based searches that are more directly related to user's meaning-driven queries.

The main obstacle to effective searching is the fact that the content of web pages is not machine-understandable. One technology that represents web content in a form that is easier for machines to understand and process is the Semantic Web (Berners-Lee, Hendler and Lassila, 2001). Semantic Web facilitates semantic searching by using agent technology, ontology, and a number of standard markup languages such as RDF, and OWL to formally model information represented in web resources so that it is accessible to humans and computers working co-operatively, together with the assistance of intelligent network services such as search agents. These markup languages are adopted to add more structural information to the web resources (Davies, 2003, Patel et al., 2003). The knowledge interoperability of the Semantic Web is dependent on ontology, which within computing is a hierarchical relationship between terms within a domain that specifies defined terms and the relationships between those terms. A domain-specific ontology is a tool for modeling resource structures and meanings and that allows software programs to do automated tasks for users, such as searching, customizing and scheduling, which have as one of their points of reference the idea of meaning.

2.3 Semantic Web

The development of the Semantic Web proceeds in layers. The main purpose of this layering approach is that it is easier to achieve consensus in each layer. Figure 2.1 visualizes the Semantic Web architecture of W3C (Semantic Web, 2001). Starting from the bottom of the "layer cake" of the Semantic Web, Extensible Markup Language (XML) (XML, 2004) is being used for self-description documents. XML enables data exchange across the web, but it does not represent any meaning or knowledge embedded in the data. On top of XML, there are Resource Description Framework (RDF) and RDF Schema (RDFS) (RDFS, 2004). RDF is a metadata model for making statements about web resources which is in a form of a subject-predicate-object expression called a triple RDF terminology. Since RDF has an XML-based syntax, it is located on top of the XML layer. RDFS is a language for describing vocabularies in RDF. RDFS is a semantic extension of RDF which is a primitive language for writing ontologies. The upper layer of Semantic Web architecture consists of a logic layer, a proof layer, and a trust layer. The logic layer is used to enhance the ontology language for writing application-specific knowledge. The proof layer executes the rules and evaluates together with the Trust layer mechanism for applications, allowing a decision to be made as to whether to trust the given proof or not.

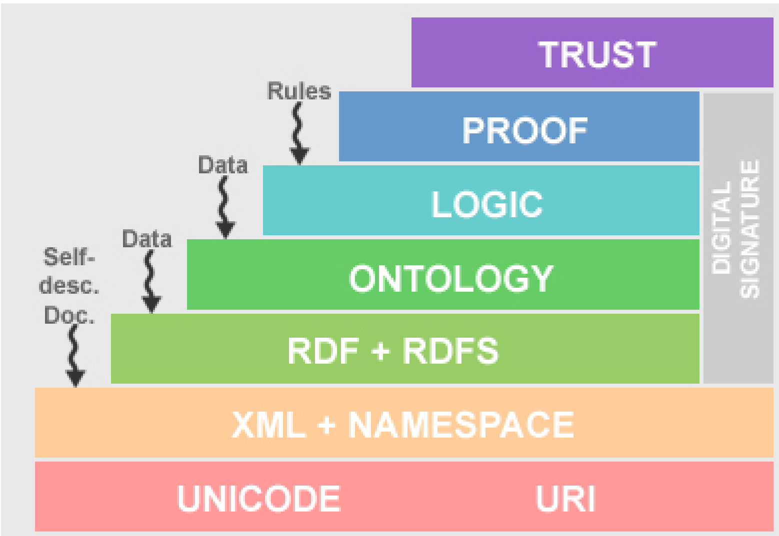


Figure 2.1. Semantic Web architecture (Semantic Web, 2001).

In the following sub-section, a number of layers of the Semantic Web architecture are described in detail. Section 2.3.1 provides more information about XML. Section 2.3.2 provides detail about RDF and RDFS. Section 2.3.3 discusses the idea of a Web Ontology Language.

2.3.1 Extensible Markup Language (XML)

Hypertext Markup Language (HTML) is derived from Standard Generalized Markup Language (SGML). It is one of the languages used for creating web pages. The main use of HTML is to format documents and display information. However, HTML does not contain any structural information. This leads to problems such as high recall and low precision in search results (For more details, please refer to Section 2.2).

XML is a language that allows users to define tags. In HTML, all the tags are pre-defined and users cannot make any changes or make new definitions. The main aim of XML is to extend the markup which ensures uniform data exchange format between applications and supports machine processing of information. During the data communication between applications, application developers are required to have a consensus on the vocabularies (tags) used in XML. Otherwise, there will be problems in communications and collaboration between applications. However, under such an approach, the semantics of XML documents is only accessible to the people who defined it. Machines have no way to understand the meaning of the data. Figure 2.2 shows an example of XML document. This example shows you how to describe a book by using the XML.

XML is useful for data exchange between applications if the involved parties have already defined what the data is during communication. If a new communication partner is involved in the communication, then the model and the mapping must be reengineered (Jannink et al., 1999, McGuinness et al. 2000). This is mainly because XML only structures the document. It does not provide any semantic information about the document (Decker et al., 2000). Ultimately, while XML may be suitable for communication and collaboration in a small community where there are high levels of shared knowledge, it is not appropriate for global communications involving diverse discourse communities.

```
<?xml version = "1.0"?>
<book isbn = "12312312">
    <title>Thomson&apos;s A Guide to Oracle 8</title>
    <author>
        <firstName>Morrison</firstName>
        <lastName>Jolinel</lastName>
    </author>
    <chapters>
        <preface num = "" pages = "3">A Guide to Oracle 8</preface>
        <chapter num = "1" pages = "22">Introduction to Client/Server Database</chapter>
        <chapter num = "2" pages = "27">Creat and Modify Tables</chapter>
    </chapters>
</book>
```

Figure 2.2. Sample XML document

2.3.2 Resource Description Framework (RDF) and RDF Schema

RDF has been proposed as one possible way to solve the problem that XML cannot achieve to support data semantics. RDF is a language, based on XML, for representing information about resources in the World Wide Web. RDF is a form of subject-predicate-object statement which is called a triple statement. The triple statement is commonly written as *(s, p, o)* where the subject s has an attribute *p* with value *o*. For example, there is statement

"Toby is the creator of the webpage http://www.comp.polyu.edu.hk/~cshwlam".

In triple form, then this statement becomes

("Toby", http://www.example.org/creator, http://www.comp.polyu.edu.hk/~cshwlam)

The property "creator" is identified by the URL and the other value is a string. Figure 2.3 shows a graphical representation of this triple. Figure 2.4 shows the RDF representation of this statement.

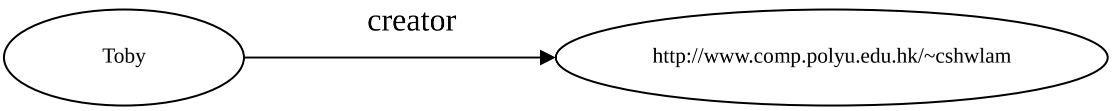


Figure 2.3. Graph representation of the example statement.

```
<?xml version = "1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:example="http://www.example.org/my-rdf-ns">
    <rdf:Description rdf:about="http://www.comp.polyu.edu.hk/~cshwlam">
        <example:creator>
        Toby
        </example:creator>
    </rdf:Description>
</rdf:RDF>
```

Figure 2.4. RDF representation of the example statement

RDF is a framework for expressing and exchanging information between applications without loss of meaning. Other than representing information about web resources, RDF could also be used to represent information about resources that cannot be directly retrieved from the Web. By using RDF, applications designers can develop using different kinds of RDF parsers and processing tools. In addition, the information in an application can not only be used by the user, but also can be used by different applications. This RDF information is distributed across the web. In addition, RDF allows different people to further extend the information in it, allowing knowledge reuse and knowledge sharing. However, RDF is limited as it describes resources by using named properties and values only. RDF Schema (RDFS) is a semantic extension of RDF. RDFS can describe the relationships between resources and groups of related resources such as classes, subclasses, domains and ranges. Since RDFS still lacks some important primitives, it is necessary to have another layer on top of RDF/RDFS.

2.3.3 Web Ontology Language (OWL)

RDF and RDFS have a limited expressive power. RDF is limited to binary predicates and RDFS is limited to a subclass hierarchy and property hierarchy. One proposed solution to this is the Web Ontology Language (OWL). OWL extends RDF/RDFS to make it easier to express semantics.

Before the W3C Web Ontology Working Group defined OWL, a number of researchers joined together to define a markup language called DAML+OIL (Broekstra et al., 2001, Fensel et al, 2001, Patel-Schneider, Horrocks, and van Harmelen, 2002). The DAML+OIL project proposed a standardized and broadly accepted ontology language for the Semantic Web. W3C found that it was necessary to propose a language to further extend the expressivity of RDF and RDFS. This led to DAML+OIL being superseded by OWL.

OWL allows resources to be described in a machine-accessible way. OWL is built upon RDF and RDFS. In OWL, instances are defined by using RDF descriptions and most RDFS modeling primitives are reserved in OWL. Compared with XML, RDF and RDFS, OWL is better for representing machine interpretable contents on to the Web. In addition, OWL supports machine reasoning by using predicate logic and description logic (Baader et al., 2002, Horrocks and Sattler, 2001). There are three different sublanguages in OWL: OWL Lite, OWL DL and OWL Full. OWL Full is the superset of OWL Lite and OWL DL. OWL DL is based on description logic and its subset OWL Lite is based on the less expressive logic. OWL Lite supports users' basic need for hierarchy classification and simple constraints. OWL DL maximizes expressiveness while retaining computational completeness. OWL Full maximizes expressiveness but it may increase the computational complexity. Figure 2.5 shows an example of OWL which was used to model the Hong Kong Tourism.

```
…
  <owl:Class rdf:ID="Custom_Tailor">
    <rdfs:subClassOf rdf:resource="#Shop_Type"/>
  </owl:Class>
  <owl:Class rdf:ID="Useful_Telephone_Number">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="General_Information"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="Handbags_Shoes_and_Leather_Goods">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Shopping"/>
    </rdfs:subClassOf>
  </owl:Class>
    <owl:Class rdf:ID="Hotel">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Accommodation"/>
    </rdfs:subClassOf>
  </owl:Class>
  </owl:ObjectProperty>…
```

Figure 2.5. Example of OWL

2.4 Problems in current ontology markup language

Ontology is defined as a specification of concepts and the relationship between those concepts. This knowledge would be shared and reused in the community. However, the current ontology markup languages such as RDF and OWL are limited since they are based on the hard-constraint logic. The knowledge representation in current ontology is either true or false and there is nothing in between. Thus, the current ontology markup language is not able to represent uncertain information.

There are many different types of uncertain information in real-life. For example, people may describe information vaguely, saying for example that the weather is very hot, that Peter is very tall, that this box is very heavy. In search functions in web application, users are interested in searching using an inexact or approximate approach such as classification by genre or by rating the similarity of the search results. The current ontology markup language cannot support reasoning if only partial information can be obtained. To solve this problem, one solution is to further extend representations to handle uncertain data. There are a number of ways to solve this problem, for example, by incorporating fuzzy logic or probabilistic theory into the ontology.

2.5 Related work

This thesis proposes an extension of the current ontology representation which supports uncertain information modeling. The extension is called Fuzzy Ontology Map (FOM) which is based on the integration of fuzzy theory and graph theory. The idea of FOM was inspired by Fuzzy Cognitive Map (Kosko, 1993, Dickerson and Kosko, 1997). Fuzzy Cognitive Map (FCM) was proposed by Professor Bart Kosko. The real world can be modeled by using FCM. We can use FCM to map the consequences of a course of action in a given situation. However, FCM did not adopt any ontology for concept representation.

In the proposed Fuzzy Ontology Map (FOM), RDF/OWL was adopted for ontology representation. Instead of predicting how events interact and play out, the proposed FOM used for modeling uncertain information and generating relationship between different concepts. The main purpose of the FOM is to let the computers (software agents) for getting more meaningful information.

There has been a great deal of work on supporting uncertainty in relational models (Dey and Sarkar, 1996, Fuhur and Rolleke, 1997, Lakshmanan, et al., 1997). However, there has been little work on supporting uncertainty in ontology. Lee et al. (Lee, Jian and Huang, 2005) proposed an algorithm to create fuzzy ontology and applied it to news summarization. This work is based on their previous work on ontology-based fuzzy event extraction agents for Chinese news summarization (Lee, Chen and Jian, 2003). The domain ontology extended to a fuzzy ontology by including a set of membership values. The created fuzzy ontology is an extension of the domain ontology which is suitable for solving uncertain reasoning problems. However, there are still some irrelevant sentences in the summarized passage. Lee et al. (Lee, Jian and Hsieh, 2005) also proposed an ontology-based genetic fuzzy agent (OFGA) for Meeting Scheduling System (MSS). The agent uses a set of fuzzy personal ontologies and meeting schedule ontologies to help the meeting host to find a suitable meeting time.

Tho et al. (Tho, et al., 2006) proposed a Fuzzy Ontology Generation Framework (FOGA) for fuzzy ontology generation on uncertainty information. This framework is based on the idea of fuzzy theory (Zadeh, 1975) and Formal Concept Analysis (FCA) (Ganter and Willie, 1999). FCA is a technique for data analysis and knowledge presentation which defines formal contexts to represent relationships between objects and attributes in a domain. By using FOGA, it could construct an ontology from uncertain information automatically. The proposed framework is used to construct Semantic Scholarly Web from a citation database for scientific publication search. The main focus of FOGA is the construction of an ontology from uncertain information. In FOGA, the concept hierarchy is represented by using OWL. Each instance of the ontology includes an element describing about the fuzzy membership value. By using such an approach, it greatly increases the file size for the OWL. In addition, each class must contain a minimum of one instance. As a result, it may waste the resource and the disk space.

Widyantoro and Yen developed a scientific paper search engine similar to that of Tho et al. They developed a web-based search engine, Personalized Abstract Search Services (PASS), for searching the abstracts of research paper (Widuantoro and Yen, 2001). The system constructs the fuzzy ontology for query refinement (Velez et al., 1997) broadening and narrowing the relationships of the user requested term. In the paper, the authors described the algorithms for building the fuzzy ontology from fuzzy narrower and broader terms. But, it is not cleared which markup language they have used in the system.

In Semantic Web, social networks trust research has gained interest in recent years. Friend-Of-A-Friend (FOAF) is one of the most popular representations to describe people's relationships (FOAF, 2005). FOAF is developed based on RDF and OWL. Some research has used FOAF as the basis for the trust in the social network such as FOAF Trust Module[4]. Jennifer and Hendler developed a film recommendation website, FilmTrust, which is based on the FOAF and trust values in social networks (Golbeck and Hendler, 2006). This website would make recommendations based on the user's friend's recommendations. Instead of using FOAF, Mazzieriei proposed an extension of RDF statement which is called a fuzzy RDF statement for the representation of trust metadata (Mazzieriei, 2004). A fuzzy RDF statement is defined as a quadruple statement which extends the RDF triple structure with a fourth element representing the trust value. The trust value is a value between 0 and 1. However, using this approach requires changing the syntax of the RDF.

Other than using fuzzy theory to extend the ontology, Ding and Peng (Ding and Peng, 2004) proposed a probabilistic extension to ontology language OWL that used Bayesian networks (Peral, 1988). The extended OWL supports the representation of uncertain ontology, reasoning, and mapping. Ding and Peng defined a number of OWL classes such as *PriorProbObj*, *CondProbObjectT* and *CondProbObjF* for markup probabilities in OWL files. However, there is a limitation in that the ontology should not have any cycle. Since cycles are not allowed in Bayesian networks, it is necessary to have pre-processing to detect and remove the cycle in the ontology. Fukushige (Fukushige, 2004) took a similar approach, proposing a vocabulary for representing probabilistic knowledge in RDF and a framework for calculating probabilities using RDF and Bayesian networks.

OWL is derived from DAML+OIL, where DAML+OIL is developed based on

[4] FOAF Trust Module, http://trust.mindswap.org/ont/trust.owl

description logic, which is a subset of first-order-logic (Horrocks, 2002). Thus, OWL is also based on description logic. A number of research works have extended description logic with probabilistic logic (Heinsohn, 1994, Jaeger, 1994). Koller et al. developed a probabilistic extension of the description logic, *CLASSIC*, which is based on Bayesian Networks and is called *P-CLASSIC* (Koller, Levy and Pfeffer, 1997). Giugno and Lukasiewicz extended the description logic *SHOQ(D)* to develop a probabilistic description logic *P-SHOQ(D)* (Giugno and Lukasiewicz, 2002). The description logic of DAML+OIL is *SHOQ(D)* (Horrocks and Sattler, 2001). Other than using probabilistic logic, there is also fuzzy extension for the description logic, e.g. Fuzzy OWL (FOWL) (Stoilos et al. 2005, Stoilos et al. 2006). FOWL is an fuzzy extension of OWL. It can capture imprecise and vague knowledge by using a new representation language called Fuzzy OWL's DL language. Stoilos and his researchers have already developed a prototype platform, Fuzzy Reasoning Engine (FiRE), for fuzzy reasoning. A comparison between FOM and some recent developments was shown in Table 2.1.

2.6 Fuzzy set theory

The proposed Fuzzy Ontology Map (FOM) is motivated by Fuzzy Set Theory (Zadeh, 1965). As mentioned in the previous section, real world knowledge is imprecise or vague. It is not practical to model knowledge by using crisp logic which must be either true (1) or false (0). Fuzzy set theory offers one way of dealing with this uncertainty.

Fuzzy set theory, an extension of traditional set theory, is used to account for the vague or imprecisely represented information. Fuzzy set theory uses a membership function μ to allow an item in a set to be any real number between 0 and 1. A fuzzy set F of X is characterized by the membership function $\mu_F(x)$ and the membership values are in the unit interval [0, 1].

$$F = \{x, \mu_F(x)) \mid x \in X\} \quad (1)$$

The value $\mu_F(x)$ shows the degree of membership of *x* relative to *F.* If $\mu_F(x) = 0$, then *x* is not a member of *F*. If $0 < \mu_F(x) < 1$, then it is uncertain that x belongs to *F*. If $\mu_F(x) = 1$, then *x* belongs to *F*. In general, the larger the membership value $\mu_F(x)$, the more the certainty that *x* belongs to *F*. Figure 2.6 illustrates an example of the membership function of a fuzzy set $\mu_F(x)$. In general, the function $\mu_F(x)$ is a bounded function with a label *F*. The label *F* is a term, such as tall and young, which the fuzzy sets refer to. For example, if the label *F* is tall, then the value of *x* is the height. When *X* is continuous, a fuzzy set *F* can be written as

$$F = \int_X \mu_F(x)/x \qquad (2)$$

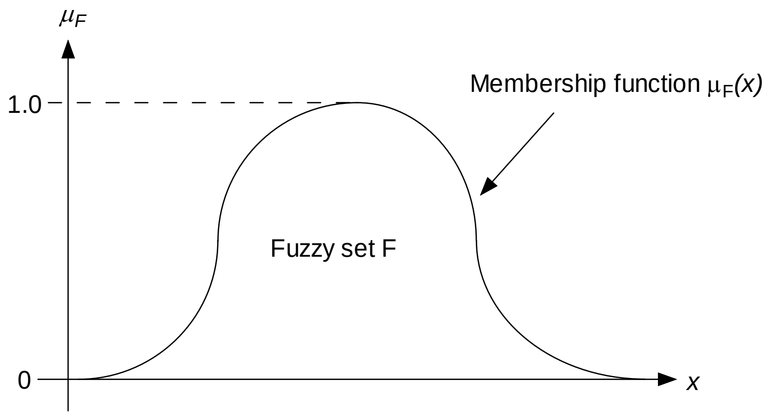


Figure 2.6. Fuzzy membership function for a fuzzy set *F*

In some situations, the membership function is discrete and it is not possible to draw a fuzzy membership figure like Figure 2.6. For example, we might say that Apple is similar to Orange with a degree of confidence in the similarity of 0.5. But since no scale is provided for this degree of similarity, it is not possible to draw a fuzzy membership figure. When *X* is discrete, a fuzzy set *F* can be written as

$$F = \sum \mu_F(x_i)/x_i \qquad (3)$$

or

$$F = \mu_F(x_1)/x_1 + \mu_F(x_2)/x_2 + ... + \mu_F(x_N)/x_N \qquad (4)$$

Fuzzy logic is a superset of crisp logic. Fuzzy logic (Zadeh, 1975) is based on fuzzy set theory which is developed for approximate representation and reasoning with imprecise information. Fuzzy logic has been applied to several areas such as information retrieval and expert systems. However, it has not yet been fully developed for use in Semantic Web. FOM is proposed to extend the ability of the current Semantic Web to deal with real world information. Current methods for ontological representations such as RDF and OWL can only match data exactly, for example, in listing all the instance information in the Hotel class (see figure 2.4). However, using FOM, it is possible to do a fuzzy search such as listing all the accommodation information which is similar to the class Hotel.

2.7 Conclusion

The background information about Semantic Web and the problem of the current ontology markup language are presented in this Chapter. Crisp ontology is limited, in as much as it is not capable of representing approximate and uncertain information effectively. An extension of the crisp ontology, Fuzzy Ontology Map (FOM), is proposed to solve such ‘hard-constraint’ problems. The algorithm and the implementation details about FOM are shown in Chapter 3.

Table 2.1. Comparison between recent developments

	Fuzzy Ontology Map (FOM)	**Fuzzy Cognitive Map (FCM)**	**Ontology-based Genetic Fuzzy Agent (OGFA)**	**Fuzzy Ontology Generation Framework (FOGA)**	**Fuzzy OWL (FOWL)**
Main purpose	■ Extends the current crisp ontology ■ Generate relationship between different concepts ■ Model uncertain information	■ Predict how events interact and play out	■ Help to meeting host to find a suitable timeslot for the meeting ■ The agent has learning ability (preference)	■ Model uncertain information	■ Extends the current crisp ontology ■ Model uncertain information
Adopted Technique	■ Graph theory ■ Fuzzy logic ■ Ontology ■ Agent	■ Graph theory ■ Fuzzy logic	■ Fuzzy logic ■ Genetic algorithm ■ Ontology ■ Agent	■ Formal Concept Analysis (FCA) ■ Fuzzy logic	■ Description Logic ■ Fuzzy Logic
Required to Change the Ontology Syntax	■ No	■ Nil	■ No	■ No	■ No
Fuzzy Information Manipulation	■ Store in an XML document	■ Nil	■ Store in the ontology document directly	■ Store in the ontology document directly	■ Stored in a new language called Fuzzy OWL's DL Language (RDF/XML)

Chapter 3. Fuzzy Ontology Map

3.1. Introduction

As mentioned before, Semantic Web is based on a number of technologies such as agent technology and ontologies. An ontology is a representation of domain knowledge. In general, an ontology is a hierarchical relationship between terms within a domain that specifies defined terms and the relationships between those terms. W3C has a number of markup language recommendations for modeling ontologies, such as Resources Description Framework (RDF) and Web Ontology Language (OWL). These markup languages are mainly based on bivalent logic, facts are represented as either true or false. Therefore, a subject in RDF/OWL is either a member of a class or not.

Although the main aim of Semantic Web is to solve problems of information retrieval, the proposed markup languages are currently not entirely practical for real-life applications. The Semantic Web uses ontologies with hard constraints (either true or false and there is nothing in between) to describe and manipulate knowledge. The expression of real-world knowledge, however, is often uncertain and imprecise. For example, we may say that the weather is very hot rather than saying that the temperature is 37.81 degrees. Current markup languages for modeling ontology cannot process these soft semantics.

In the current ontology representation, it is not possible to model uncertain information.

In this research work, Fuzzy Ontology Map (FOM) is proposed for extending the capability of the current ontology to model real world vague information. By using FOM with the ontology, it is possible to develop real-world applications and systems that can deal with imprecise or vague information.

3.1.2. Organization of the Chapter

The rest of this chapter describes the Fuzzy Ontology Map (FOM) in detail. Section 3.2 shows the details of the Fuzzy Ontology Map. Section 3.3 describes the implementation details and experimental results. Section 3.4 provides a brief summary of this chapter.

3.2. Fuzzy Ontology Map (FOM)

Fuzzy Ontology Map (FOM) is a connection matrix which collects the fuzzy membership values between each class in an ontology. The ontology is a hierarchal relationship between concepts within a domain which is modeled by using certain markup language such as Resource Description Language (RDF) and Ontology Web Language (OWL). The constructed ontology can be viewed as a graph. FOM is developed based on the ontology graph and fuzzy logic. Section 3.2.1 shows the theoretical details of FOM. Section 3.2.2 shows the how to construct a FOM.

3.2.1. Theoretical Analysis of Fuzzy Ontology Map (FOM)

This section describes the theoretical detail of how to develop a FOM. For ease of explanation, RDF is adopted to illustrate how to construct a FOM. Since OWL uses the RDF mechanisms for data values, it is also capable of developing a FOM from OWL.

Definition 1. *An ontology* Ω *is defined as a finite set of triples* $<s, p, o>$ *where s is a subject name, p is a predicate name and o is an object name (which could be another subject* *name). In general, the predicate p is defined as "a kind of" relation between the subject s and object o.*

An ontology document (RDF/OWL) is a collection of triple statements. The directed and labeled graph is one of the representations for displaying ontology data. A graph is a collection of vertices connected by edges. In a directed and labeled graph, the edges point from one vertex to another and each edge has a label.

Definition 2. *Suppose* V *be a finite set of vertices,* $E \subseteq V \times V$ *be a set of edges and* $l : E \rightarrow L$ *be a mapping from edges to a set* L *of strings called labels. The triple* $\Omega = (V, E, l)$ *is a directed and labeled graph.* $V = S \cup O$ *where S is the set of subject names s and O is the set of object names o.*

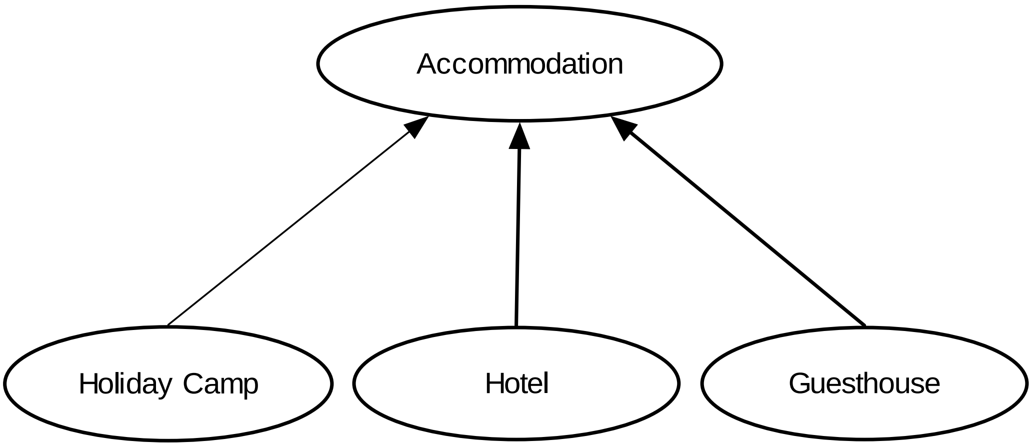


Figure 3.1. Example of an ontology graph

In an ontology graph, a vertex represents a subject or literal value and an arrow represents a predicate. Figure 3.1 illustrates an example of an ontology graph. Since the current ontology markup language is developed based on crisp logic, in order to support modeling vague or imprecise real world knowledge, a fuzzy extension of ontology is proposed.

Definition 3. *Suppose* V *be a finite set of vertices,* $E \subseteq V \times V$ *be a set of edges. The edges E are assigned by a continuous fuzzy value and a label F.* $\mu_F : E \to [0,1]$ *is the continuous membership value.* $l : E \to L$ *is a mapping from edges to a set* L *of strings called labels. The fuzzy ontology graph is a quadruple* $\Omega' = (V, E, l, \mu_F)$.

The fuzzy set F would be used to describe different kinds of relationship such as similarity and preference. The fuzzy relationship would be defined by the creator of the ontology during the ontology modeling stage or defined by the users who apply the ontology. Figure 3.2 illustrates the graph of fuzzy accommodation ontology. In Figure 3.2, the fuzzy membership value describes the similarity between two concepts. For instance, A2 Hotel is similar to A1 Accommodation with fuzzy membership value 0.8. A3 Guesthouse is similar to A1 Accommodation with fuzzy membership value 0.6. This means that A2 Hotel would be more similar to A1 Accommodation than A3 Guesthouse. This example will be used throughout this chapter. Figure 3.3 shows the simplified version of this fuzzy ontology.

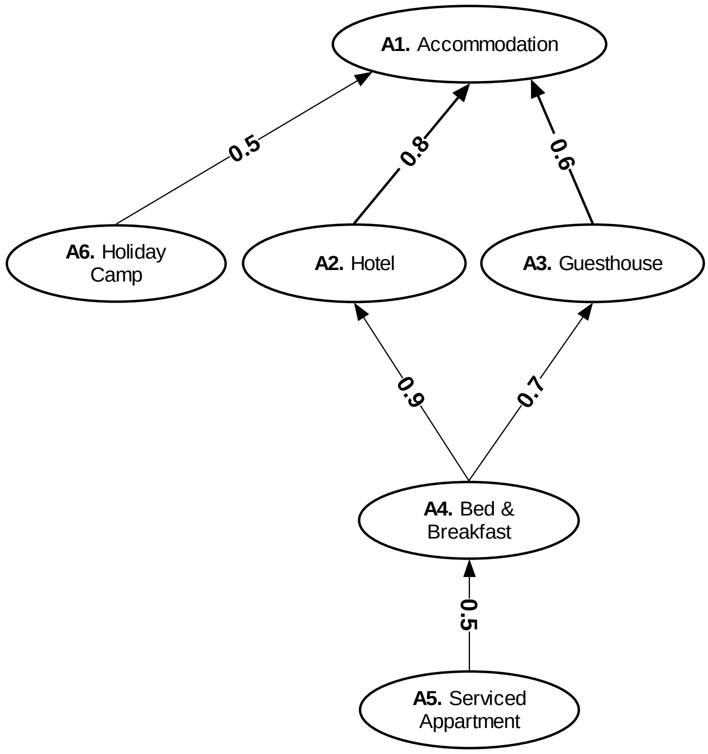


Figure 3.2. Graph of fuzzy accommodation ontology

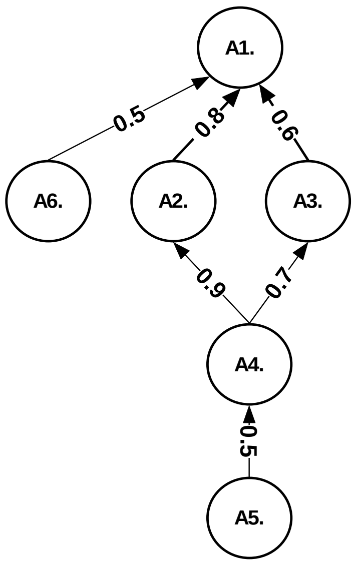


Figure 3.3. A simplified graph of fuzzy accommodation ontology

A fuzzy ontology map is a connection matrix which illustrates the links between classes. The fuzzy ontology can be used to create a fuzzy ontology map FOM . Figure 3.4 shows the fuzzy ontology map which is created from the fuzzy accommodation ontology.

		A1	A2	A3	A4	A5	A6
	A1	-	0	0	0	0	0
	A2	0.8	-	0	0	0	0
	A3	0.6	0	-	0	0	0
$\boldsymbol{FOM_{base}}$=	A4	0	0.9	0.7	-	0	0
	A5	0	0	0	0.5	-	0
	A6	0.5	0	0	0	0	-

Figure 3.4. Base form of fuzzy ontology map FOM_{base} of accommodation ontology

In the FOM, the *i*-th row lists the membership value of the edges e_{ik} from the class A_{i}. The *i*-th column lists the fuzzy value of the edges e_{ki} directed into A_{i}, In this example, the membership value represents the similarity between classes. In general, the matrix shown in Figure 3.4 would be treated as the base form of fuzzy ontology map FOM_{base}.

In this example, the fuzzy set is similarity. A3 (Guesthouse) is similar to A1 (Accommodation) with membership value 0.6 ($e_{A3,A1} = 0.6$) and A4 (Bed & Breakfast) is similar A3 (Guesthouse) with membership value 0.7 ($e_{A4,A3} = 0.7$). In some sense, A4 (Bed & Breakfast) is similar to A1 (Accommodation) indirectly.

Definition 4. *Suppose A, B, C are the classes in FOM. There is an edge* e_{BA} *connected from class B to class A with membership value* μ_{BA} *and an edge* e_{CB} *connected from class C to class B with membership value* μ_{CB}. *As a result, class A is reachable from class C by class B. If there is no direct edge from class C to class A, then we would derive a virtual edge* e_{CA} *with membership value* μ_{CA}, *i.e.*

$$\mu_{CA} = \mu_{CB} \times \mu_{BA} \tag{5}$$

In Definition 4, the real edges are the edges with membership values greater than 0 in the base form of FOM $FOM_{base.}$ Any created edge that does not previously exist in the base form of FOM FOM_{base} is called a virtual edge. The reachable classes mean that one class can be reached by another class by using the real edges. The main purpose of Definition 4 is to create edges for the reachable classes. Thus, by using Definition 4, it could make relationship for the reachable classes. Algorithm 1 shows how to create virtual edges for reachable classes in a fuzzy ontology map.

Algorithm 1: Create_virtual_edge_for_reachable_classes(FOM)

Input: Fuzzy Ontology Map *FOM*

Output: Fuzzy Ontology Map with virtual edge FOM_V

```
C = classes in FOM;
FOM_V = empty matrix with the same size as FOM;
    for each class I of C do
        for each class J of C do
            for each class K of C do
                if (I is not the same as J AND J is not the same as K) then
                    if (edge e_kj and edge e_ji exists in FOM) then
                        if (FOM_V [i, k] exists)
                            FOM_V [i, k] = COMP(FOM_V [i, k], μ_kj *μ_ji)
                        else
                            FOM_V [i, k] =μ_kj *μ_ji;;
                        endif
                    endif
```

```
                    endif
               endfor
          endfor
     endfor
FOM_V = FOM_V + FOM;
return FOM_V;
```

* *COMP(x, y)* is a comparative function.

During the creation of virtual edges, there is a chance that more than one virtual edge exists between two classes. To handle this, Algorithm 1 contains a comparative function, called *COMP(x, y),* for selecting a suitable virtual edge. We can choose one of the following measures:

$$COMP(x,y) = \max(x,y) = \begin{cases} x & if\ (x > y) \\ y & otherwise \end{cases} \tag{6}$$

$$COMP(x,y) = \min(x,y) = \begin{cases} x & if\ (x < y) \\ y & otherwise \end{cases} \tag{7}$$

When more than one virtual edge, each virtual edge should have membership value which can be used for comparison. If the fuzzy relationship between classes is tight and restricted in the ontology, the relationship between classes should not be created easily, e.g. preference, then it is recommended that min(x, y) be used as the comparative function. The virtual edge with smaller membership value is created by using min(x,y). Hence, it can retain the relationship between classes of the ontology in tight condition. If the fuzzy relationship between classes is flexible, i.e. the relationship between classes could be created easily, then it is recommended that max(x,y) be used as the comparative function. The choice between Equations 6 and 7 depends on the user. Figures 3.5a and 3.5b show the FOM with virtual edges for reachable classes between A1 and A4 by using max(x,y) and min(x,y) as the comparative function respectively.

To create virtual edges for all reachable classes in a fuzzy ontology map, Algorithm 1 is processed recursively. Algorithm 2 shows how to create virtual edges for all reachable classes. The algorithm tries to create virtual edges for all reachable classes until no virtual edge is created. Figures 3.6a and 3.6b show the FOM with all virtual edges for reachable classes by using max(x,y) and min(x,y) as the comparative function respectively.

Algorithm 2: Create_ virtual_edge_for_all_reachable_classes(*FOM*)

Input: Fuzzy Ontology Map *FOM*

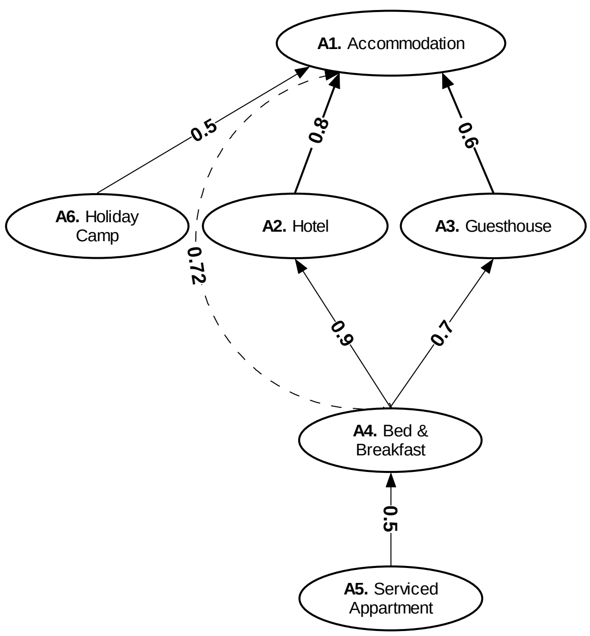
Output: Fuzzy Ontology Map with virtual edge FOM_V

Step1: FOMv= FOM

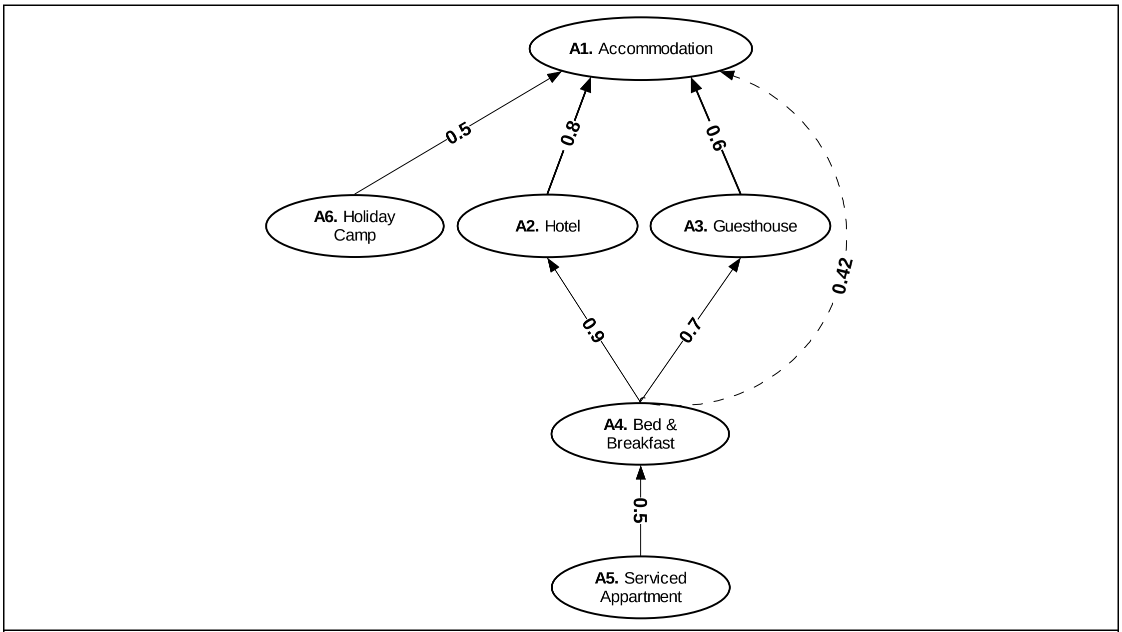
Step2: FOMv=Create_virtual_edge_for_reachable_classes(*FOMv*)

Step3: if virtual edge(s) is created in step 2, return to step 2

Step4: return *FOMv*



a.



b.

Figure 3.5. (a) FOM with virtual edges for reachable classes A1 and A4 by using max(x,y) as the comparative function (b) FOM with virtual edges for reachable classes A1 and A4 by using min(x,y) as the comparative function

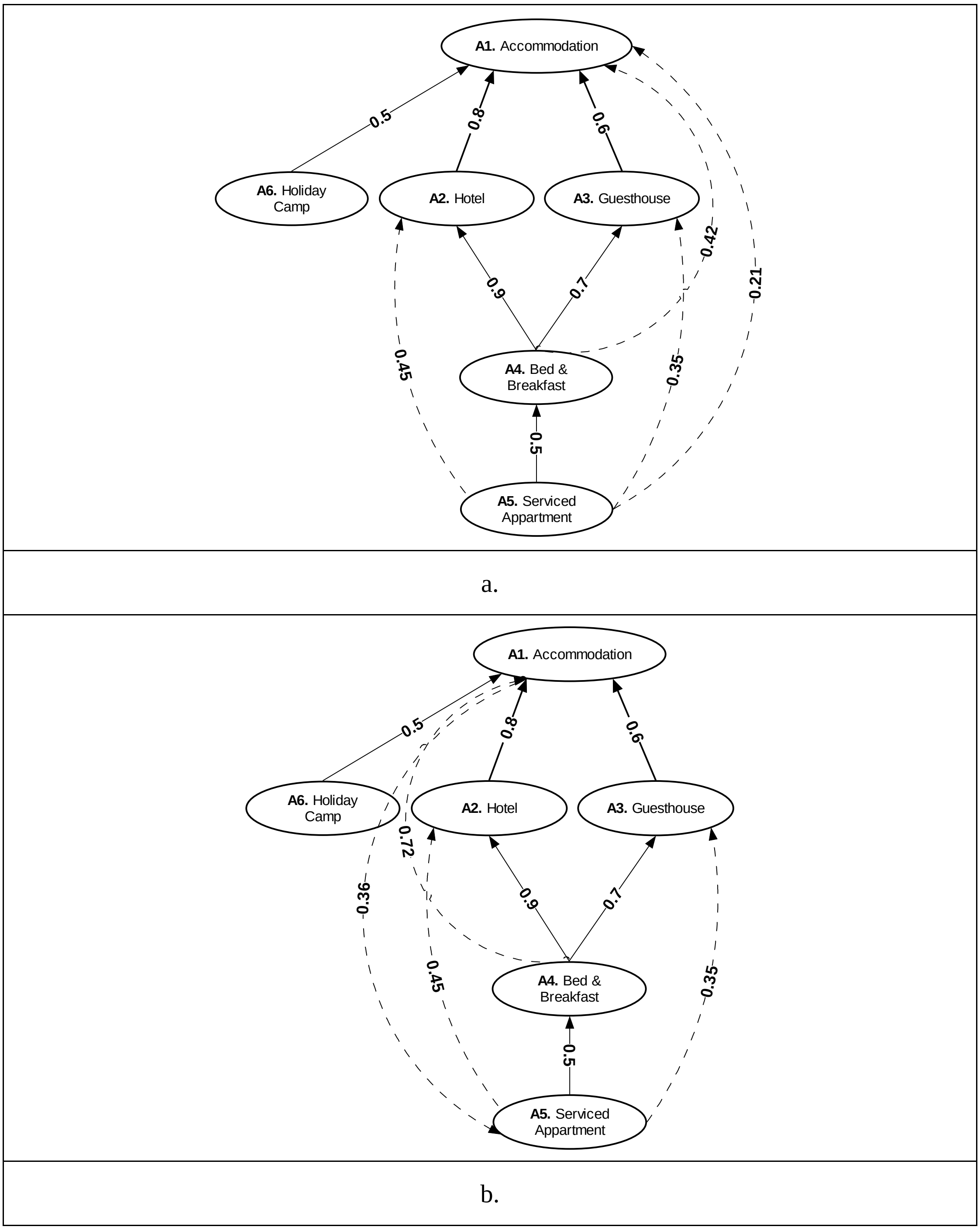


Figure 3.6. (a) FOM with all virtual edges for reachable classes by using min(x,y) as the comparative function (b) FOM with all virtual edges for reachable classes by using max(x,y) as the comparative function

We defined that there are two types of fuzzy set: symmetric and asymmetric. A symmetric fuzzy set is the one in which the fuzzy relation between two classes is non-directional such as a level of similarity. The degree of similarity is symmetric as there is no difference between 'class A similar to class B' and 'class B similar to class A' (non-directional). It means that there is no difference in the two (opposite) direction (A->B = B->A). An asymmetric fuzzy set is the one in which the fuzzy relation between two classes is directional such as level of preference. Suppose class A is preferred to class B, then there is a difference in the opposite direction. It means that if class A is preferred to class B (A->B), then it is not the same in opposite direction (B->A). In the fuzzy accommodation ontology example, the fuzzy set is similarity. It can be stated that the fuzzy set is *symmetric*.

In the accommodation example, there is an edge from A2 (Hotel) to A1 (Accommodation) with a membership value 0.8 and an edge from A3 (Guesthouse) to A1 (Accommodation) with a membership value 0.6. In some sense, A2 is similar to A3 in some level. This means that if there are two classes (e.g. A and B) which point to the same class (e.g. C), then a virtual edge would be created between two classes by using definition 5. Definition 5 is applied when the fuzzy set is symmetric. If the fuzzy set is asymmetric, then definition 6 is applied.

Definition 5. *Suppose A, B, C are the classes in FOM and the fuzzy set is symmetric. There is an edge* e_{BA} *connected from class B to class A with the membership value* μ_{BA} *and an edge* e_{CA} *connected from class C to class A with the membership value* μ_{CA}*. If there is no edge between class B and class C (either from class B to class C or from class C to class B), then two virtual edges* e_{BC} *and* e_{CB} *would be derived with the membership value* μ*, i.e.*

$$\mu = sim(\mu_{BA}, \mu_{CA}) = \frac{\min(\mu_{BA}, \mu_{CA})}{\max(\mu_{BA}, \mu_{CA})} \quad (8)$$

The unreachable class means that one class cannot be reached by another class by using the real and virtual edges. The main purpose of Definition 5 is to create relationship for unreachable classes. Definition 5 can only be applied if the fuzzy set is symmetric. If the fuzzy set is asymmetric, then Definition 6 should be adopted. When the fuzzy set is symmetric, if two unreachable classes are pointed to the same class, in some sense, these two unreachable classes have a relationship with each other. Equation 8 is proposed to calculate the fuzzy value between the two unreachable classes by ratio.

Definition 6. *Suppose A, B, C are the classes in FOM and the fuzzy set is asymmetric. There is an edge* e_{BA} *connected from class B to class A with the membership value* μ_{BA} *and an edge* e_{CA} *connected from class C to class A with the membership value* μ_{CA}*. If there is no edge between class B and class C (either from class B to class C or from class C to class B), then two virtual edges* e_{BC} *and* e_{CB} *would be derived with the membership value* μ_{BC} *and* μ_{CB} *by:*

$$\mu_{BC} = asymm(\mu_{BA}, \mu_{CA}) = \frac{\dfrac{|\mu_{BA} - \mu_{CA}|}{\mu_{CA}}}{\max(\dfrac{|\mu_{BA} - \mu_{CA}|}{\mu_{CA}}, \dfrac{|\mu_{CA} - \mu_{CBA}|}{\mu_{BA}})} \quad (9)$$

$$\mu_{CB} = asymm(\mu_{CA}, \mu_{BA}) = \frac{\dfrac{|\mu_{CA} - \mu_{BA}|}{\mu_{BA}}}{\max(\dfrac{|\mu_{BA} - \mu_{CA}|}{\mu_{CA}}, \dfrac{|\mu_{CA} - \mu_{BA}|}{\mu_{BA}})} \quad (10)$$

where *max(a,b)* is a function to select the maximum value between value *a* and value *b*.

Similar to Definition 5, the main purpose of Definition 6 is to create relationship for unreachable classes. However, Definition 6 can only be adopted if the fuzzy set is asymmetric. Since the fuzzy set is asymmetric, the membership values should be calculated separately for both directions. For example, the fuzzy set is preference. If the membership value is 0.1 from class B to class A, and the membership value is 0.9 from class C to class A. First, we calculate the absolute difference between two membership values and then divide by the other membership value. As the fuzzy set is asymmetric, to reserve the directional information, the absolute difference should divided by the other membership value, i.e. $\frac{|\mu_{BA} - \mu_{CA}|}{\mu_{CA}}$ and

$\frac{|\mu_{CA} - \mu_{BA}|}{\mu_{BA}}$. These values maybe greater than one, hence it is required to normalization (see Equation 9 and 10). Thus, the membership value from class B to class C is 0.11 and the membership value from class C to class B is 1.

Algorithms 3 and 4 are further created for creating virtual edge for unreachable classes for symmetric fuzzy set and asymmetric fuzzy set, respectively. Figures 3.7a and 3.7b shows the FOM with all virtual edges for reachable classes by using max(x,y) and min(x,y) as the comparative function respectively.

Algorithm 3: [Symmetric Fuzzy Set] Create_virtual_edge_for_unreachable_classes(FOM)
Input: Fuzzy Ontology Map with virtual edge FOM_v (reachable classes) **Output:** Fuzzy Ontology Map with virtual edge FOM_w (reachable and unreachable classes)

C = classes in FOM_v;
FOM_w = empty matrix with the same size as FOM_v;
    for each class I of C do
        for each class J of C do
            for each class K of C do
                if (I is not the same as J AND J is not the same as K) then
                    if (edge e_{ji} and edge e_{ki} exists in FOM_v) then
                        if ($FOM_w[j, k]$ exists AND $FOM_w[k, j]$ exists)
                            $FOM_w[j, k] = COMP(FOM_w[j, k], sim(\mu_{ji}, \mu_{ki}))$
                            $FOM_w[k,j] = FOM_w[i, k]$
                        else
                            $FOM_w[j, k] = sim(\mu_{ji}, \mu_{ki});$
                            $FOM_w[k, j] = FOM_w[j, k];$
                        endif

endif

endif

endfor

endfor

endfor

$FOM_w = FOM_V + FOM_{w;}$

return FOM_w;

Algorithm 4:[Asymmetric Fuzzy Set] Create_virtual_edge_for_unreachable_classes(FOM)

Input: Fuzzy Ontology Map with virtual edge FOM_v (reachable classes)

Output: Fuzzy Ontology Map with virtual edge FOM_w (reachable and unreachable classes)

C = classes in FOM_v;

FOM_w = empty matrix with the same size as FOM_v;

for each class I of C do

for each class J of C do

for each class K of C do

if (I is not the same as J AND J is not the same as K) then

if (edge e_{ji} and edge e_{ki} exists in FOM_v) then

if ($FOM_w [j, k]$ exists OR $FOM_w [k, j]$ exists)

$FOM_w [j, k] = COMP(FOM_w [j, k], asymm(\mu_{ji,} \mu_{ki}))$

$FOM_w[k,j] = COMP(FOM_w [j, k], asymm(\mu_{ki,} \mu_{ji});$

else

$FOM_w[j, k] = asymm(\mu_{ji,} \mu_{ki});$

$FOM_w[k,j] = asymm(\mu_{ki,} \mu_{ji});$

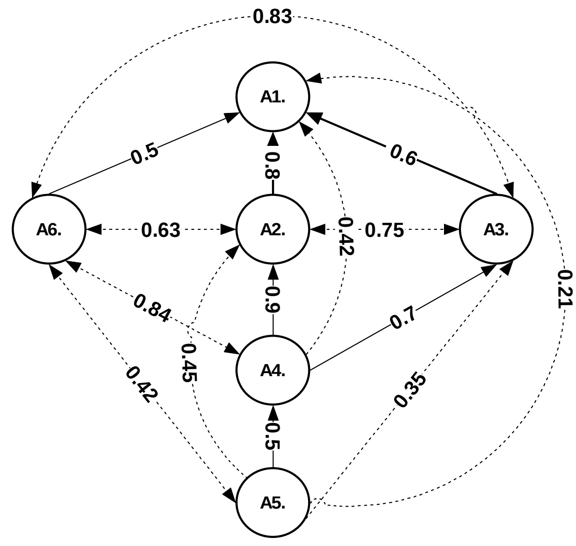
endif

```
                    endif
                endif
            endfor
        endfor
    endfor
```

$FOM_w = FOM_V + FOM_{w;}$

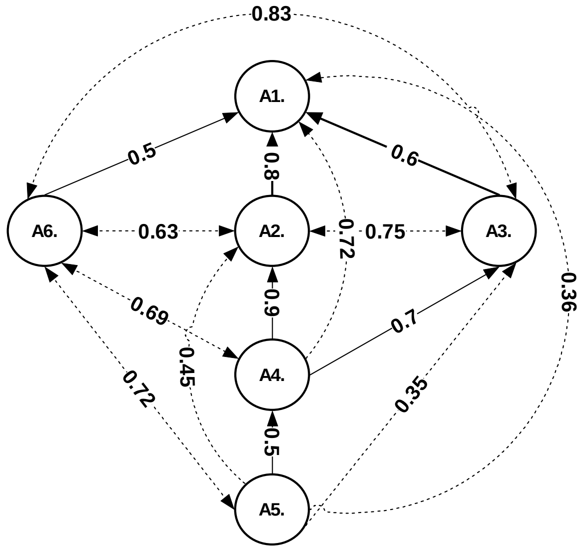
return FOM_w;

In Figure 3.7, we can see that each class in FOM is nearly connected to all classes. If the fuzzy set is symmetric, then it is possible to further create opposite directed edges. Definition 7 shows how to create a virtual edge if the fuzzy set is symmetric. However, if the fuzzy set is asymmetric, it is impossible to create opposite directed edges, for example, if the asymmetric fuzzy set is preference. Suppose class A is preferred to class B with a membership value 0.8, since the class A has a higher preference level than class B, it is not possible to create a virtual edge from class B to class A using the same approach as in definition 7. Algorithm 5 shows how to create the opposite edge for classes which have only one directed edge. Note that algorithm 5 is applied for symmetric fuzzy sets only. Figure 3.8 shows the final fuzzy ontology map of the accommodation ontology.



	A1	A2	A3	A4	A5	A6
A1	-	0	0	0	0	0
A2	0.8	-	0.75	0	0	0.63
A3	0.6	0.75	-	0	0	0.83
A4	0.42	0.9	0.7	-	0	0.84
A5	0.21	0.45	0.35	0.5	-	0.42
A6	0.5	0.63	0.83	0.84	0.42	-

a.



	A1	A2	A3	A4	A5	A6
A1	-	0	0	0	0	0
A2	0.8	-	0.75	0	0	0.63
A3	0.6	0.75	-	0	0	0.83
A4	0.72	0.9	0.7	-	0	0.69
A5	0.36	0.45	0.35	0.5	-	0.72
A6	0.5	0.63	0.83	0.69	0.72	-

b

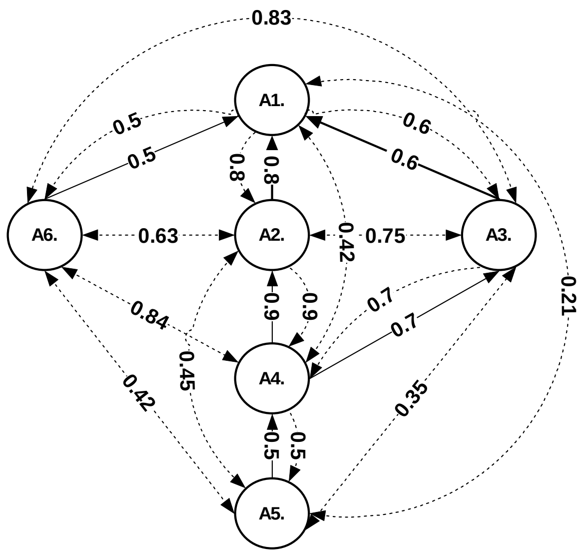
Figure 3.7. (a) FOM with all virtual edges for reachable and unreachable classes by using min(x,y) as the comparative function (b) FOM with all virtual edges for reachable and unreachable classes by using max(x,y) as the comparative function

Definition 7. *Suppose A and B are classes in FOM and the fuzzy set is symmetric. If there is an edge starting from class A and ending at class B with the membership value* μ_{AB} *and there is no edge in the opposite direction, then we can create a virtual edge from class B to class A with the membership value* μ_{symm}, *i.e.*

$$\mu_{symm} = \mu_{AB} \tag{11}$$

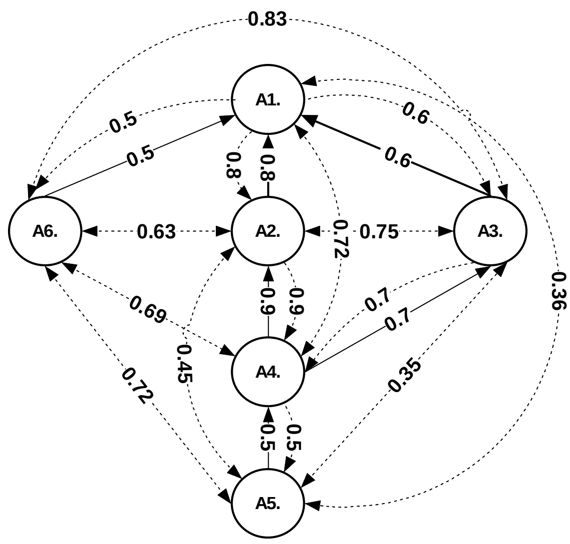
Algorithm 5: Create_virtual_edge_for_classes_only_have_one_directed_edge(FOM)
Input: Fuzzy Ontology Map with virtual edge *FOMw* (reachable and unreachable classes) **Output:** Fuzzy Ontology Map with virtual edge FOM_F

C = classes in FOM_w;
FOM_F = empty matrix with the same size as FOM_w;
    for each class I of C do
        for each class J of C do
            if (I is not the same as J) then
                if (edge e_{ji} exists in FOM_w AND edge e_{ij} not exists in FOM_w) then
                    $FOM_F[i, j] = FOM_w[j, i]$
                endif
            endif
        endfor
    endfor
$FOM_F = FOM_F + FOM_w$;
return FOM_F;



	A1	A2	A3	A4	A5	A6
A1	-	0.8	0.6	0.42	0.21	0.5
A2	0.8	-	0.75	0.9	0.45	0.63
A3	0.6	0.75	-	0.7	0.35	0.83
A4	0.42	0.9	0.7	-	0.5	0.84
A5	0.21	0.45	0.35	0.5	-	0.42
A6	0.5	0.63	0.83	0.84	0.42	-

a.



	A1	A2	A3	A4	A5	A6
A1	-	0.8	0.6	0.72	0.36	0.5
A2	0.8	-	0.75	0.9	0.45	0.63
A3	0.6	0.75	-	0.7	0.35	0.83
A4	0.72	0.9	0.7	-	0.5	0.69
A5	0.36	0.45	0.35	0.5	-	0.72
A6	0.5	0.63	0.83	0.69	0.72	-

b

Figure 3.8. (a) Final FOM which used min(x,y) as the comparative function (b) Final FOM which used max(x,y) as the comparative function

The fuzzy ontology map created by algorithm 4 is called final FOM FOM_F. The final FOM FOM_F would be used to enhance the RDF search function. However, if the final FOM FOM_F is used directly for searching, the computation time would be very great as there are many entries in FOM_F. To further reduce the computation complexity, a threshold T is proposed for removing the virtual edges with small membership values (see Eqn. (12)). The threshold T

would be any numeric value between 0 and 1. Doing this would further reduce the searching size and speed up the searching speed. Eqn. (13) – Eqn. (15) suggests some methods for determining the value *T*. Figure 3.9 shows the FOM_T with threshold *T* = 0.4. Algorithm 6 shows how to filter the virtual edges with threshold *T.* The filtered FOM is a FOM which is initially a fully-connected FOM and is subsequently pruned by using the threshold *T*. The fuzzy ontology map created by Algorithm 6 is called $FOM_{T.}$ Figure 3.9 shows the filtered fuzzy ontology map FOM_T of the accommodation ontology.

$$FOM[i,j] = THRESH(FOM[i,j],T) = \begin{cases} FOM[i,j] & if\,(FOM[i,j] \geq T) \\ 0 & otherwise \end{cases} \tag{12}$$

$$T = \frac{MIN(FOM)}{MAX(FOM)} \tag{13}$$

$$T = \frac{MAX(FOM) - MIN(FOM)}{2} \tag{14}$$

$$T = \frac{\sum_{i=1}^{C}\sum_{j=1}^{C} FOM[i,j]}{C*C} \tag{15}$$

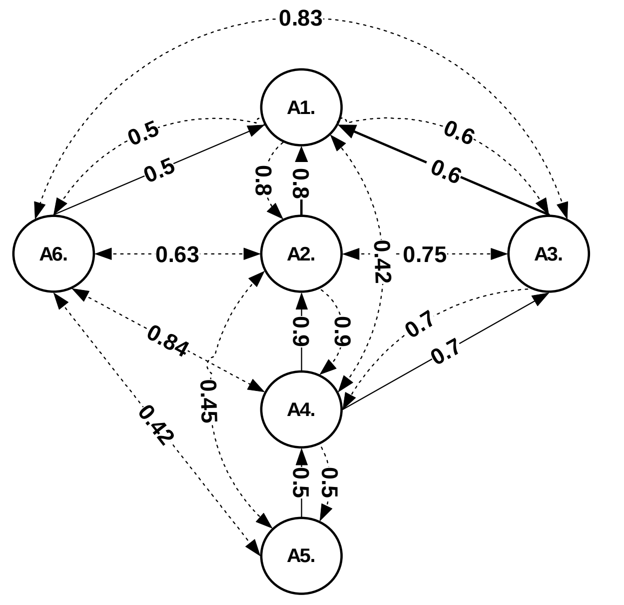
where *MIN* returns the smallest value, which must be greater than zero, in the fuzzy ontology map, *MAX* returns the largest value in the fuzzy ontology map and *C* is the total number of class in the FOM.

Algorithm 6: Filtering_Virtual_Edges (*FOM, T*)
Input: Fuzzy Ontology Map FOM_F **Output:** Fuzzy Ontology Map FOM_T

C = classes in FOM_F;
FOM_T = empty matrix with the same size as FOM_F;
    for each class *I* of *C* do
        for each class *J* of *C* do

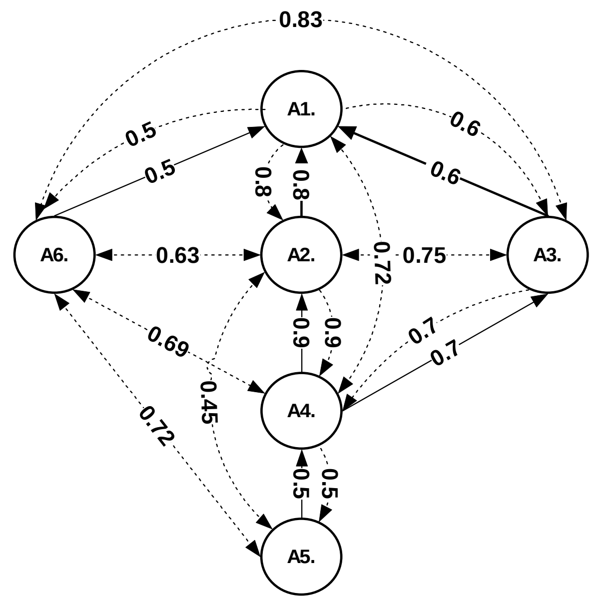
```
                if (I is not the same as J) then
                    if (edge e_ij exists in FOM_F AND edge e_ij is virtual) then
                        FOM_T [i, j] = THRESH(FOM_w [i, j], T)
                    endif
                endif
            endfor
        endfor
return FOM_T;
```

* *THRESH(x, T)* is a threshold function. For more details, see Eqn. (12)



	A1	A2	A3	A4	A5	A6
A1	-	0.8	0.6	0.42	0	0.5
A2	0.8	-	0.75	0.9	0.45	0.63
A3	0.6	0.75	-	0.7	0	0.83
A4	0.42	0.9	0.7	-	0.5	0.84
A5	0	0.45	0	0.5	-	0.42
A6	0.5	0.63	0.83	0.84	0.42	-

a.



	A1	A2	A3	A4	A5	A6
A1	-	0.8	0.6	0.72	0	0.5
A2	0.8	-	0.75	0.9	0.45	0.63
A3	0.6	0.75	-	0.7	0	0.83
A4	0.72	0.9	0.7	-	0.5	0.69
A5	0	0.45	0	0.5	-	0.72
A6	0.5	0.63	0.83	0.69	0.72	-

b.

Figure 3.9. (a) FOM_T which used min(x,y) as the comparative function (b) FOM_T which used max(x,y) as the comparative function

3.3.2. Analysis of Fuzzy Ontology Map

The technical approach to constructing the fuzzy ontology map (FOM) is shown in this section. As shown in the previous sub-section, the fuzzy membership value is one of the main components for creating a FOM. However, in current W3C standards, an RDF statement is a triple statement which contains subject, predicate and object. Although it would be easy to create a fuzzy ontology by adding a fuzzy membership value to the triple statement, it would be necessary to change the syntax of the statement.

After adding a fuzzy membership value, the triple statement would become a quadruple statement composed of a subject, predicate, object and fuzzy membership value. This is a straightforward way to create a fuzzy ontology. However, this also leads to a compatibility problem. Although adding an element to the statement is a clear way to solve the problem, the system developed by quadruple statement may not be applicable in other domains.

Since it is not practical to change the syntax of the RDF, another possible solution would be to include the fuzzy membership information in the class instances directly. Figure 3.10 shows how to include the fuzzy information in the instance. But using such an approach would require each class instance to insert data such as the fuzzy membership value, class relation information (e.g. related to which class, related from which class). This would greatly increase the document file size since there is a lot of duplicated data. Besides, there is a limitation that each class must at least have one instance. Again, it is also not practical to insert the fuzzy membership information directly in the instance of the class in the ontology.

```
…
  <Hotel rdf:ID="Hotel_1">
    <hasFuzzyValue rdf:datatype="http://www.w3.org/2001/XMLSchema#float"
    >0.9</hasFuzzyValue>
    <relatedFrom rdf:resource="#Hotel_1"/>
    <relatedTo rdf:resource="#Bed_n_Breadfast_1"/>
  </Hotel>
  <Guesthouse rdf:ID="Guesthouse_1">
    <relatedTo rdf:resource="#Bed_n_Breadfast_1"/>
    <hasFuzzyValue rdf:datatype="http://www.w3.org/2001/XMLSchema#float"
    >0.8</hasFuzzyValue>
    <relatedFrom rdf:resource="#Guesthouse_1"/>
  </Guesthouse>
…
```

Figure 3.10. Example of ontology which inserts fuzzy membership information to the class instances directly

To ensure applicability and compatibility, the fuzzy ontology map which has the fuzzy information between classes is proposed to create as a XML document. One of the advantages of using XML for modeling fuzzy information is that it is a W3C standard. Another advantage is that XML provides a surface syntax for structured documents which are mainly used for describing data. As a result, in our proposed algorithm, a fuzzy ontology is created using two files: a RDF/OWL document for the domain concept hierarchy and a XML document for the fuzzy information. Figure 3.11 shows an example of a fuzzy XML document. The use of such an approach ensures backward compatibility. If the system did not support fuzzy searching, it would still use the RDF/OWL document for exact match searching. Figure 3.12 shows the Document Type Definition (DTD) which used to validate the syntax of the XML. Note that the generated FOM is also stored as a fuzzy XML document.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<fuzzy>
    <fuzzyset name="similarity" issymm="true" />
    <fuzzyrelation fromclass="hotel">
        <toclass membershipvalue="0.9">accommodation</toclass>
    </fuzzyrelation>
</fuzzy>
```

Figure 3.11. Example of the fuzzy XML

```
<!DOCTYPE fuzzy[
        <!ELEMENT fuzzy (fuzzyset, fuzzyrelation*)>
        <!ELEMENT fuzzyset EMPTY>
        <!ATTLIST fuzzyset
          name CDATA #REQUIRED
          issymm (true | false) "true">
        <!ELEMENT fuzzyrelation (toclass*)>
        <!ATTLIST fuzzyrelation fromclass ID #REQUIRED>
        <!ELEMENT toclass (#PCDATA)>
        <!ATTLIST toclass membershipvalue CDATA #REQUIRED>
]>
```

Figure 3.12. DTD for the fuzzy XML

3.3 Implementation and Experiments

This section shows in detail how the fuzzy ontology map is implemented and describes a number of tests for evaluating the performance of the FOM.

The algorithm of fuzzy ontology map (FOM) was implemented by using JAVA. The input of the program is the fuzzy ontology (RDF/OWL document and fuzzy XML document). The output of the program is the FOM which is also stored as an XML file with the same structure as the fuzzy XML. To evaluate the performance of the proposed algorithm, the algorithm was tested on both synthetically generated and real-life ontologies. All experiments were run on a PC with 1.82 GHz CPU, 768MB memory and Windows XP platform.

3.3.1 Experiments on Synthetic Data

Experiment I: Processing time for creating a FOM

In this experiment, a number of ontology graphs were randomly generated by using a JAVA program. In these synthetic ontology graphs, the number of classes ranges from 10 to 2000 with a step of 10. Furthermore, the fuzzy membership value between connected edges is also randomly generated by using the program. Eqn. (13) – Eqn. (15) were used to determine the threshold T for filtering the virtual edges in the FOM. In this experiment, the fuzzy relationship is assumed to be symmetric. For each ontology graph, the processing time for generating a FOM was recorded.

$$T = \frac{MIN(FOM)}{MAX(FOM)} \tag{13}$$

$$T = \frac{MAX(FOM) - MIN(FOM)}{2} \tag{14}$$

$$T = \frac{\sum_{i=1}^{C}\sum_{j=1}^{C} FOM[i,j]}{C * C} \tag{15}$$

where *MIN* returns the smallest value, which must be greater than zero, in the fuzzy ontology map, *MAX* returns the largest value in the fuzzy ontology map and *C* is the total number of class in the FOM.

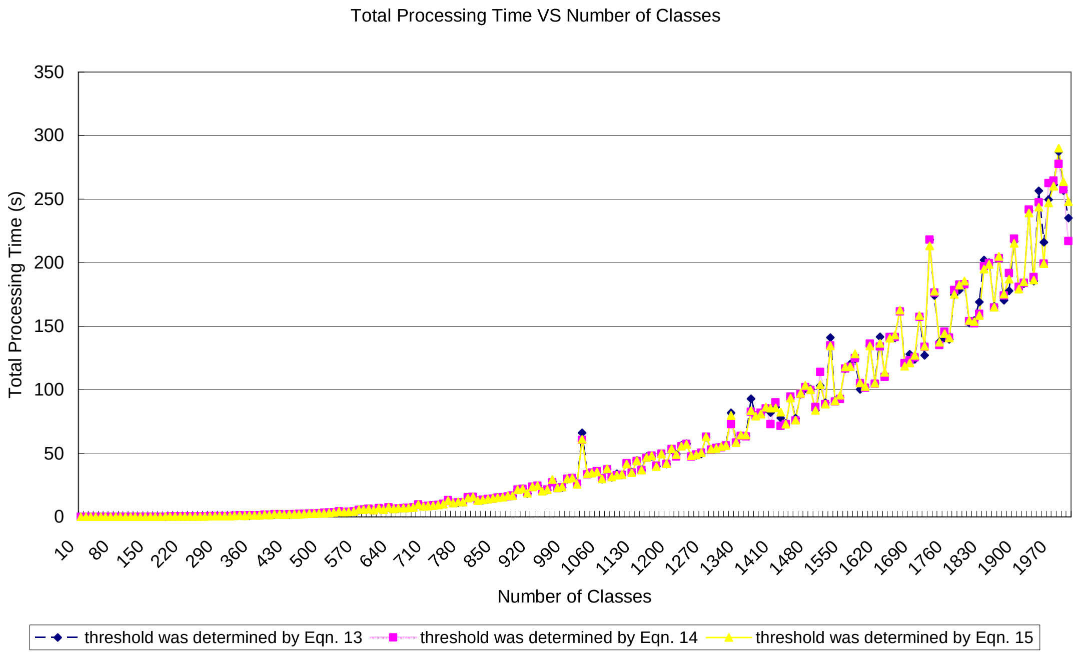


Figure 3.13. Total Processing Time Vs. Number of Classes

Figure 3.13 shows the processing time for creating a FOM. The total processing increases non-linearly with the number of classes in the ontology. The total processing time for different threshold determination methods are nearly the same. The increase in the processing time is mainly due to creating a FOM which depends on the number of classes in the ontology. Thus, the relationship between total processing time and the number is class is exponential. Since the algorithm attempts to create virtual edges for each class, the processing time would be increased if more classes exist in the ontology.

Experiment II: Number of Edges in the filtered FOM

The experimental setup was the same as in Experiment I. This time, the number of edges in the initial and final FOM were recorded. As mentioned in Section 3.2 (P.40), the filtered FOM is a FOM which is initially a fully-connected FOM and is subsequently pruned by using the threshold *T*. Figure 3.14 illustrates the number of edges in the FOM as a function of the number of classes. Note that Eqn. (13) – Eqn. (15) were adopted to determine the threshold *T* for filtering the virtual edges. Compared with other threshold determination methods, if Eqn. (13) was adopted, the FOM has the greatest number of edges. If Eqn. (14) was adopted for determining the threshold *T*, the FOM has the smallest number of edges.

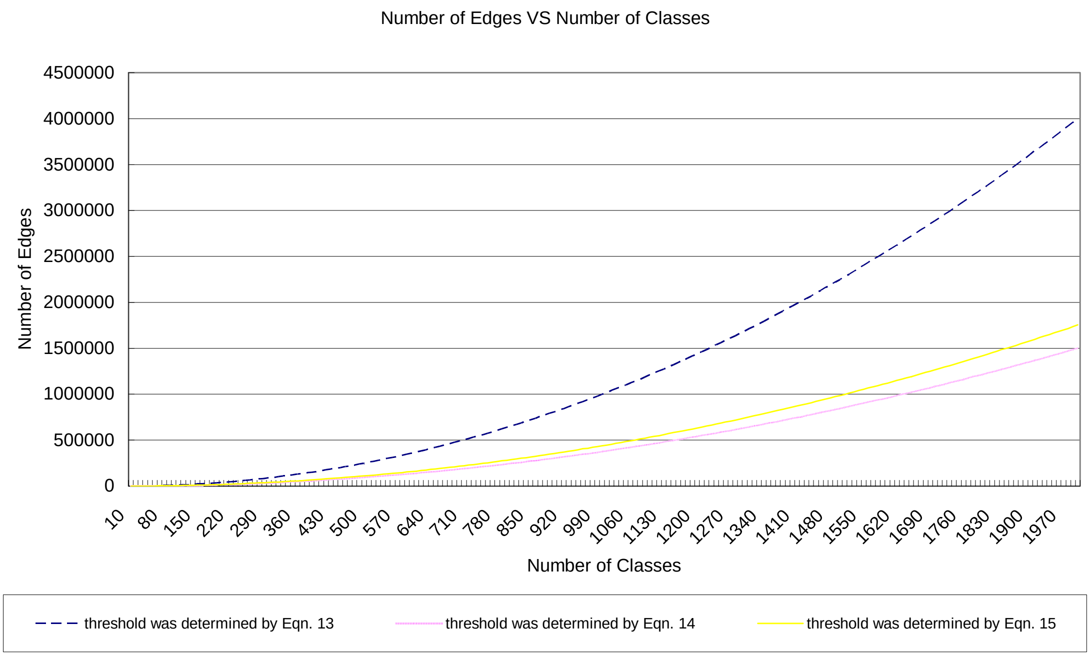


Figure 3.14. Number of Edges Vs. Number of Classes

The ratio between the number of edges in filtered FOM and base FOM was tabulated in Table 3.1. When the threshold was determined by Eqn. (13), the average ratio between the number of edges in the filtered FOM and base FOM is 3.99. When the threshold was determined by Eqn. (14), the number of edges in the filtered FOM is nearly 1.51 times more than the number of the edges in the base FOM. When the threshold was determined by Eqn. (15), the average ratio between the number of edges in the filtered FOM and base FOM is 1.77. In this experiment, the ratios between the number of edges in filtered FOM and base FOM were consistent over the entire range.

Table 3.1. The ratio between the number of edges in filtered FOM and base FOM

Ratio between the number of edges in filtered FOM and base FOM	Threshold was determined by		
	Eqn. (13)	**Eqn. (14)**	**Eqn. (15)**
Min	3.12	1.48	1.75
Max	4.08	1.8	2.04
Average	3.99	1.51	1.77

Experiment III: Average processing time for creating an edge in the FOM

In this experiment, the average processing time required for creating an edge in the FOM was investigated. The total processing time and number of edges in the base and filtered FOM were recorded. The average processing time for creating an edge was calculated by the total processing time over the number of edges created in filtered FOM. Table 3.2 shows part of the experimental results. Figure 3.15 shows the time for creating an edge in with respect to different number of classes. It shows that the time is positive proportional to the number of classes. However, in some classes, the processing time for creating an edge fluctuated. It is probably because the most membership values in base form of FOM were greater than the threshold value T. In addition, the threshold value is one of the factors that affects the processing time. In this experiment, when the threshold was determined by Eqn. 13, the time to create an edge in FOM is smallest.

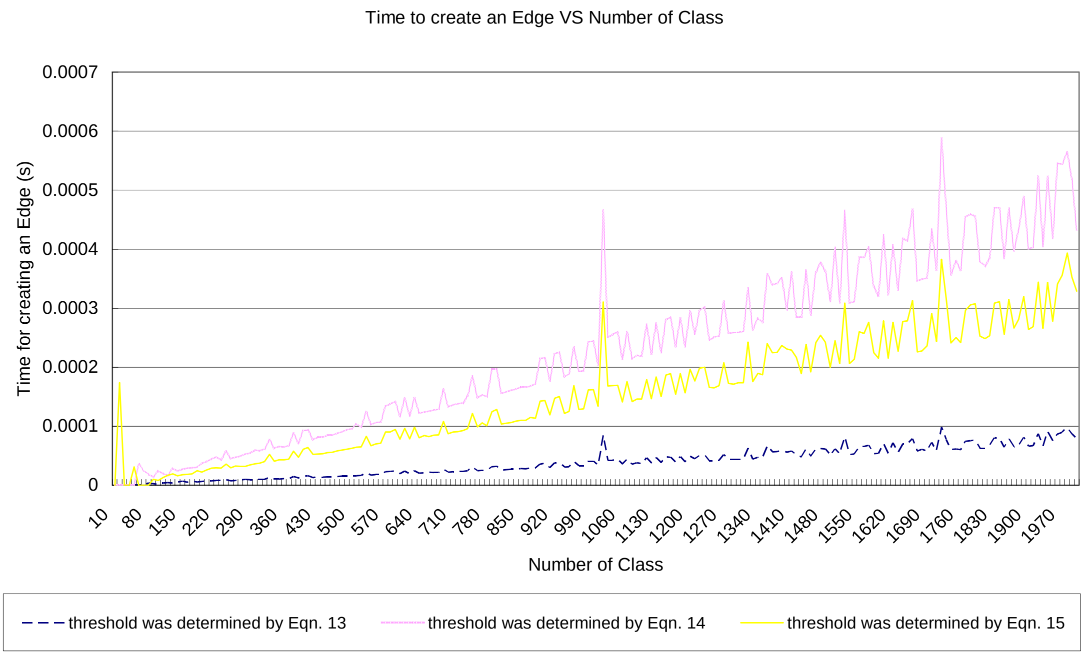


Figure 3.15. Time to Create an edge Vs. Number of Class

From Experiments I – III, it showed that there is no great difference in total processing time for creating a FOM by using various approaches to determine the threshold T. However, when the threshold T was determined by Eqn. (13), the number of edges in filtered FOM is largest and the time for creating an edge is smallest. The main purpose of the threshold is to minimize the number of edges of the FOM. By doing so, this would further reduce the searching size and speed up the searching speed. Eqn. (13) – Eqn. (15) are suggestions to calculate the threshold value in a more flexible way. From the experimental results, by using Eqn. (13) for the threshold value, it could create more edges within the similar total processing time. Thus, it would retain more class relation information in the FOM. It depends on the user to adopt which threshold determination methods. If the system requires a small number of class relation information, then Eqn. (14) and Eqn. (15) are good choices. If the system requires a higher accuracy which need a large number of class relation information, then Eqn. (13) is a good choice. The purposed Eqn. (13) – Eqn. (15) for calculating threshold is not the optimal threshold. They are suggestions for determine the threshold value instead of giving a threshold value between 0 and 1 casually.

Table 3.2. Average processing time for creating an edge in FOM

num of class	Threshold was determined by		
	Eqn. 13	Eqn.14	Eqn.15
100	2.01911E-06	2.4181E-05	7.88177E-06
200	6.79134E-06	3.96717E-05	2.60038E-05
300	9.28126E-06	5.9343E-05	3.63182E-05
400	1.53995E-05	9.28126E-05	6.09468E-05
500	1.5854E-05	9.57121E-05	6.25092E-05
600	1.94292E-05	0.000115789	7.79896E-05
700	2.22179E-05	0.000132523	8.74022E-05
800	3.18411E-05	0.000196199	0.000128466
900	3.66147E-05	0.000216077	0.000143361
1000	4.05698E-05	0.00024437	0.000161775
1100	3.64438E-05	0.000217588	0.000146037
1200	4.91891E-05	0.000294815	0.000196563
1300	4.332E-05	0.000258789	0.000173758
1400	5.59652E-05	0.000297139	0.000230647
1500	6.12197E-05	0.000402766	0.000244716
1600	7.0386E-05	0.000424356	0.000278641
1700	7.26884E-05	0.000433858	0.000290843
1800	6.28201E-05	0.000379007	0.00025263
1900	6.62585E-05	0.000400317	0.000263803
2000	7.83842E-05	0.000432746	0.000328913

3.3.1 Experiments on Real-life Data

Two real life ontologies were adopted to test the performance of the proposed algorithm. The first is the accommodation ontology which was presented before. The other was borrowed from Buche (Buche, et al., 2005) and is a substrate ontology. The substrate ontology has 5 levels with 14 classes. Figure 3.16 shows the ontology graph. The fuzzy set in the substrate ontology was defined as preference. For example, the user prefers food made by Fresh Cheese to food made by Cheese with membership value 1.

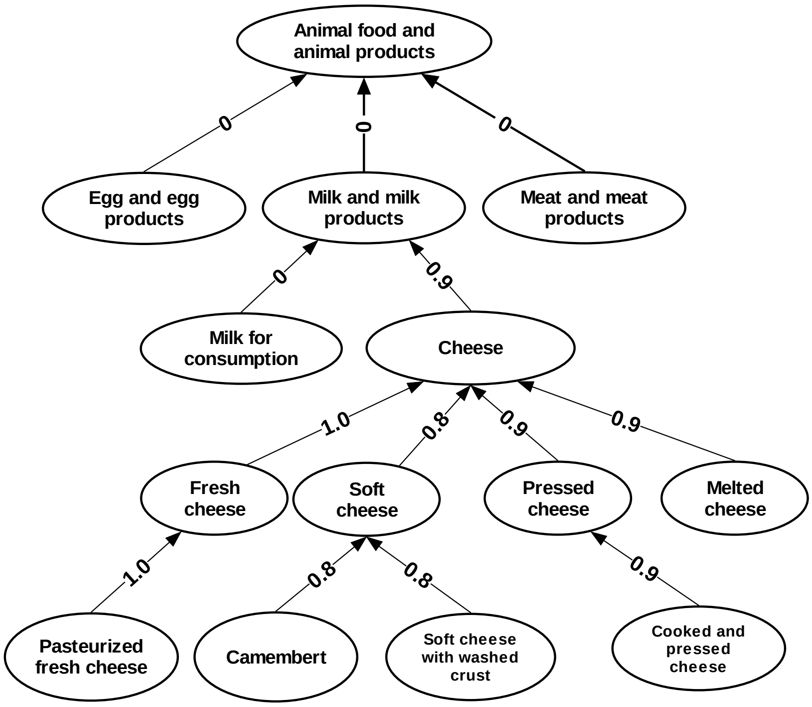


Figure 3.16. Substrate ontology (Buche, et al., 2005)

Experiment IV: Processing time for creating a FOM

The proposed FOM algorithm was applied to these two ontologies. The FOM was created 10 times for each ontology. The processing time for creating a FOM was recorded and the results were tabulated (see Table 3.3). In this experiment, Eqn. (13) was adopted to determine the threshold *T* for filtering the virtual edges in the FOM.

In the base form of these FOMs, the number of classes in the substrate and accommodation ontologies is 14 and 6 respectively. Recall that base form FOM means that the original FOM without creating any edges by inference. Filtered FOM means that some edges in FOM were filtered by using a threshold value. The average processing time for creating a filtered FOM in substrate and accommodation ontology were 923.5ms and 884.3ms respectively. Although the number of classes in the substrate ontology is about twice the number in the accommodation ontology in base FOM, there is no great difference in the processing time.

According to the synthetic data test, the processing time for 20 classes is smaller than 1 ms. In real life data, the processing time for these two ontologies was around 1000 ms. This is probably because most of the processing time was spent on I/O processes such as reading and writing the XML document.

Table 3.3. FOM Processing Time in Substrate Ontology and Accommodation Ontology

No.	FOM Processing Time in Substrate Ontology (ms)	FOM Processing Time in Accommodation Ontology (ms)
1	938	891
2	953	875
3	922	890
4	953	875
5	922	875
6	906	891
7	907	875
8	906	875
9	922	890
10	906	906

Experiment V: Number of edges in FOM with different threshold values

The impact of the threshold value in creating FOM was shown in this experiment. The number of edges in base and filtered FOM was recorded. The threshold value *T* ranged from 0.1 to 1 with a step of 0.1. Tables 3.4 and 3.5 show the number of edges in FOM for substrate and accommodation ontology respectively. When the threshold is 0.1, the ratio between the number of edges in filtered FOM and the number edges in base FOM is around 9 times and 5 times in substrate and accommodation ontology respectively.

By using Eqn. (13), the threshold value for substrate ontology is 0.12 and the number of edges in filtered FOM is 63. In accommodation ontology, the threshold value is 0.39 and the number of edges in final FOM is 26. This shows that Eqn. (13) for calculating the threshold value *T* is acceptable for filtering the edges.

Table 3.4. Number of Edges in FOM (Substrate Ontology)

Threshold Value	Num of edges (initial)	Num of edge (final)
0.1	9	65
0.2	9	47
0.3	9	33
0.4	9	29
0.5	9	25
0.6	9	19
0.7	9	17
0.8	9	15
0.9	9	12
1	9	10

Table 3.5. Number of Edges in FOM (Accommodation Ontology)

Threshold Value	Num of edges (initial)	Num of edge (final)
0.1	6	30
0.2	6	30
0.3	6	30
0.4	6	26
0.5	6	24
0.6	6	22
0.7	6	17
0.8	6	10
0.9	6	7
1	6	6

Experiment VI: File size of FOM

The file size of a document is also a main concern for system development. In this experiment, the file size of the Base and Filtered FOM were compared. Again, Eqn. (13) was adopted to calculate the threshold value for filtering the virtual edges of FOM. The experimental results are tabulated in Table 3.6. The file size of the filtered FOM is around 3 times of the base FOM. The file size of filtered FOM is 5.47KB for substrate ontology and 2.24KB for accommodation ontology.

Table 3.6. File size of FOM

Ontology	Base FOM file size (KB)	Filtered FOM file size (KB)
Substrate	2.28	5.47
Accommodation	1.26	2.24

3.4. Summary

In this chapter, the algorithm of fuzzy ontology map (FOM) is presented. Figure 3.17 shows the flow diagram of how to create a FOM. The purpose of the proposed algorithm is to solve the problem of the hard constraints inherent in current semantics markup languages. By using FOM, we are able to develop an application which supports fuzzy searching.

A fuzzy ontology would be created by using the RDF/OWL document with the FOM. In general, the FOM is a connection matrix which collects the fuzzy information in the ontology. To ensure the applicability and capability, the FOM is written as an XML file. A data-type definition (DTD) was defined to validate the syntax of the XML.

To evaluate the performance of the proposed algorithm, a number of tests are done on syntactic and real-life data for evaluation. In coming chapters, the framework of information filtering by using the fuzzy ontology will be shown. A number of semantic web applications were implemented to show the capability and applicability of FOM.

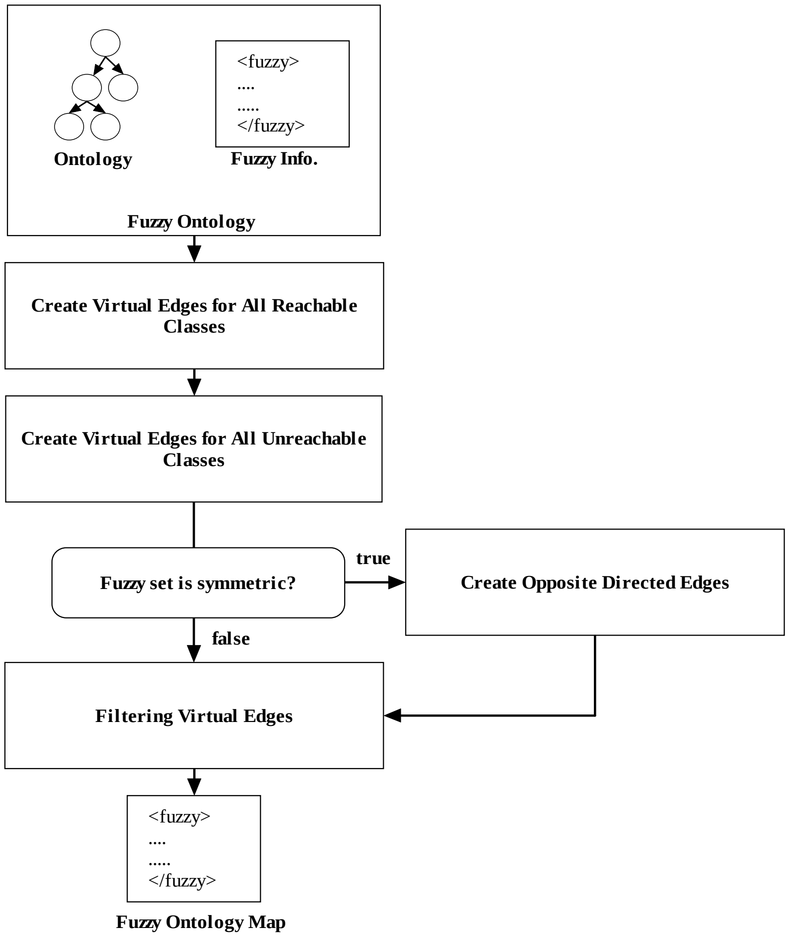


Figure 3.17. Flow diagram for creating a fuzzy ontology map

Chapter 4. Fuzzy Ontology Map in Application - SemTour:HK

4.1 Introduction

The theoretical and technical idea of fuzzy ontology map (FOM) was presented in Chapter 3. The next step is to apply the FOM in applications. In this chapter, a description of the SemTour:HK information portal is presented. SemTour:HK is a travel information portal which combines the Semantic Web technology with the FOM. Unlike other travel information portals, SemTour:HK does not use exact match searching. Rather, users can use fuzzy search functions. SemTour:HK makes use of a travel ontology for storing travel information. One way to model a domain ontology is to reuse or extend an existing ontology, if two domains are sufficiently similar. It is difficult, however, to model a domain ontology completely from scratch. The most common way to model a fresh ontology is to invite an expert in ontology to model the domain ontology. In this chapter, instead of inviting an expert to model the whole travel ontology for the information portal, the travel ontology was modeled by collecting and analyzing the structural information from a number of travel related websites. The travel ontology was modeled by using OWL and RDF. The travel information portal, SemTour:HK, was developed by using the travel ontology and FOM. A small user study was conducted. Some evidence is also established that a user could benefit by using fuzzy search.

4.1.1 Organization of the Chapter

The rest of this chapter describes the travel information portal, SemTour:HK, in detail. Section 4.2 shows some related work. Section 4.3 describes the details of the travel ontology. Section 4.4 describes the system architecture of SemTour:HK. Section 4.5 shows the fuzzy searching by using Fuzzy Ontology Map in SemTour:HK. Section 4.6 shows the user study and presents a brief conclusion with discussion.

4.2 Related Work

There are a number of research projects related to the travel domain. However, most of them are guiding systems rather than tourist information portals. CRUMPET is a research project funded by the European Union. The main aim is to create user-friendly and personalized mobile services for Tourism (Poslad, et al., 2001). CRUMPET adopted multi-agent technology and GPS technology to create a context-aware system. CRUMPET provides tourists with two different kinds of information: static and dynamic. Static information is information collected according to the user’s profile and request. Dynamic information is information gathered according to the user’s location. CRUMPET also learns user preferences and interests to further filter irrelevant information. CRUMPET does not adopt reuse knowledge from other domains.

Cyberguide (Abowd, et al., 1997) is a personalized tourist guide for museum visitors. The tourist guiding system provides information to users based on the data of user’s position and orientation. There are two different types of Cyberguide: indoor and outdoor. However, like CRUMPET, it has weak support for knowledge sharing and reasoning.

MyCampus (Sadeh, et al., 2002) is a research project developed at Carnegie Mellon University. MyCampus is a Semantic Web environment for context-aware mobile services. The current implementation of MyCampus combines a number of technologies such as OWL, reasoning, context-aware agent and OWL Rule Extension. The system can provide location-based movie recommendations and weather information. MyCampus is a user-friendly and intelligent application. It fully utilizes the latest Semantic Web technologies.

Johanna and Schubert (Lim and Foo, 2003) created a web-based virtual exhibition system (VES) based on an XML-based digital archive. They defined a rich layered set of metadata of image and text artifacts. These metadata ease the development and maintenance of the content in the system. The idea of a virtual exhibition system is similar to our proposed portal. VES uses XML for processing, which provides syntax for structured documents but has no semantic constraints such as string literals. The following sections show how to integrate a tourist information portal with the latest Semantic Web Technology

4.3 Construction of Travel Ontology

4.3.1 Collection of Structural Information

The structural information from a number of travel related web sites was collected. This included terms used in the site map and the website menu. Web developers often group related contents into categories. The site structure is a common way for people to define a domain (e.g. travel). Instead of inviting an ontology expert to model the ontology, the structural information from a number of websites were collected and recorded for analysis.

A number of websites that related to Hong Kong Travel from Google Web Directory and Open Directory (domz.org) were collected. In Google Web Directory, there are 27 websites related to Hong Kong Travel (Regional > Asia > Hong Kong > Travel and Tourism > Travel Guides). In Open Directory, there are 31 websites that are related to Hong Kong Travel (Regional > Asia > Hong Kong > Travel and Tourism > Travel Guides). Since Google Web Directory integrates its search technology, PageRank (Brin and Page, 1998), with Open Directory for searching, some websites in Open Directory are duplicated in the Google

Directory. After removing duplicates, there are 32 websites (2 being unreachable). Table 4.1 shows information about these websites. The recorded structural information contains 153 terms. After filtering and grouping similar terms, the most commonly used term is "Shopping". Twelve websites contained this term. Table 4.2 shows the top ten most common used terms in Hong Kong Travel-related websites. The terms are further filtered and grouped together with similar meanings (See Table 4.3) and this information was adopted to model the upper-level travel ontology.

Table 4.1. Websites related to Hong Kong Travel Guides.

No.	Name	Address
1	Lonely Planet - Hong Kong	http://www.lonelyplanet.com/dest/nea/hong.htm
2	Footprint Guides Hong Kong	http://www.footprintguides.com/Hong-Kong/
3	Regi Tour	http://www.regit.com/regitour/hongkong/regitour.htm
4	Dr Martin Williams	http://www.drmartinwilliams.com
5	Walk The Talk	http://www.walkthetalk.hk/
6	Arthur Frommer's Budget Travel Online	http://www.frommers.com/destinations/hongkong/
7	rec.travel Guide to Hong Kong	http://www.math.toronto.edu/~joel/hongkong.html
8	Hong Kong Tourist Guide +	http://www.yp.com.hk/yptourist_e03/en/html/tourist_index.aspx
9	PassPlanet.com - Hong Kong +	http://www.passplanet.com/HK/index.htm
10	Hong Kong Hotels Guide	http://www.hong-kong-hotels-guide.com/
11	12hk : The Unofficial Guide	http://www.12hk.com
12	Hong Kong FastFacts	http://www.hkfastfacts.com/
13	Hong Kong Travel	http://www.hong-kong-travel.org/
14	Writing, Photography and Nature Tourism in East Asia	http://martinwilliams.tripod.com/index.html
15	Travelocity's Destination Guide	http://dest.travelocity.com/DestGuides/geo_main/0,1743,TRAVELOCITY|2771,00.html

16	Stuie's Hong Kong Page	http://uk.geocities.com/expatbeamish/SB/HK/
17	Hong Kong Tong	http://www.hongkongtong.net/
18	Hong Kong Travel Guide	http://www.luketravels.com/hong-kong/
19	Hong Kong on Web	http://www.hongkongonweb.net/
20	Hong Kong Help	http://www.hongkonghelp.com/
21	Visiting Hong Kong	http://www.visitinghongkong.co.uk/
22	Hong Kong Travellers	http://home4u.hongkong.com/lifestyle/travel/hktravellers
23	Discover Sai Kung	http://www.discoversaikung.com/
24	Travallo: Hong Kong	http://www.travallo.de/laender/asia/china/hongkong.html
25	Worldsurface.com - Hong Kong Guide	http://www.worldsurface.com/browse/location-country.asp?locationid=112
26	BootsnAll	http://www.BootsnAll.com/asiatravelguides/hk/hk.shtml
27	CNN City Guides: Hong Kong	http://www.whatsontheplanet.com/wow/ptnr/cnn/page.jsp?fx=destination&loc_id=147486&xml_set=wow.city
28	Asia Friends Network - Hong Kong Tourism	http://www.countries.asiafriendsnetwork.com/HongKong/
29	Explore Sai Kung	http://www.exploresaikung.com/
30	Hong Kong Streets	http://www.hkstreet.com/
31	I Love Hong Kong	http://free.hostdepartment.com/i/ihearthk/
32	UnRealHongKong.com	http://www.unrealhongkong.com/

+ - unreachable

Table 4.2. Ten most commonly used categories in Hong Kong travel guide websites

Rank	Category Term	Frequency
1	Shopping	12
2	Hotels	6
3	Getting Around	6
4	Link	6
5	Food	5
6	History	5
7	Attractions	5
8	Festivals	5
9	Accommodation	5
10	Transportation	4

Table 4.3. Ten most commonly used categories after filtering and grouping

Rank	Category Term	Frequency
1	See / Sightseeing / Sights / Spots / Unique Sights / Interesting Places / Attractions / Landmarks / Places to visit / Getting Around / Go Around	18
2	Food / Cruises / Restaurant / Dining / Eat / Bars and Restaurants / Eating and Drinking / Food & Drink	16
3	Shopping / Shops / Shopping & Malls / Buy	15
4	General Information / General / Overview / General HK Info / City Facts & Info / Country Info / History	14
5	Accommodation / Places to stay / Sleep / Hotels	13
6	News & Events / News / Events / Festivals and Events / Festival / Festivals and Holidays / Public Holidays	11
7	Transportation / Transport / Getting There	10
8	Others / Misc / Link	6
9	Nightlife / Night	5
10	Weather / Local Weather	4

4.3.2 Travel Ontology Design

To ensure that the ontology would be reused and shared with others, an upper-level travel ontology was modeled. An upper-level ontology is an ontology that provides a set of basic concepts. If an ontology contained all the concepts, the ontology would be too specific and would not be reused by others. After collecting and analyzing the structural information, an upper-level travel ontology was defined. The travel ontology contains seven main classes, each with subclasses. Figure 4.1 shows details of the class relationships in the travel ontology.

To allow the ontology to be reused by others, the travel ontology was modeled in Web Ontology Language (OWL) by using Protégé[5]. Protégé is a free, open-source platform with a friendly user interface that provides a set of tools for constructing domain model and knowledge-based applications with ontologies. Figure 4.2 depicts a partial OWL of the travel ontology. Three other ontologies: Cuisine, Shop and District were also defined to enhance the functionality of SemTour:HK. Figure 4.3 shows the class relationships of the cuisine

5 Protégé, http://protege.stanford.edu/

ontology.

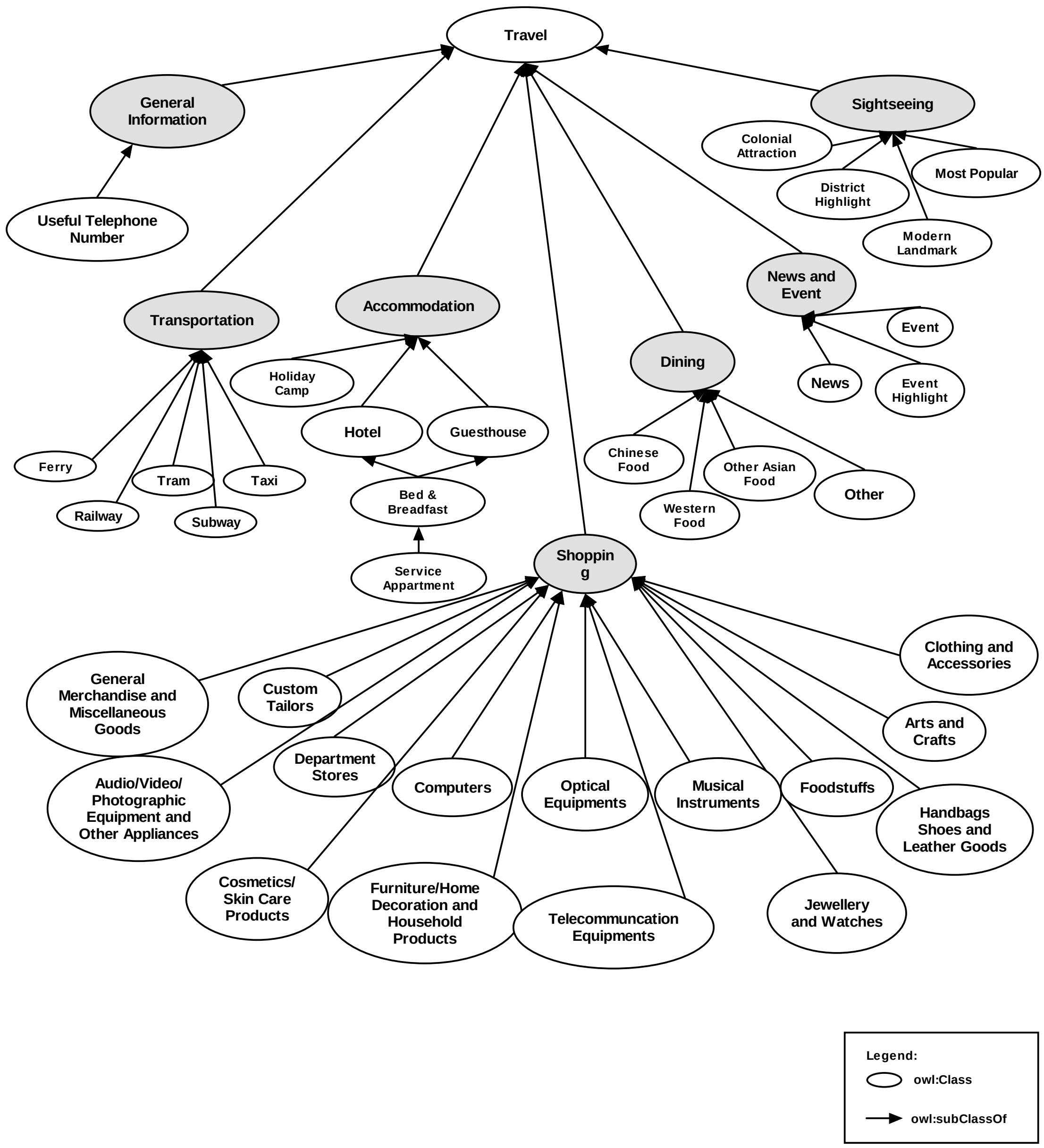


Figure 4.1. Travel Ontology

```
…
  <owl:Class rdf:ID="Custom_Tailor">
    <rdfs:subClassOf rdf:resource="#Shop_Type"/>
  </owl:Class>
  <owl:Class rdf:ID="Useful_Telephone_Number">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="General_Information"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="Handbags_Shoes_and_Leather_Goods">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Shopping"/>
    </rdfs:subClassOf>
  </owl:Class>
    <owl:Class rdf:ID="Hotel">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Accommodation"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:ObjectProperty rdf:ID="hasCusine">
    <rdfs:range rdf:resource="#Cuisine_Type"/>
    <rdfs:domain>
      <owl:Class>
        <owl:unionOf rdf:parseType="Collection">
          <owl:Class rdf:about="#Chinese_Food"/>
          <owl:Class rdf:about="#Other_Asian_Food"/>
          <owl:Class rdf:about="#Western_Food"/>
          <owl:Class rdf:about="#Other"/>
        </owl:unionOf>
      </owl:Class>
    </rdfs:domain>
  </owl:ObjectProperty>
...
```

Figure 4.2. The partial OWL of the travel ontology

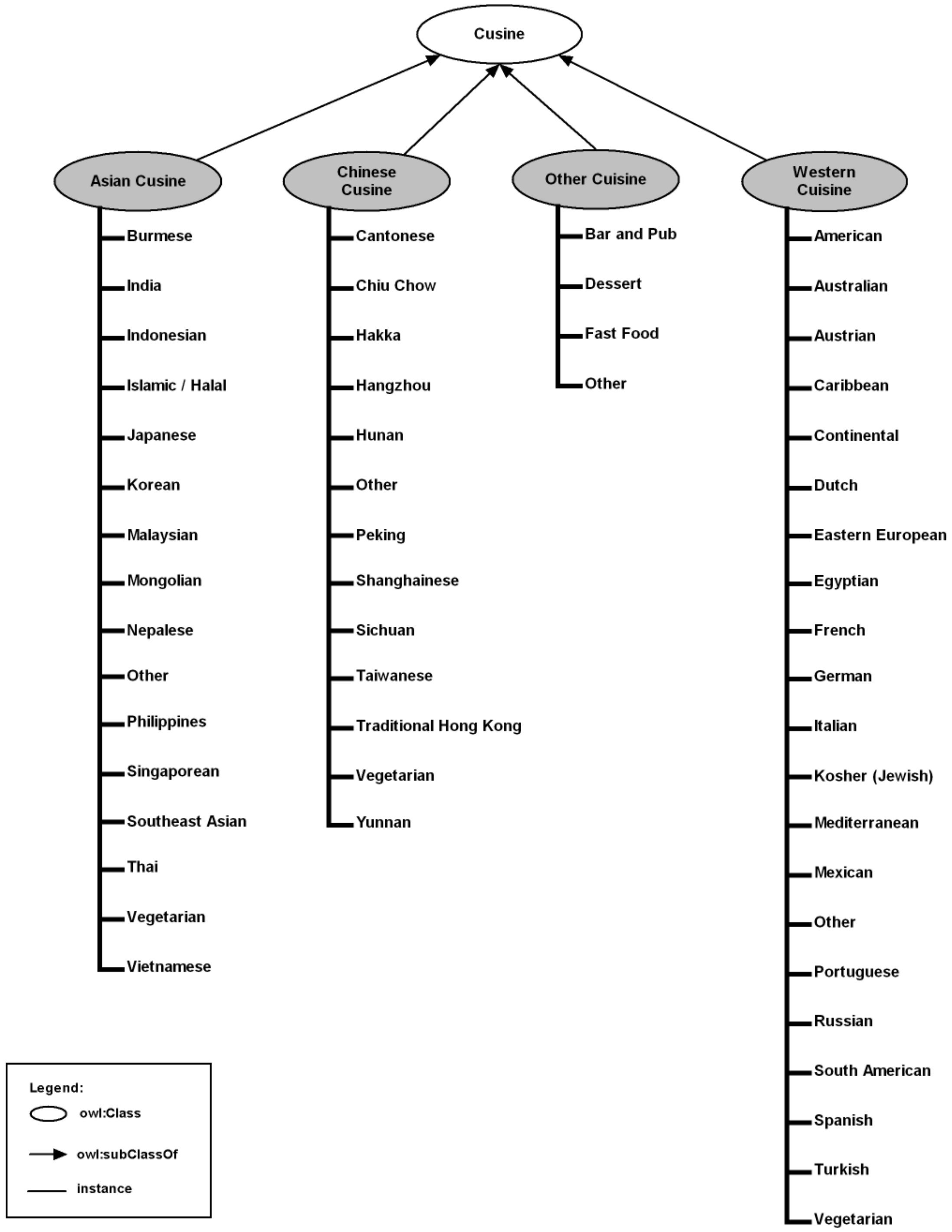


Figure 4.3. Cuisine Ontology

```xml
<Hotel rdf:ID="THE_MARCO_POLO_HONGKONG_HOTEL">
    <hasTelephoneNumber rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >2113 0088</hasTelephoneNumber>
    <hasFaxNumber rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >2113 0011</hasFaxNumber>
    <hasEmailAddress rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >hongkong@marcopolohotels.com </hasEmailAddress>
    <hasURL rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >www.marcopolohotels.com </hasURL>
    <hasDistrict rdf:resource="#Tsim_Sha_Tsui"/>
    <hasRoom rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
    >710</hasRoom>
    <hasStar rdf:datatype="http://www.w3.org/2001/XMLSchema#float"
    >5.0</hasStar>
    <hasStreet rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Canton Road</hasStreet>
    <hasName rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >The Marco Polo Hongkong Hotel</hasName>
    <hasAddress rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Harbour City, 3 Canton Road, Tsimshatsui, Kowloon</hasAddress>
    <hasStandardRoomPrice rdf:datatype="http://www.w3.org/2001/XMLSchema#float"
    >1050.0</hasStandardRoomPrice>
  </Hotel>
```

Figure 4.4. An instance of hotel

4.3.3 Properties in Travel Ontology

There are a number of class properties in the travel ontology and seven main classes. Each class has its own subclasses and class properties such as:

- *hasDesc*: Description about the resource
- *hasName:* Name of the resource
- *hasTelephoneNumber:* Telephone number of the resource
- *hasEmailAddress:* Email address of the resource

- *hasFaxNumber:* Fax number of the resource
- *hasOpeningHours:* Opening hours of the resource
- *hasRoom:* Number of room of the resource
- *hasStar:* Rank (number of star) of the resource
- *hasURL*: URL about the resource

There were more than twenty properties in the travel ontology. Some class properties, such as *hasName* and *hasDesc*, were used by different classes. The cuisine ontology has four subclasses; each subclass having a number of instances. The travel ontology dining subclass has a property called *hasCuisine* which refers to these instances. The travel ontology has around 420 instances among seven main classes for demonstration and evaluation. Figure 4.4 shows a hotel instance in the travel ontology.

4.3.4 Construction of Fuzzy Ontology Map (FOM)

The purpose of FOM is to further extend the functionality of the current Semantic Web application. As mentioned before, the ontology markup language is based on bivalent logic. As a result, a subject in RDF/OWL either belongs to a member of a class or it does not. However, real-world information is vague or imprecise. In this research work, a FOM is constructed to extend the capability of the current ontology to model real world vague information.

In general, searching in RDF/OWL is exact match searching. This limitation would be extended by using FOM. In this application, an accommodation FOM was developed. The fuzzy set for this FOM is similarity. By using this FOM, the ontology would include information about the similarity between classes. By using this FOM, a fuzzy search function for accommodation would be developed. Figure 4.5 shows the fuzzy ontology of accommodation class (which is the same as the accommodation example). In this accommodation FOM, the class Holiday Camp is similar to the class Accommodation with membership value 0.5. The greater the membership value, the greater the similarity.

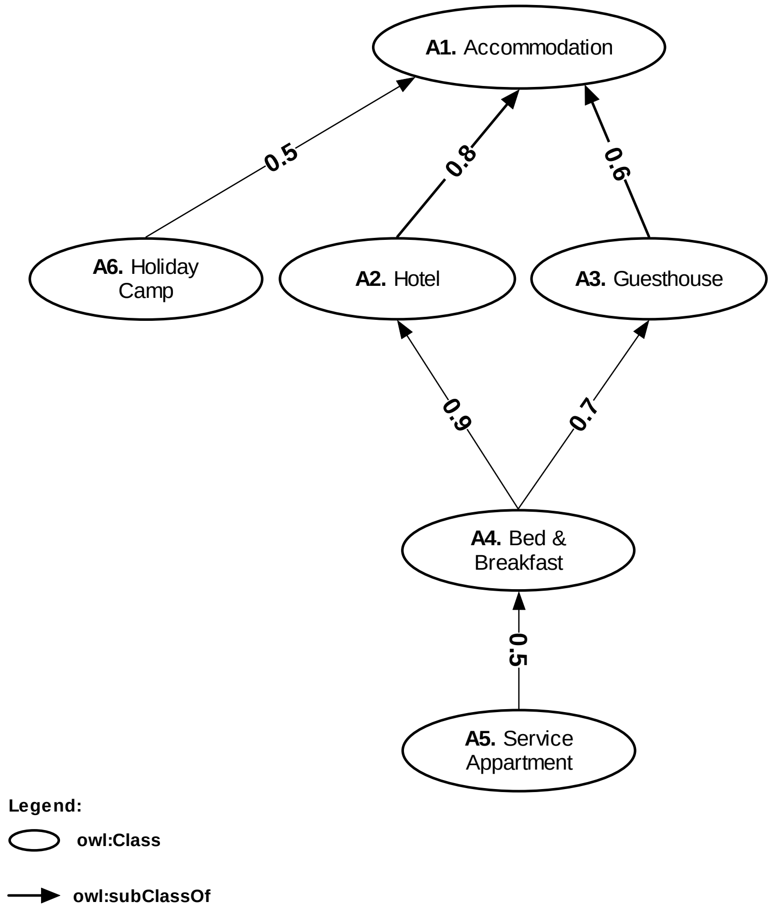
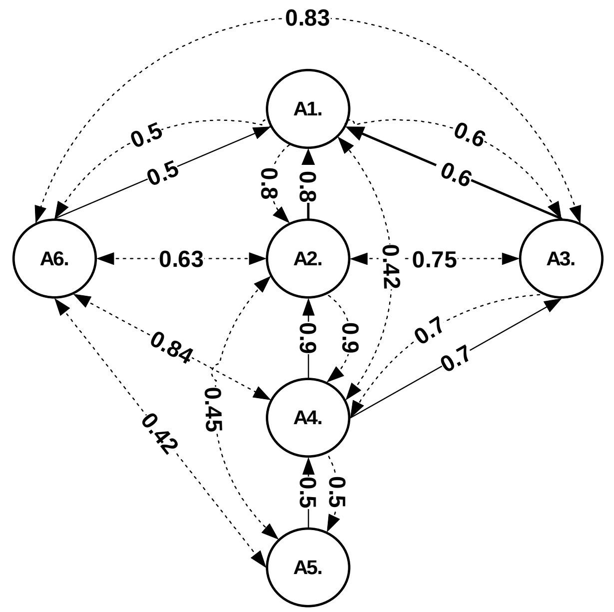


Figure 4.5. Accommodation fuzzy ontology map

A FOM of accommodation is created by using the proposed algorithm (for more details, please refer to Chapter 3). The membership threshold value was set to 0.0 and min(x,y) is adopted as the comparative function. Figure 4.6 shows the matrix of accommodation FOM. The FOM is saved as a fuzzy XML document. Figure 4.7 shows part of the content of the accommodation FOM. The FOM was applied together with the travel ontology to develop the SemTour:HK portal.



	A1	A2	A3	A4	A5	A6
A1	-	0.8	0.6	0.42	0	0.5
A2	0.8	-	0.75	0.9	0.45	0.63
A3	0.6	0.75	-	0.7	0	0.83
A4	0.42	0.9	0.7	-	0.5	0.84
A5	0	0.45	0	0.5	-	0.42
A6	0.5	0.63	0.83	0.84	0.42	-

Figure 4.6. Matrix of accommodation fuzzy ontology map

```
<fuzzy>
<fuzzyset name="similarity" issymm="true" />
<fuzzyrelation fromclass="Accommodation">
<toclass membershipvalue="0.8">Hotel</toclass>
<toclass membershipvalue="0.6">Guesthouse</toclass>
<toclass membershipvalue="0.42000002">Bed_n_Breakfast</toclass>
<toclass membershipvalue="0.5">Holiday_Camp</toclass>
</fuzzyrelation>
<fuzzyrelation fromclass="Hotel">
<toclass membershipvalue="0.8">Accommodation</toclass>
<toclass membershipvalue="0.75">Guesthouse</toclass>
<toclass membershipvalue="0.9">Bed_n_Breakfast</toclass>
<toclass membershipvalue="0.45">Service_Appartment</toclass>
<toclass membershipvalue="0.625">Holiday_Camp</toclass>
</fuzzyrelation>
…
</fuzzy>
```

Figure 4.7. The content of the fuzzy XML of fuzzy ontology map of accommodation

4.4 SemTour:HK

The details of the semantic tourist information portal – SemTour:HK are shown in this section. In Section 4.4.1, the system architecture of SemTour:HK is shown. In Section 4.4.2, information on how to use fuzzy ontology map to develop a fuzzy searching in SemTour:HK is shown.

4.4.1 System Architecture of SemTour:HK

SemTour:HK is a tourist information portal which allows tourists to search for information about accommodation, sightseeing, and dining in Hong Kong. The portal was developed based on the latest Semantic Web technology and implemented in Java on the Jena Semantic Framework. Figure 4.8 depicts the system architecture of SemTour:HK. Users can access the information portal through the web browser. Figure 4.9 shows the homepage of SemTour:HK. The tourist information server is a web server that handles user requests.

In this prototype, Apache Tomcat[6] was employed as the tourist information server. After the information server receives the user request, the information server forwards the request to Jena Framework (Carroll et al., 2004) to gather the related information. Jena is an open source Java framework for building Semantic Web applications. It offers a number of APIs for handling RDF, RDFS and OWL. The main purpose of using Jena is to parse and query the travel ontology. Jena searches for the data from the travel ontology by using the Simple Protocol And RDF Query Language (SPARQL). SPARQL is a query language and protocol for accessing RDF designed by the W3C RDF Data Access Working Group. Figure 4.10 shows an example of SPARQL that is used to get related information about Guesthouse. The search returns as an RDF graph and is sent back to the tourist information server. Then, the server generates a result page and displays this to the user. Figure 4.11 shows a screenshot of

[6] Apache Tomcat, http://tomcat.apache.org/

a shopping page in SemTour:HK.

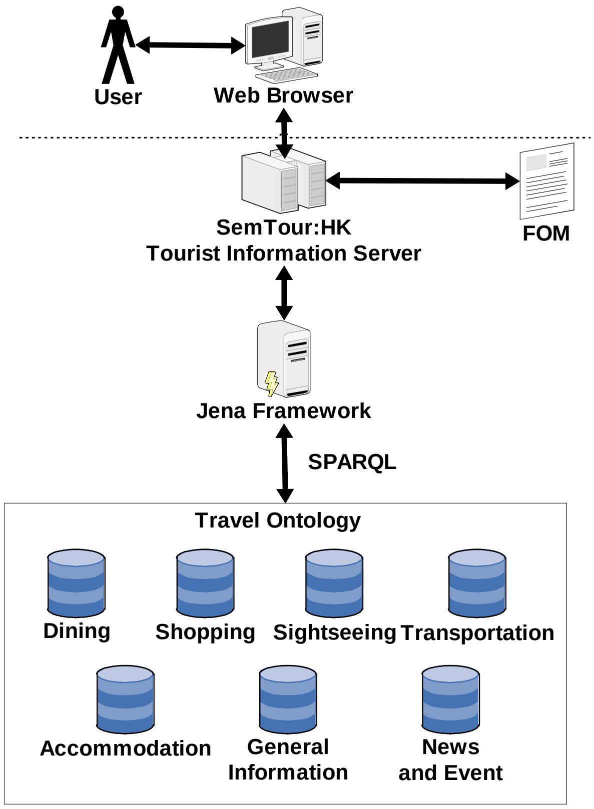


Figure 4.8. System architecture of SemTour:HK

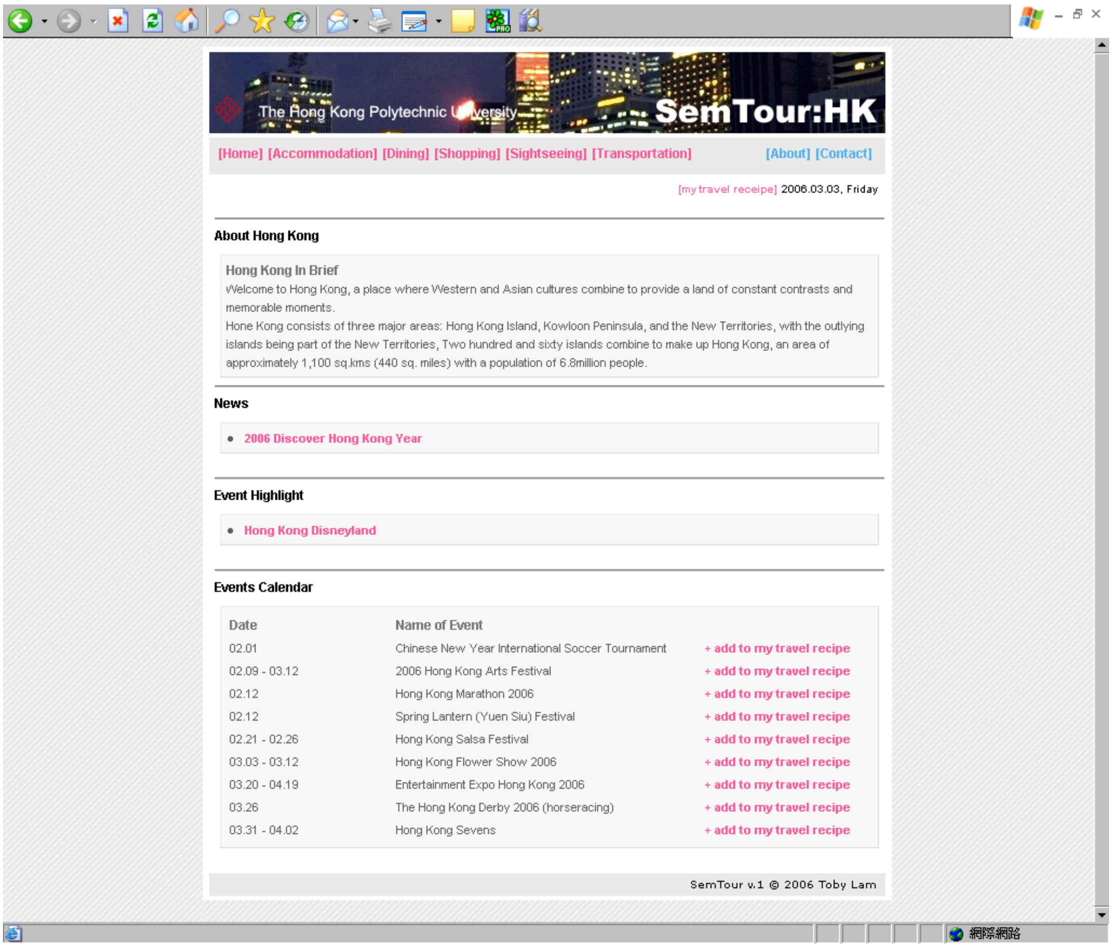


Figure 4.9. Screenshot of SemTour:HK Homepage

```
# Select related information about Guesthouse
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX travel: <http://www.comp.polyu.edu.hk/~cshwlam/ontology/travel.owl#>
SELECT ?URI ?NAME ?ADDRESS ?DISTRICT ?ROOM ?URL ?EMAIL ?PRICE ?FAX ?PHONE
WHERE { ?URI rdf:type travel:Guesthouse .
?URI travel:hasName ?NAME .
?URI travel:hasAddress ?ADDRESS .
?URI travel:hasDistrict ?DISTRICTURI .
?DISTRICTURI travel:districtName ?DISTRICT .
?URI travel:hasRoom ?ROOM .
OPTIONAL {?URI travel:hasURL ?URL } .
OPTIONAL {?URI travel:hasEmailAddress ?EMAIL }.
OPTIONAL {?URI travel:hasStandardRoomPrice ?PRICE } .
OPTIONAL {?URI travel:hasFaxNumber ?FAX }.
OPTIONAL {?URI travel:hasPhoneNumber ?PHONE        }
}
```

Figure 4.10. A SPARQL example used in SemTour:HK

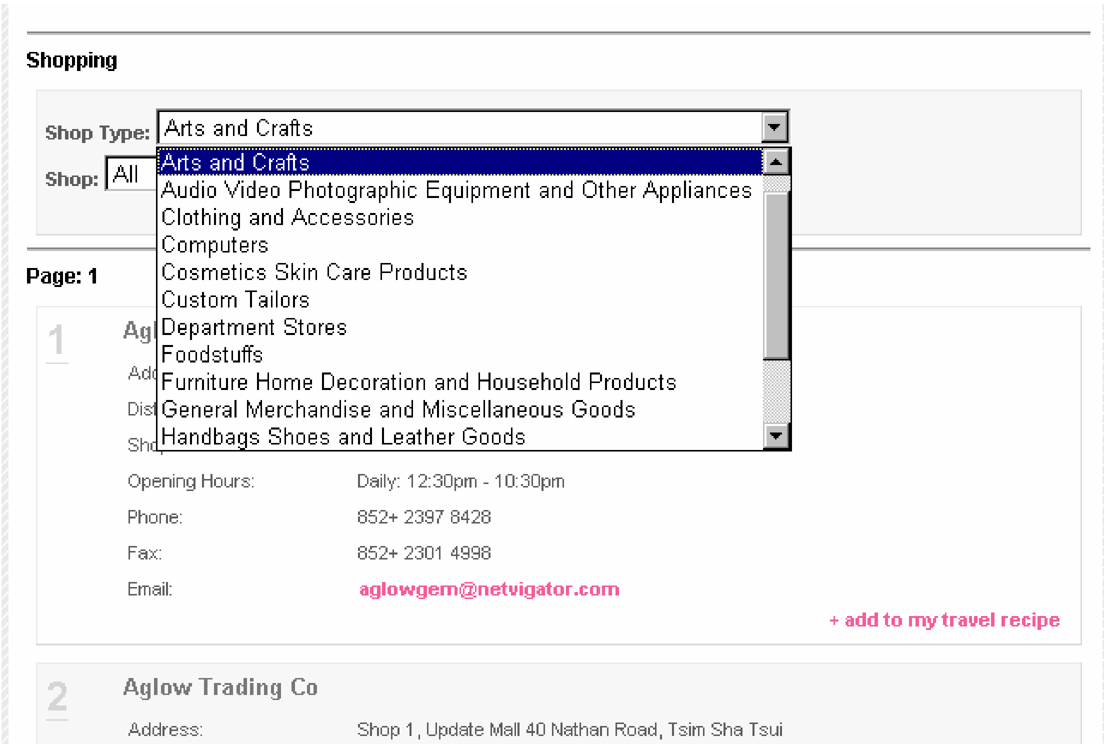


Figure 4.11. Screenshot of a shopping page in SemTour:HK

4.5 Fuzzy Searching by using Fuzzy Ontology Map

SemTour:HK supports fuzzy searching by using the fuzzy ontology map. A questionnaire is designed for users to proceed to fuzzy searching. Users can use the questionnaire to find similar accommodation to a particular class. Figure 4.12 shows a screenshot of the fuzzy searching page. The page which contains two fields for users to enter data into. The first one is the relation from and the other is the minimum level. The relation from field is used to let the user select the accommodation type that he/she would like to search for. For example, if the user would like to find accommodation which is similar to a hotel, then the relation from field should select hotel. The minimum level lets the user filter the search result. The search result must be greater than or equal to the select value. To ease user understanding, five levels of similarity are defined: similar (1.0), very similar (0.8), more or less similar (0.5), not very similar (0.2) and not similar (0.0).

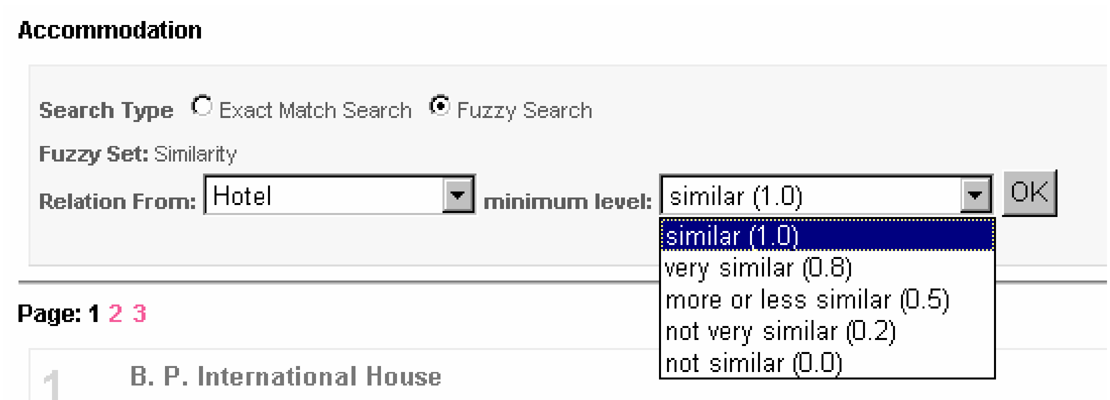


Figure 4.12. Screenshot of fuzzy searching page

After the user selects the fuzzy searching criteria and submits it to the tourist information server, the server locates the accommodation fuzzy ontology map. After the server finds the FOM, it tries to parse the FOM. After this, the server tries to locate the relation from accommodation in the FOM. The class related to this accommodation is retrieved for further processing. Classes with a smaller membership value than the user selected are removed. The filtered classes are sorted in descending order according to their membership value. After this, a number of SPARQL queries is generated depending on the number of filtered classes. These SPARQL queries send to JENA to query the database. The server collects the query result for each class and then displays it to the user. Figure 4.13 shows the sequence diagram of the fuzzy search function.

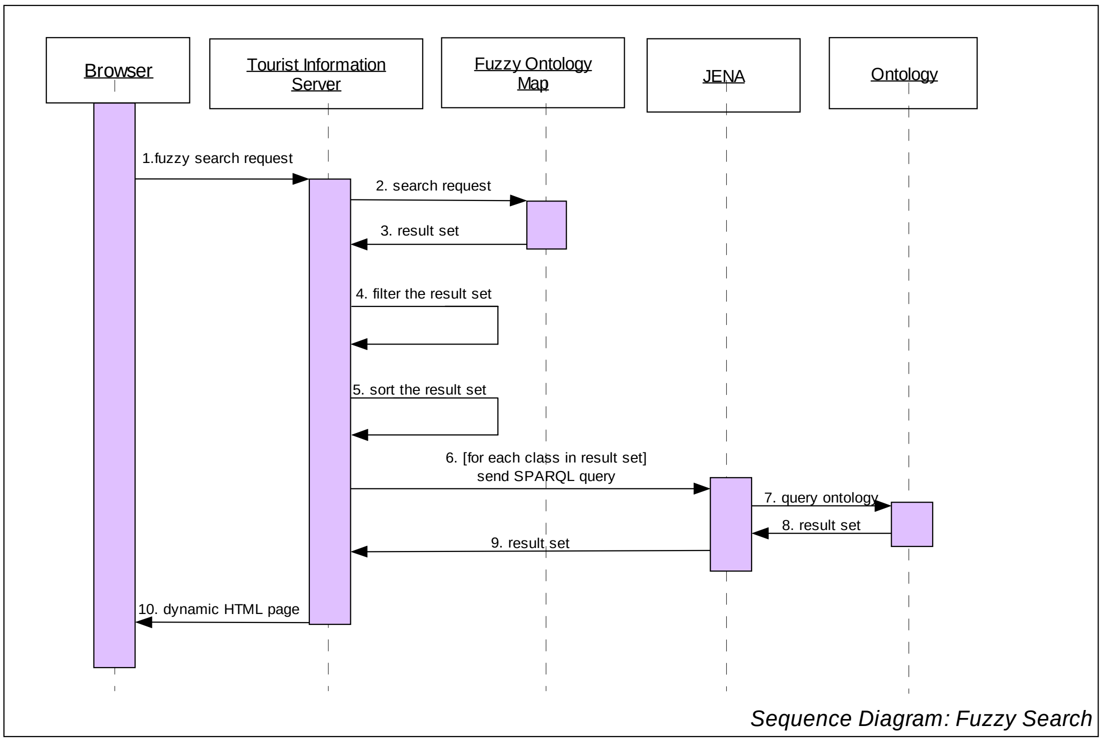


Figure 4.13. Sequence diagram of fuzzy search function

4.6 Evaluation and User Study

A prototype of a tourist information portal, SemTour:HK, was developed using Java Server Pages (JSP) based on a Jena framework. The tourist information server is installed in a P4 1.8GHz computer with 768MB memory. The main objective of the experiment is to evaluate the performance of the fuzzy ontology map.

Experiment I: Processing Time for exact match searching

SemTour:HK has six main pages: home, accommodation, dining, shopping, sightseeing, and transportation. The processing time required for generating the result page is recorded. Each page is requested 10 times and the processing time is recorded. Figure 4.11 shows the processing time for each page. The home page has the longest processing time. The average processing time for the home page is 3240.5ms. The transportation page has the shortest processing time. The average processing time for the transportation page is around 829.6ms. The home page has the longest processing time as it shows four different classes of information. The home page of SemTour:HK has four different classes of information: About Hong Kong, News, Event Highlight and Events Calendar. They mainly come from the News and Event and General Information classes in the travel ontology. Since the home JSP page required four different SPARQL queries to get the related information, the processing time is longer that of the others. The transportation page has the shortest processing time as the SPARQL query is simple compared with other pages. Transportation has a smaller number of instances than other classes. Table 4.4 shows the average processing time for each page.

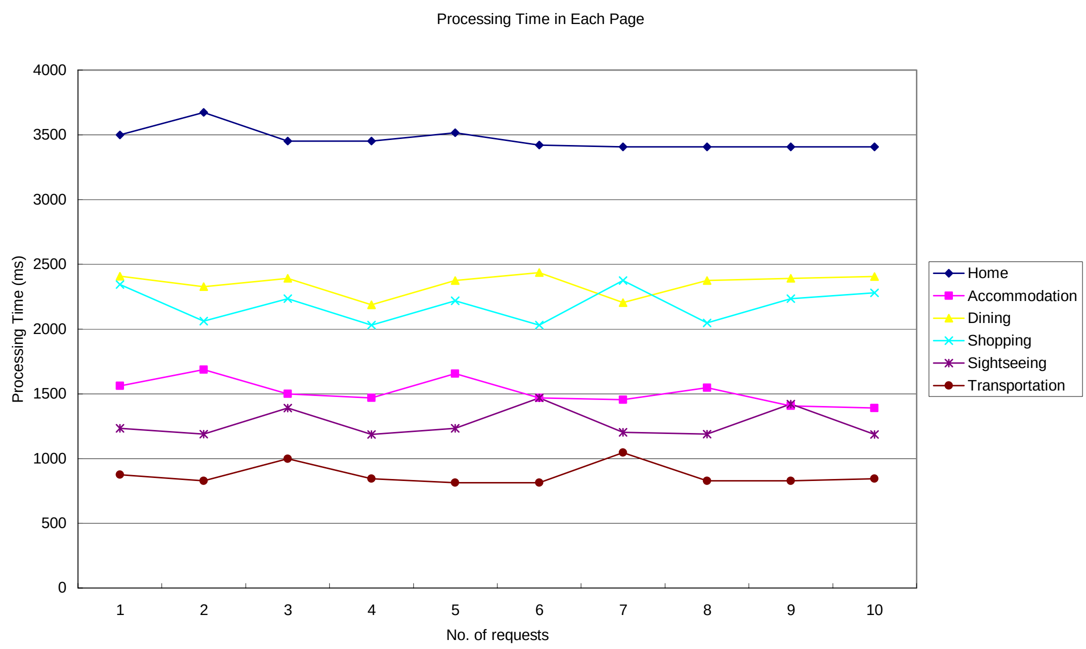


Figure 4.11. Processing time for each page

Table 4.4. Average processing time for each page

Name of the page	Average Processing Time (in ms)
Home	3464
Accommodation	1514.1
Dining	2350.2
Shopping	2185.9
Sightseeing	1270.3
Transportation	872

Experiment II: Processing Time for fuzzy searching

In this experiment, the processing time for fuzzy searching in the accommodation page was recorded. The processing time for fuzzy searching was compared with the exact match searching. Fuzzy searching with different minimum levels of similarity were sent to the server 10 times. The processing time was recorded. The fuzzy searching criterion of relation from was set to Hotel. Figure 4.12 shows the processing time for fuzzy searching with different minimum levels of similarity and exact match searching of the accommodation page.

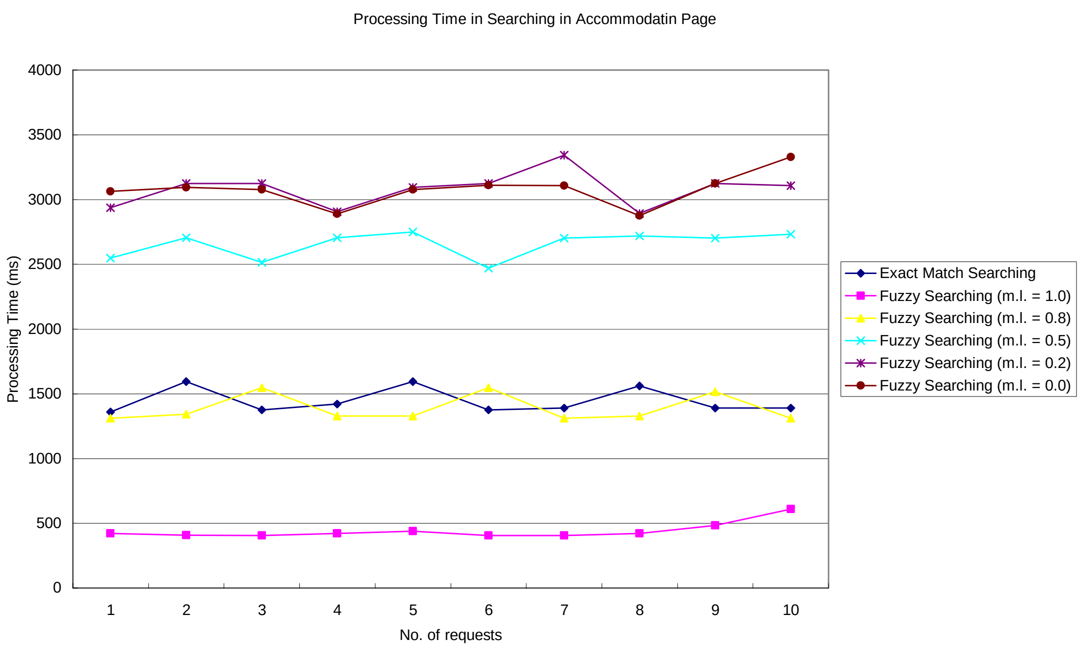


Figure 4.12. Processing time in fuzzy searching and exact match searching in accommodation page

The processing for fuzzy searching depends on the minimum level of the similarity. When the minimum level of the similarity was not similar (0.0), the fuzzy searching tries to find all the data that is related to the class Hotel. As a result, the processing time was much longer than others, with an average processing time of around 2.79s. When the minimum level of the similarity was more or less similar (0.5), the average processing time for fuzzy searching is nearly the same as the exact match searching which is around 1.4s. This is probably because the number of search results is nearly the same as in the exact match searching. When the minimum level of the similarity was similar (1.0), the average processing time was around 0.4s. Since there is no class similar to Hotel with a membership value equal to 1.0, no data was retrieved. As a result, there is a great drop in the processing time. Table 4.5 shows the average processing time for fuzzy searching with different minimum levels of similarity.

The processing time affects the performance of the system. To further reduce the processing time, the accommodation FOM would be filtered with a higher threshold value (current threshold value is 0). However, using such an approach, there is some data loss.

Table 4.5. Processing time for searching the accommodation page

Type of Search	Average Processing Time (in ms)
Exact Match Searching	1445.1
Fuzzy Searching (m.l. = 1.0)	402.2
Fuzzy Searching (m.l. = 0.8)	1261.5
Fuzzy Searching (m.l. = 0.5)	2413.5
Fuzzy Searching (m.l. = 0.2)	2798.2
Fuzzy Searching (m.l. = 0.0)	2795.4

* m.l. = minimum level

Experiment III: User Acceptance Test

To test the effectiveness of the system, 20 people aged 18 to 25 were invited to use the system and answer interviewer questions. Users were invited to browse the website and then asked to answer the questions shown in Table 4.6.

Table 4.6. Questions used in the interview

Question	Choice
1. Can the system help tourist to find related tourist information in Hong Kong?	Yes / No
2. Is the loading time acceptable?	Yes / No
3. Is Fuzzy Searching useful for tourist?	Yes / No
4. Is the system easy to use?	Yes / No
5. Will you recommend this website if your friend will visit to Hong Kong?	Yes / No

Table 4.7. Result of the interview

Answer	Question Number				
	1	2	3	4	5
Yes	16 (80%)	12 (60%)	15 (75%)	18 (90%)	15 (75%)
No	4 (20%)	8 (40%)	5 (15%)	2 (10%)	5 (15%)

The results of the survey are shown in Table 4.7. Most of the interviewees agreed that the system can help tourists find tourism related information in Hong Kong. However, nearly half of those interviewed felt that the loading time was not acceptable, especially the home page. 75% of the interviewees agreed that the Fuzzy Search function is useful for them to find similar accommodation. 75% of the interviewees would recommend this portal site to a friend coming to Hong Kong.

Most of the users felt that the fuzzy search function should be employed on other pages, especially the Dining page. Since there are lots of choices in cuisine, it is difficult for them to get the cuisine they are interested in by using exact match searching. They are required to spend time browsing all the different kinds of cuisine. However, if fuzzy search was employed on the Dining page, then once the user selects a specific cuisine (e.g. Traditional Hong Kong), then the system would give recommendations to the user with similar cuisines (e.g. Chiu Chow). This would help to further reduce the browsing time.

4.6 Discussion and Conclusion

A tourist information portal based on the latest Semantic Web technology was implemented. The fuzzy search function developed by using a fuzzy ontology map was successfully developed. Most users felt the fuzzy search function in the portal was useful. The portal is easy to use and can provide tourist information related to Hong Kong. A number of the interviewees found the loading time unacceptable. In the future, the whole portal site will be re-designed and re-implemented. To reduce page loading times, the coming version of SemTour:HK will have less information on the first page and will fine tune the JSP program . In addition, the fuzzy search function will be added in other kinds of pages such as Dining and Shopping.

This chapter described how to model a travel ontology by collecting and analyzing structural information from a number of travel related websites. Using Google Directory and Open Directory, 32 Hong Kong related travel guide websites were collected. The terms used in the structural information were collected for analysis. After grouping and filtering terms with similar meanings, seven main classes were defined in the travel ontology. Each class had its own subclasses and properties. Based on the travel ontology, around 400 instances were input. To demonstrate how the ontology works, Java and Jena Semantic Framework were employed to develop a semantic tourist information portal called SemTour:HK. A fuzzy searching function was developed in the accommodation page to allow users to find similar accommodation. The user study showed the fuzzy search function was useful for the user to reduce browsing time.

In conclusion, a new approach to modeling a domain ontology is proposed. A travel ontology was successfully modeled and a semantic web application was developed by using the modeled ontology. In addition, a fuzzy search function based on fuzzy ontology map was successfully developed. The feedback for this fuzzy search function has been quite positive.

Chapter 5. iJADE FreeWalker

5.1 Introduction

Recent developments in communication systems and Information Technology have had a profound influence on different industries. These technologies have changed customer behaviors. For example, more and more tourists now gather information from the Internet and plan their own tours. There are some applications such as personalized pricing (priceline.com) and recommendation systems for tourists. However, these applications are only useful for tourists planning their travels. Moreover, there are many wireless devices such as PDA and mobile phone on the market. These mobile devices are small, lightweight and long lasting. It is a good idea for the tourists to use their mobile device as a personal tourist guide. In this chapter, a new mobile application specialized for tourists, iJADE FreeWalker (iJFW), is presented.

iJADE FreeWalker (Ao Ieong et al., 2005, Lam and Lee, 2006) is developed based on the intelligent agent-based platform, iJADE[7]. A number of systems have already been developed based on iJADE such as an authentication system (Lee, 2002), a surveillance system (Lee, 2003), a web mining system (Lee and Liu, 2004a), a stock advisor system (Lee, 2004) and a weather forecasting system (Lee and Liu, 2004b). iJFW is a location-aware application which utilizes the Global Positioning System (GPS) to gather geographic information. iJFW not only provides positioning information, it is also integrated with

[7] http://www.ijadk.org

different intelligent agents to enhance the functionality of the guidance system. Furthermore, the system adopts the fuzzy ontology map to provide recommendations to users according to the their preferences.

5.1.1 Organization of the Chapter

The rest of this chapter describes the iJADE FreeWalker (iJFW) in detail. Section 5.2 describes background information about iJFW. Section 5.3 describes the system architecture of iJFW. Section 5.4 shows the fuzzy ontology map in iJFW. Section 5.5 shows the experimental results. Section 5.6 presents a brief conclusion and future work.

5.2 Background

iJFW is an intelligent agent-based tourist guide system based on the integration of several cutting-edge technologies including ontology agents technology and the Global Positioning System (GPS) technology. iJFW is developed based on iJADE. In this section, the background information about iJFW is presented. Section 5.2.1 shows an overview of agent technology. Section 5.2.2 presents the system architecture of iJADE. Section 5.2.3 shows information about the Global Positioning System.

5.2.1 Agent Technology

An agent is a complex software entity which is situated in an environment to achieve goals for a user (Jennings, Sycara and Wooldridge, 1998). It can perceive and response upon its environment autonomously which its behavior depends partially on its own experience. Professor Michael Wooldridge defines agent is a computer system that is capable of independent action on behalf of its user or owner. (Wooldridge , 2002) Unlike an arbitrary program, the agent is goal-orientated, persistent, reacts to the environment and is autonomous (Franklin and Graesser, 1996). There are different types of agents such as

1. Symbolic reasoning agents / deliberative agents – contain an explicitly represented, symbolic model of the world which can make decisions by symbolic reasoning
2. Reactive agents – do not have any internal symbolic models, act by stimulus-response to the current state of the environment.
3. Hybrid agents – combine the best of reasoning and reactive agents
4. Mobile agents – migrate from one machine to another and execute in a platform-independent environments
5. Information agents – manage and manipulate the information from distributed sources. It can be mobile or static.

Recently, scientists tried to build intelligent agents that can mimic human intellectual behavior for the purposes of problem solving, scheduling, data mining and to generally assist humans in all of their activities. The developers of agents have implemented various multi-agent systems such as planning and scheduling (Pechoucek et al., 2006). Compared with traditional client-server technologies and code-on-demand technologies, systems which are developed by using agent technology have the following advantages (Lange and Oshima, 1999):

1. Reduce network load – Traditional distributed systems mainly rely on the communication protocols involving multiple interactions to complete a task. This leads to a lot of network traffic. It is possible for users to package conversation with an agent and dispatch it to a destination host. Using such approaches could greatly reduce the network load.
2. Reduce network latency – Real time responses are critical for control systems. Network latencies have a great impact on control systems. By using agent technology, we can

dispatch autonomous agents to the controller of the system. The agent executes commands directly with respect to the external environments.

3. Asynchronous process – Agents can be dispatched into different hosts throughout the network. After dispatch, the agents are independent and can operate asynchronously and autonomously.
4. Heterogeneous – Agents work on their execution environments. This enables the agents to work on different hardware and software configurations. It provides a seamless environment for heterogeneous system integration.

5.2.2 Intelligent Java Agent-based Development Environment (iJADE)

iJADE is an intelligent agent-based development environment which is able to develop a fully integrated intelligent multi-agent based system. It is a basic framework and development environment for intelligent agent-based applications. iJADE is a contemporary conceptual model for intelligent agents. It consists of four layers (see Figure 5.1):

1. Application Layer: This is the uppermost layer that consists of different intelligent agent-based applications. This layer accepts the data result from the conscious layer and is connected to external applications.
2. Conscious Layer: This is an intelligent layer that includes a Sensory Area, Logic Reasoning Area and Analytical Area.
3. Technology Layer: This layer provides all the necessary mobile agent implementation APIs for the development of intelligent agent components in the 'Conscious Layer'.
4. Supporting Layer: This layer provides a programming language and protocols to support the development of the ‘Technology Layer’.

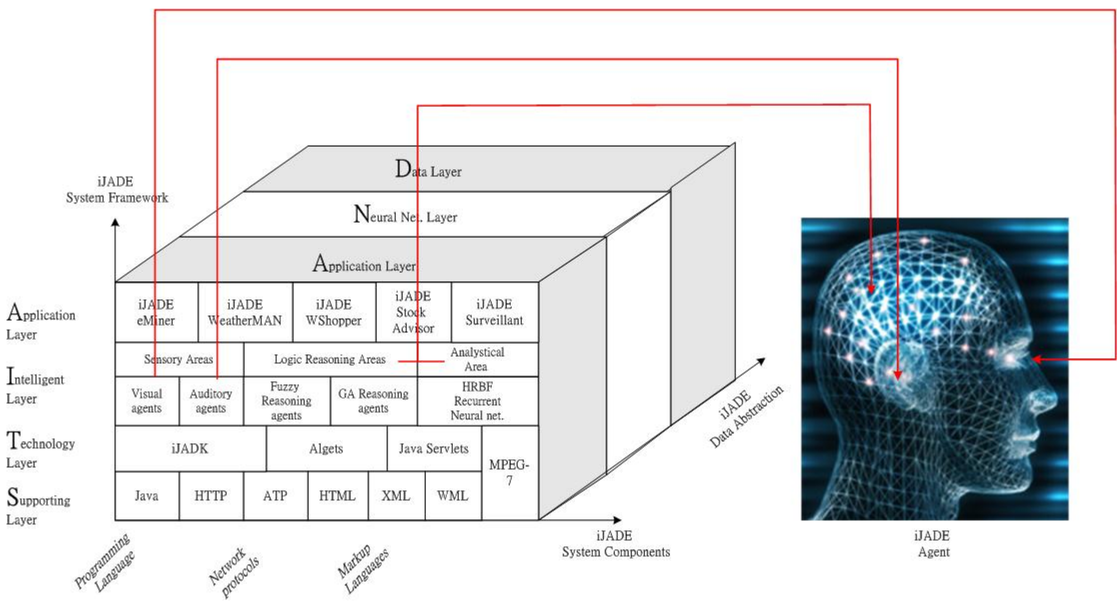


Figure 5.1. iJADE Framework

iJFW is an agent-based tourist guiding system which is developed based on this iJADE platform. For details about iJFW, please refer to Section 5.3.

5.2.3 Global Positioning System (GPS)

GPS is used to gather geographical information. GPS receivers are not expensive and are widely used. There are some consumer products developed for outdoor usage such as hiking, flying and sailing. Furthermore, the receivers can also be used for route guiding. The GPS is adopted to develop our tourist guiding system as it is convenient to use and accurate. Using GPS allows the identification of the coordinates of the user's position.

GPS was designed by the United States Department of Defense in 1978. At first, GPS was used for military positioning and navigation. In 1984, due to the crash of a Korean Airlines fight, part of GPS was made available for public use. In 1995, the system was operational with degraded accuracy (around 100 meters). In 2000, the system was operational with an accuracy of around 15m. This spurred the development of the GPS applications[8].

However, since the US Military have the right to turn off the GPS signal on a regional basis at any time, it would greatly affected the performance of the GPS applications such as car navigation system, tracking, traffic and fleet management system. As a result, another position system called GALILEO was begun[9]. GALIEO is a global navigation infrastructure under civil control. It consists of 30 satellites, 6 more than the GPS. The main aim of GALIEO is to provide an open service with the same performance and service guarantees as GPS.

At present, GPS devices are quite common. Some mobile devices already have a build-in GPS receiver. But GPS has some shortcomings when receiving satellite signals. Since the satellites are located at a high earth orbit, the transmitted signal is very weak when received on earth. When the GPS receivers are located under cover of bridges, or between tall buildings, the performance is greatly reduced. In addition, the performance of the receiver depends on the weather and GPS does not work inside buildings. In iJFW, GPS technology is adopted to receive the user's geographical information. The system provides location-aware information for the users. In Section 5.3, the system architecture about iJFW is presented.

5.3 iJADE FreeWalker (iJFW)

iJADE FreeWalker (iJFW) integrates agent technology and ontology to form an intelligent tourist guide system. In general, location awareness means the execution of services can be dynamically adapted depending on a user's current location. Location-aware mobile tourist guide systems have been designed using two major approaches. One of the approaches is a

[8] Office of the Press Secretary, The White House – http://www.navcen.uscg.gov/news/archieve/2000/may/SA.htm

[9] GALILEO – European Satellite Navigation System, http://europa.eu.int/comm/dgs/energy_transport/galileo/index_en.htm

client-server communication model which uses a remote procedure call (RPC) technique to transmit the data. The second one uses agent technology.

In a client-server communication approach, two separate computers communicate with each other over the network by sending and receiving messages. These messages are either requests from clients or responses from a server. By using this approach, the network traffic is quite large (Karnik and Tripathi, 1998) since the responses often contain a large volume of data.

Agent technology has three advantages over the client-server communication approach. First, agents can transmit over the Internet with a varying degree of autonomy. In a heterogeneous network environment, the agent can migrate freely as it performs tasks on behalf of users. Second, agents can perform assigned tasks for users even when the user is off-line. Third, agents reduce network traffic and communication delay significantly as it is not necessary to transfer large amounts of data over the network. Hence, the network bandwidth is utilized in a more effective way.

Mobile devices, such as mobile phones and PDA, have limited storage, bandwidth and calculation power, making agents suitable for handheld devices. To ensure the usability of the system, the proposed tourist guiding system was developed under an agent platform. The location awareness agent was developed. The agent captures the geographical information by using the GPS system. An ontology-based context model was proposed to represent tourist information. The context model enables knowledge sharing and context reasoning in the tourist domain.

iJADE FreeWalker is composed of four major components: 1) iJADE FreeWalker Client, 2) GPS Agent, 3) Fuzzy Search Agent, and 4) iJADE Tourist Information Center. Figure 5.2 shows the system diagram of iJADE FreeWalker.

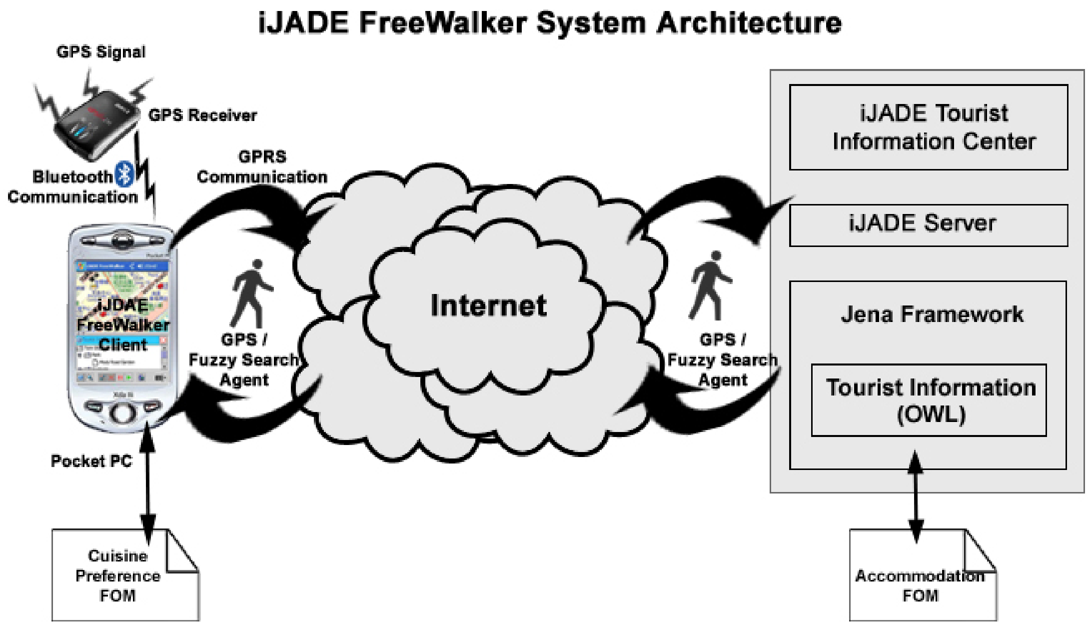


Figure 5.2. iJADE FreeWalker System Architecture

5.3.1 iJADE FreeWalker Client

The iJADE FreeWalker Client is a graphical user interface for displaying maps and tourist information for users. The client gathers the user's location information by using the GPS receiver. The GPS receiver is employed to receive simultaneous GPS data and to ascertain the user's location. The GPS receiver is connected with a pocket PC via Bluetooth. Figure 5.3 shows a screenshot from the iJADE FreeWalker Client.

Figure 5.3. Screenshot from iJADE FreeWalker Client

5.3.2 GPS Agent

Mobile devices have a narrow bandwidth in wireless connection. This is a critical problem in developing a mobile information retrieval system. To overcome such problem, agent technology is utilized in iJADE FreeWalker. In a single request, the agent can conduct multiple interactions with different information database systems. The results are then sent back to the device so as to reduce the network load.

The GPS Agent is an intelligent agent which can freely migrate from one host to another. First, the GPS Agent captures the user location information from a GPS receiver. Then, it migrates from the client (end-user handheld device) to a remote iJADE Tourist Information Center through GPRS communication. When the GPS Agent reaches the information center, it uses SPAQRL to query the server to collect the tourist information with respect to the user's geographical information. Finally, the GPS Agent returns to the client with related tourist information. The client collects the information from the GPS Agent and shows the context information to the user.

5.3.3 Fuzzy Search Agent

iJFW has two fuzzy search functions to help users find their accommodation and dining information. There are two fuzzy ontology maps in iJFW. The cuisine preference FOM is located in the client side (Pocket PC) and the accommodation FOM is located in the sever side (Tourist Information Center). The Fuzzy Search Agent collects the search criteria which is entered by the user and is then dispatched to the server side to search for related information. Section 5.4 shows details of the fuzzy search function.

5.3.4 iJADE Tourist Information Center

The Tourist Information Centre has two core components: an iJADE server and a Jena Framework (Carroll et al., 2004). Jena is an open source Java framework for building Semantic Web applications. It offers a number of APIs for handling RDF, RDFS and OWL. The main purpose of using Jena is to parse and query the travel ontology. Jena searches the data from the travel ontology by using the SPARQL Protocol and RDF Query Language (SPARQL). Figure 5.4 shows an example of SPARQL that is used to get information about Guesthouses. The search returns as an RDF graph and sends back to the tourist information server.

iJADE Server acts as a communication platform. It is a container for receiving and sending GPS Agents through GPRS. When the GPS Agent arrives at the information center, it uses SPARQL statements to parse OWL. The related tourist information will be sent back to client. After processing, the client displays the tourist information to the user.

```
# Select related information about Guesthouse
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX travel: <http://www.comp.polyu.edu.hk/~cshwlam/ontology/travel.owl#>
SELECT ?URI ?NAME ?ADDRESS ?DISTRICT ?ROOM ?URL ?EMAIL ?PRICE ?FAX ?PHONE
WHERE { ?URI rdf:type travel:Guesthouse .
?URI travel:hasName ?NAME .
?URI travel:hasAddress ?ADDRESS .
?URI travel:hasDistrict ?DISTRICTURI .
?DISTRICTURI travel:districtName ?DISTRICT .
?URI travel:hasRoom ?ROOM .
OPTIONAL {?URI travel:hasURL ?URL } .
OPTIONAL {?URI travel:hasEmailAddress ?EMAIL }.
OPTIONAL {?URI travel:hasStandardRoomPrice ?PRICE } .
OPTIONAL {?URI travel:hasFaxNumber ?FAX }.
OPTIONAL {?URI travel:hasPhoneNumber ?PHONE}
}
```

Figure 5.4. A SPARQL example used in iJADE FreeWalker

5.4 Fuzzy Ontology Map in iJADE FreeWalker

iJFW has two fuzzy search functions for the user: 1) Accommodation Fuzzy Search and 2) Cuisine Preference Search. These fuzzy search functions use the Fuzzy Ontology Map (FOM).

5.4.1 Accommodation Fuzzy Search

Accommodation fuzzy search helps the user find similar accommodation. This function is the same as the fuzzy search function in SemTour:HK (Chapter 4). Figure 5.5 shows a screenshot of the questionnaire-like form for users to proceed to fuzzy searching. The form contains two fields. The first is the relation from and the other is the minimum level. The relation from field is used to let the user select the accommodation type that he/she would like to search for such as Guesthouse, Hotel, Holiday Camp. The minimum level lets the user filter the search result. five levels were defined: Similar (1), Very Similar (0.8), More or Less Similar (0.5), Less Similar (0.2) and Not Similar (0).

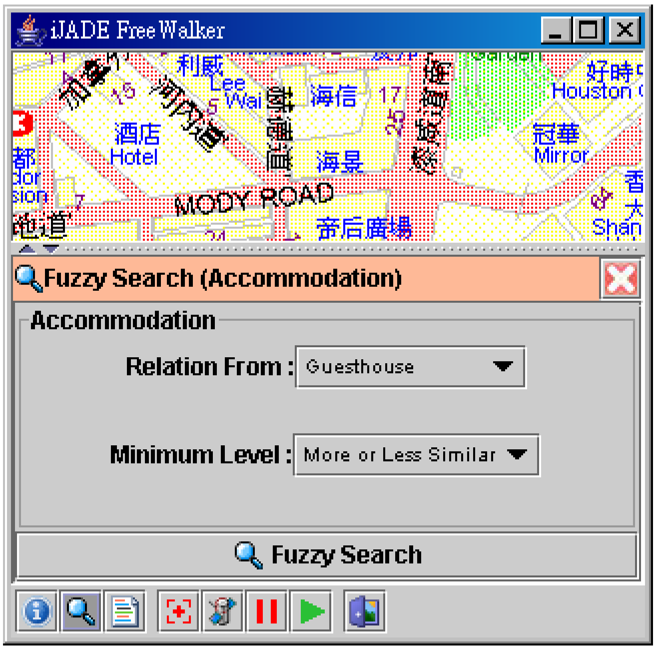


Figure 5.5. Screenshot of accommodation fuzzy searching in iJFW

After users select the search criteria and click the “Fuzzy Search” button, a Fuzzy Search Agent is created. This agent, with the users’ search criteria, dispatches itself to the Tourist Information Server. After the agent arrives, the server locates the accommodation fuzzy ontology map for searching. Then, the server tries to locate the relation information in the Accommodation FOM. After this, the server queries the database. The Fuzzy Search agent collects the results, dispatches it back to the client and displays the information to the user. Figure 5.6 shows a screenshot of the accommodation fuzzy search.

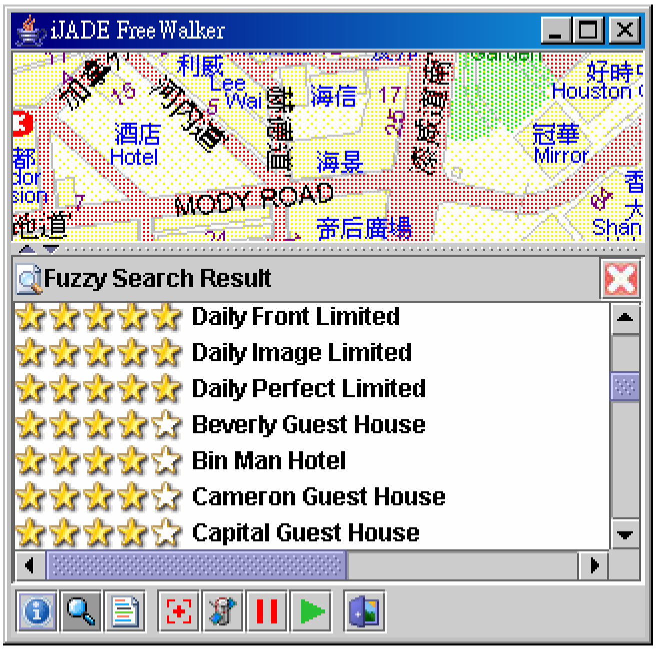


Figure 5.6. Screenshot of accommodation fuzzy search result in iJFW

5.4.2 Cuisine Preference Search

Cuisine Preference Search is another function which adopts FOM for searching. The purpose of this function is to help users find their favorite restaurant. Figure 5.7 shows a screenshot of the Cuisine Preference Search. There are four types of cuisine that users can select: Asian, Chinese, Other, and Western. The users can select different levels of preference for each cuisine. There are five levels for preference: much more than average, more than average, average, less than average and much less than average. The fuzzy membership for these preferences are 1 (much more than average), 0.8 (more than average), 0.5 (average), 0.2 (less than average) and 0 (much less than average) respectively.

Figure 5.7. Screenshot of cuisine preference panel in iJFW

After the user selects the preferences and presses the “Save Preference” button, a fuzzy ontology map is created and stored for the user. After updating the preference information, there is a change in the information panel. The dining information in the information panel is sorted according to the users’ preference. Figure 5.8 shows a screenshot of the information panel when the user has selected “much more than average” in Asian Cuisine, “More than average” in Chinese Cuisine and “average” for others. Other than the change of information panel, the user can carry out a fuzzy search by using the preference FOM. After selecting the preference and pressing the “Fuzzy Search” button, a Fuzzy Search agent is created. Again, this agent, with preference FOM, is dispatched to the Information Server to retrieve the dining information for the user. Unlike the accommodation fuzzy search, instead of using server side FOM, the Fuzzy Search agent searches using the client’s FOM. Therefore, every user can have his/her own preference FOM and carry out customized fuzzy searches. Figure 5.9 shows the search result screen by using user’s preference FOM.

Figure 5.8. The sorted dining information

Figure 5.9. Screenshot of dining search result in iJFW

5.5 Performance Evaluation

This section describes the experiments which were used to determine the performance of iJFW. O2 XDAII Pocket PC with Java Virtual Machine was used as the mobile device. The GPS receiver is a Holux GR-230 model. The pocket PC and the GPS receiver are connected using Bluetooth. The iJADE platform and the iJADE Tourist Guide are installed in the Pocket PC. Figure 5.10 shows the iJADE FreeWalker running in the mobile device. The tourist information center is installed in a P4 1.8GHz computer with 768MB memory.

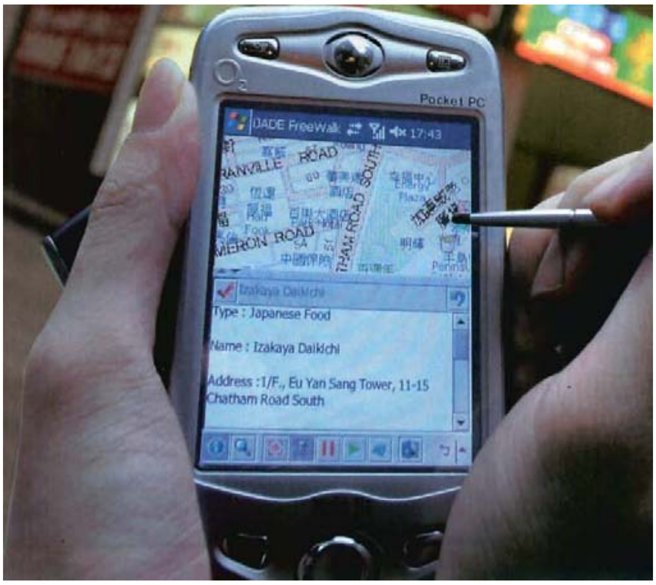


Figure 5.10. iJADE FreeWalker

Experiment I: Response Time for gathering location-aware information

This experiment seeks to evaluate the efficiency of the GPS Agent in iJFW in terms of the whole round trip time (RTT) of the GPS Agent. The RTT is the total time that the GPS Agent takes to travel to the Information Center with the users' GPS information, get the location-aware information such as shopping and dining information from the center, return to the client and display the results to the user.

This test used two different setups. Setup One situated iJFW client and the iJADE Tourist Information Center within the same LAN. Setup Two connects iJFW client to the iJADE Tourist Information Center by using a 36.6kbit/s wireless network connection. Table 5.1 shows results of the RTT after 50 trials for each setup.

Table 5.1. Round trip time for GPS Agent to gather location-aware information

Time (ms)	Setup One. Within Same LAN	Setup Two. 36.6k Wireless Connection
Min RTT	238	8800
Max RTT	475	9948
Average RTT	393.84	8920.68

The average response time for location-aware information for Setup One is around 0.4 seconds and for Setup Two is around 9 seconds. The response time is slow when the iJFW client is connected to the internet using a wireless network, since the speed of the wireless network is only 36.6kbit/s. This is one of the main factors affecting the response time. In addition, the outdoor environment (weather and buildings) may affect the wireless network signal and hence the results of the experiment. One possible way to decrease the response time is to use a higher speed wireless network such as 3G.

The average response time for GPS Agent to gather geographical information is 393.84ms (within the same LAN) and 8920.68ms (36.6k wireless connection). Suppose the time used for sending data to and from the server is zero if the Pocket PC and the server placed within the same LAN. Thus, 393.84ms is the time used for processing in the server. As a result, in 36.6k wireless connection, the time used for sending data to and from the server is 8920.68 - 393.84 = 8526.84ms = 8.53s. 3G technology has a transmission speed at 2Mbit/s. Proportionally, in 3G wireless connection, the time used for sending data to and from the server is around (36.6 * 1000 * 8.5) / (2 * 1000 * 1000) ~ 0.16s. The transmission response time would be decreased by 98% proportionally if we adopted 3G technology for the experiments.

Experiment II: Response Time for fuzzy searching

This experiment evaluates the performance of the Fuzzy Search Agent in iJFW. As in Experiment I, it records the whole round trip time (RTT) of the Fuzzy Search Agent. Two experimental setups are also adopted in this experiment. For the searching criteria of accommodation fuzzy search, the value of relation from is guesthouse and the value of the minimum level is more or less similar. For the searching criteria of cuisine fuzzy search, the value of Asian cuisine is much more than average, the value of Chinese cuisine is more than the value of the other cuisine and Western cuisine is average. Results of the RTT for cuisine searching and accommodation searching after 50 trials for each setup are shown in Table 5.2 and Table 5.3 respectively.

Table 5.2. Round trip time for Fuzzy Search Agent to gather dining information

Time (ms)	**Setup One Within Same LAN**	**Setup Two. 36.6k Wireless Connection**
Min RTT	2188	25664
Max RTT	2597	30314
Average RTT	2420.6	26240.7

Table 5.3. Round trip time for Fuzzy Search Agent to gather accommodation information

Time (ms)	**Setup One. Within Same LAN**	**Setup Two. 36.6k Wireless Connection**
Min RTT	2959	22110
Max RTT	3556	36003
Average RTT	3243.5	24212.4

The average response time for cuisine searching for Setup One and Setup Two is 2.4 seconds and 26.2 seconds. The average response time for accommodation searching for Setup One and Setup Two is 3.4 seconds and 24.2 seconds. When the client is connected to the server with 36.6k wireless network, it leads to longer response times. As mentioned before, one of the factors is the slow wireless network connection speed. It could be improved if the experiments adopted a higher speed network.

In this experiment, the number of search result items is limited to 50. When the Fuzzy Search Agent dispatches the results from the information centre to the client, it may require a long time due to the large result data size. Another possible way to decrease the response time is to limit the number of search result items to a much smaller amount such as 10 or 20.

Experiment III. Precision test

This test evaluates the precision of received geographical information. In our proposed system, GPS coordinates are used to locate the position of user and nearby landmarks. 30 candidates were invited to use the system and then ask these them to do a survey on the accuracy of the system. In this study, candidates are required to visit a number of places in Tsim Sha Tsui (Hong Kong) by using the iJADE FreeWalker. During the test, the system is connected to the GPS receiver to receive real time GPS location, and then gathers context tourist information by agents. The results are tabulated in Table 5.4.

17 candidates felt that the location-aware tourist information shown in iJFW is accurate. 6 candidates failed to receive the GPS data, and 7 candidates thought there was error in the result. The failed cases were analyzed and concluded that there is a variation in receiving the valid GPS data for two reasons: 1) candidates used the GPS receiver improperly and 2) the poor weather conditions. The performance of the GPS receiver depended on the weather and outdoor environment. The Author believes that a new generation of GPS receiver could overcome these problems. 57% of candidates agreed that the system provided them with accurate location aware tourist information by using the iJFW.

Table 5.4: Accuracy of the iJADE Tourist Guide System with real GPS receiver

Do you think the iJADE Tourist Guide system is accurate?	**Number of candidates**
Very accurate	2
Accurate	10
Normal	5
inaccurate	7
Very inaccurate	6

Experiment IV. Usability test

This test evaluated the effectiveness of iJFW. After the previous test, 30 candidates were given a questionnaire to ask whether the system is suitable for tourists. 17 candidates thought that iJFW could replace tourist guidebooks. 13 candidates thought that iJFW would be a subsidiary tool for traveling. The results showed that most candidates agreed the system can help tourists find tourism-related information. The system was able to receive the categorized and location-aware tourist information according to their geographical information. 20 of the candidates felt that the fuzzy search function is useful for finding a particular cuisine. However, most of the users (23 candidates) felt that the accommodation search function is useful. One of the main reasons is that the users did not understand the purpose of this function. It would be better to have some explanation of the fuzzy search function in the future iJFW.

For the ontology in iJFW, users could find out the information they want by using the categorization tree (ontology) in the information panel. 25 candidates thought the categorization very detailed and 5 candidates thought the information not precise enough. 5 candidates suggested that the system should provide detailed information (i.e. coupon, discount) for each restaurant and shop. Since iJADE FreeWalker can run on handheld devices (i.e. palm, PPC), all candidates agreed that the system supports a high degree of mobility and flexibility.

5.6 Conclusion and Future work

This chapter has described how to develop an agent-based location-aware tourist guide system – iJADE FreeWalker (iJFW). iJFW was developed based on the intelligent Java Agent Development Environment (iJADE). iJFW adopted a number of Semantic Web technologies such as an ontology and agents. In addition, it applied our proposed fuzzy ontology map to assist users in finding dining and accommodation information.

iJFW has four main components: a iJFW client, a GPS Agent, a Fuzzy Search Agent and a Tourist Information Center. The prototype of iJFW was developed based on a JAVA program. The client of iJFW was deployed on a O2 XDAII Pocket PC with Java Virtual Machine and iJADE platform. The iJFW client was connected to the GPS receiver to gather the geographical information by using Bluetooth. The tourist information center is installed in a P4 1.8GHz computer with 768MB memory. The Tourist Information Center employed a Jena framework for parsing and querying the tourist database which is in OWL format.

A number of tests were done for performance evaluation. The average response time for gathering the location-aware tourist information is around 0.3 seconds (within the same LAN) and 9 seconds (36.6kbit/s wireless network connection). The average response time for fuzzy searching is around 3 seconds (within the same LAN) and 25 seconds (36.6kbit/s wireless network connection). When the iJFW Client connected to the Tourist Information Center with 36.6kbit/s wireless network connection, the response time was quite long. There are a number of factors that affect the response time.

In the experiments, the wireless connection for the iJFW is 36.6kbit/s which is quite slow. To further decrease the response time, the iJFW will adopt a higher speed wireless connection network such as Enhanced GPRS[10] (EGPRS) and third-generation (3G[11]) technology. The bandwidth for EDGE can be up to 236.8kbit/s and 3G can be up to 2Mbit/s. Proportionally, the average processing time for gathering location-aware information for EDGE and 3G would be 1.6 seconds and 0.45 second respectively. The average response time would be greatly reduced by using these wireless technologies. Another possible way to reduce the response time is to limit the number of fuzzy search results to a smaller amount such as 20 or 10.

This chapter showed how to develop a tourist guide system in a mobile device. The system adopted a number of Semantic Web technologies such as Agent and Ontology and integrated a fuzzy ontology map with iJFW to support fuzzy searching for the user. The user acceptance tests showed that most of the users made positive comments on iJFW. They felt that the system was innovative and useful for tourists. Since the iJFW is still in the preliminary development stage, its functionalities are very limited. According to user comment, the Author summarized the suggestions for future enhancement:

1. Voice interface - To extend the usability and interaction, a voice interface would be used as the input for the system. The system uses Automatic Speech Recognition as the communication tool to mimic a human-like tourist guide. Users can chat with the tourist guide and query the relevant tourist information by using a voice interface.
2. Learn user preferences – To further increase the usability and functionality of the system, a Neural Network (NN) module would be added to learn users' preferences as to the tourist information.
3. Route recommendation – To optimize the tourist route from a set of itinerary with time and money constraints.
4. Fuzzy searching – To provide more fuzzy search functions which can show the

[10] EDGE – high-speed data in GSM/GPRS networks, http://www.ericsson.com/technology/whitepapers/edge_wp_technical.pdf
[11] Telecom Resources – 3G, http://www.freewebs.com/telecomm/3g.html

customized tourist information based on the user's criteria/preferences.

In conclusion, a mobile ontology-based location-aware agent-based tourist guide system – iJADE FreeWalker has been successfully developed. This integrates GPS and agent technology to provide a location-aware tourist information retrieval system. It provides well-organized tourist information by using ontology. In addition, it supports fuzzy searching for the users by using a fuzzy ontology map. Tourists can access the nearby tourist information anywhere and anytime using a small handheld device with limited computational power and limited network bandwidth. In the future, the Author will further improve the system with features such as using voice interface, learning user preferences and route recommendation.

Chapter 6. Conclusion and Further Research

6.1 Organization of the Chapter

This chapter concludes the thesis and is organized in two sections. Section 6.2 summarizes the major contributions of the thesis. Section 6.3 suggests possible further research on FOMs.

6.2 Summary of Major Contributions

A fuzzy ontology map is proposed for extending the current hard semantics ontology. The goals of this work are i) to define a way to utilize the structure of ontology and the fuzzy relationships within them, ii) to show how the fuzzy relation inference can be integrated into applications.

The major contributions of this thesis can be summarized in three areas: i) Fuzzy Ontology Map – an extension of the crisp ontology, ii) Fuzzy Relation Inference Algorithms, iii) Prototypes of application.

I. Fuzzy Ontology Map (Chapter 3)

A Fuzzy Ontology Map is proposed for the construction of the fuzzy ontology. Since current ontology markup languages are based on the bivalent logic/first order logic, they are not capable of modeling uncertain and imprecise information. A Fuzzy Ontology Map (FOM), an XML format document, is proposed to extend the crisp ontology. The ontology is treated as a graph. Each path in the ontology graph would include fuzzy information (fuzzy membership values). A FOM is a connection matrix in which each matrix entry is the fuzzy membership value of the path in the ontology graph. It is possible to create a fuzzy ontology by using an FOM with ontology markup languages such as RDF and OWL.

II. Fuzzy Relation Inference Algorithms (Chapter 3)

A set of algorithms for fuzzy relation inference are proposed for the FOM. Two types of fuzzy set were defined: *symmetric* and *asymmetric*. A symmetric fuzzy set is one where the fuzzy relation is non-directional between classes in the ontology. For example, the level of similarity is symmetric as there is no difference between class A similar to class B and class B similar to class A (non-directional). An asymmetric fuzzy set is one where the fuzzy relation is directional between classes in the ontology. For example, the level of preference is asymmetric since there is a difference between class A is preferred to class B (A>B) and class B is preferred to class A (B>A) (directional). By using the fuzzy relation inference algorithms, if two classes in the ontology are not directly connected, the fuzzy membership value would be calculated by the paths that connect them in an FOM.

III. Prototypes of applications (Chapter 4 and 5)

Two applications prototypes were developed to demonstrate how the fuzzy ontology works. These prototypes are SemTour:HK and IJADE FreeWalker (iJFW).

A prototype, SemTour:HK, was developed to demonstrate how an FOM works in a web-based tourist information portal. It shows how to fuzzy search using an FOM (Chapter 4). 20 interviewees were invited to do a user acceptance test. 80% of the interviewees agreed that the system can help tourists find tourism related information in Hong Kong. 75% of them agreed that the fuzzy search function is useful for finding accommodation.

iJADE FreeWalker (iJFW) is a context-aware agent-based tourist guiding system for mobile devices (Chapter 5). The average response time for gathering location-aware tourist information is around 0.3 seconds (within the same LAN) and 9 seconds (36.6kbit/s wireless network connection). A cuisine fuzzy searching was implemented. 20 of the 30 candidates felt that the fuzzy search function is useful to find their cuisine of choice. The average response time for fuzzy searching is around 3 seconds (within the same LAN) and 25 seconds (36.6kbit/s wireless network connection).

6.3 Advantage of using FOM

As mentioned in Section 1.1 Motivation of Research, the current two-valued-based logical methods in Semantic Web are not able to process soft semantics. The work described in this thesis is about developing an extension of the current ontology representation which supports uncertain information modeling. The ontology markup language was extended so that each path in the ontology graph could include fuzzy information. From the experimental results of the prototypes, most users felt that the fuzzy search function was useful for the user to reduce the browsing time. It provides a more convenient way for searching. In addition, by using FOM, it can help to get the right search result in a short period of time. An extension of the crisp ontology was achieved in this work.

6.4 Further Research

There are two directions for further enhancement of this work. One direction is the automatic generation of fuzzy ontologies. The other is to further extend or modify existing query languages to support fuzzy queries.

6.4.1 Automatic Fuzzy Ontology Generation

In this thesis, the fuzzy ontology is created by the ontology developer. The fuzzy relation between classes depends on the view of the developer. Since different people may have different views of an ontology, there is a limitation that the fuzzy ontology map only represents the developer's view. The fuzzy ontology in the current proposal is quite subjective. In addition, constructing a fuzzy ontology for a certain domain is difficult. It is a good direction to create the fuzzy ontology in an automatic approach. By doing so, the created ontology is more objective. In addition, it could shorten the construction time. There are a number of works about ontology generation from various data types such as textual data (Maedche and Staab, 2001, Navigli, Velardi and Gangemi, 2003), dictionary (Morin, 1999) and relational schemata (Rubin et al., 2002, Stojanovic, Stojanovic and Volz, 2002). Tho and his researchers proposed an idea to create a fuzzy ontology automatically using fuzzy conceptual clustering (Tho et al., 2006). Since there is still little work on fuzzy ontology construction, this is one of the core further research directions in this area.

6.4.2 Fuzzy Query Language

In this thesis, a Fuzzy Ontology Map is proposed for creating fuzzy ontology. Using a fuzzy ontology allows fuzzy searching. The results of the fuzzy search are ranked according to the fuzzy membership value in an FOM. The results are sorted by membership values. The higher the membership value (nearer to 1), the better the result. The results are filtered by a threshold (minimum level of membership value). In the implementation, the related classes are first selected and sorted. Then each related class is queried using SPARQL. Finally, all the search results are combined and returned to the user. The processing time of this approach is quite long since it must submit a number of queries to the server. To further improve the efficiency of the system, it is better to extend or modify the current ontology query language to support fuzzy queries. There are a number of research work in fuzzy query language for relational databases. For example, SQLf (Bosc and Pivert, 1995) and FQUERY97 (Zadrozny and Kacprzyk, 1998).

As mentioned before, there has been little work done on fuzzy ontologies. Thus, findings about query language for fuzzy ontology are also limited. To achieve the ultimate goal of creating an intelligent question-and-answer or Web IQ system, it is necessary to develop a fuzzy query language for fuzzy ontology. In the future, based on the work of this thesis, the Author will further research these two interesting areas.

References

Abowd, G. D., Atkeson, C. G., Hong, J., Long, S., Kooper, R. and Pinkerton, M. (1997). Cyberguide: A mobile context-aware tour guide, *ACM Wireless Networks*, Vol. 3(5), pp. 421-433

Antoniou, G., and van Harmelen, F. (2004). A Semantic Web Primer. MIT Press.

Baader, F., Calvanese, D., McGuinness, D. Nardi, D., and Patel-Schenider, P. (2002). The Description Logic Handbook: Theory, Implementation and Applications. Cambridge University Press.

Berners-Lee, T., Hendler. J., and Lassila, O. (2001). The Semantic Web, Scientific American, pp.34-43.

Bosc, P. and Pivert, O. (1995). SQLf: A relational database language for fuzzy querying, IEEE Transactions on Fuzzy Systems, Vol. 3(1), pp.1-17.

Brin, S. and Page, L. (1998). The Anatomy of a Large-Scale Hypertextual Web Search Engine, Computer Networks, Vol. 30, pp.107-117

Broekstra, J., Klein, M., Decker, S., Fensel, D., van Harmelen, F., and Horrocks, I. (2001). Enabling knowledge representation on the Web by Extending RDF Schema, Proceedings of the 10th World Wide Web Conference (WWW10), Hong Kong, China, May 2001, pp. 467-478

Buche, P., Dervin, C., Haemmerle, O. and Thomopoulos, R. (2005), Fuzzy querying of incomplete, imprecise, and heterogeneously structured data in the relational model using ontologies and rules, IEEE Transactions on Fuzzy Systems, Vol. 13 (3), pp. 373-383

Carroll, J. J., Dickinson, I., Dollin, C., Reynolds, D., Seaborne, A. and Wilkinson, K (2004). Jena: Implementing the Semantic Web Recommendations, Proceedings of the 13th international World Wide Web conference (WWW 2004), New York, USA, May 2004, pp. 74-83

Davies, J., and Fensel, D. (2003). Towards the semantic web: ontology-driven knowledge management. Wiley.

Decker, S., Melnik, S., Van Harmelen, F., Fensel, D., Klein, M., Broekstra, J., Erdmann, M., and Horrocks, I. (2000). The Semantic Web: The Roles of XML and RDF, IEEE Internet Computing, Vol. 4(5), pp. 63-73.

Dey, D. and Sarkar, S. (1996). A probabilistic relational model and algebra, ACM Transactions on Database Systems, Vol. 21(3), pp. 339-369.

Dickerson, J.A. and Kosko, B. (1997). Fuzzy Engineering. Prentice Hall.

Ding, Z. and Peng, Y. (2004). A probabilistic extension to ontology language OWL, Proceedings of the 37th Hawaii International Conference on System Sciences (HICSS' 04), Track 4, Volume 4.

Fensel, D. Horrocks, I., van Harmelen, F., McGuinness, D., and Patel-Schneider, P. (2001).OIL: An Ontology Infrastructure for the Semantic Web, IEEE Intelligent Systems, Vol. 16(2), pp. 38-45.

FOAF (2005), FOAF vocabulary specification, http://xmlns.com/foaf/0.1/.

Franklin, S., and Graesser, A (1996). Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agents, Proceedings of the Third International Workshop on Agent Theories, Architecture and Languages, Springer-Verlag.

Fuhr, N. and Rolleke, T. (1997). A probabilistic relational algebra for the integration of information retrieval and database systems, ACM Transaction on Information Systems, Vol. 15(1), pp. 32-66.

Fukushige, Y. (2004). Representing Probabilistic Knowledge in the Semantic Web, Position paper for the W3C Workshop on Semantic Web for Life Sciences, Cambridge, MA, USA, Oct. 2004.

Ganter, B. and Wille, R. (1999). Formal Concept Analysis: Mathematical Foundations. Springer.

Giugno, R. and Lukasiewicz, T. (2002). P-SHOQ(D): A probabilistic extension of SHOQ(D) for probabilistic ontologies in the semantic web, INFSYS, Wien, Austria, Research Report 1843-02-06.

Golbeck, J. and Hendler, J. (2006) FilmTrust: movie recommendations using trust in web-based social networks, Proceedings of the 3rd IEEE Consumer Communications and Networking Conference (CCNC 2006), Las Vegas, Nevada, USA, Jan. 2006, Vol. 1, pp. 282- 286.

Gruber, T. (1993). What is an ontology ? http://www-ksl.stanford.edu/kst/what-is-an-ontology.html.

Handschuh, S., and Staab, S. (2003a). Annotation for the semantic web. IOS Press.

Handschuh, S., and Staab, S. (2003b). CREAM – Creating metadata for the semantic web, Computer Networks, Vol. 42, pp. 579-598.

Heinsohn, J. (1994). Probabilistic description logics, Proceedings of the 10th Conference on Uncertainty in Artificial Intelligence, Washington, USA, July 1994, pp. 311–318.

Horrocks, I. (2002). DAML+OIL: A description logic for the semantic web, IEEE Data Engineering Bulletin, Vol. 25(1), pp. 4-9.

Horrocks, I., and Sattle, U. (2001). Ontology Reasoning in the SHOQ(D) Description Logic. Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI 2001), Seattle, Washington, USA, Aug. 2001, pp. 199-204.

HTML (1999). HTML 4.01 Specification, http://www.w3.org/TR/html401/

Islam, M.Z., and Brankovic, L. (2004). A Framework for Privacy Preserving Data Mining, Proceedings of Australasian Workshop Data Mining and Web Intelligence (DMWI 2004), Dunedin, New Zealand, Jan. 2004, pp. 163-168.

Jaeger, M. (1994). Probabilistic reasoning in terminological logics, Proceedings of the KR-94, 4th International Conference on Principles of Knowledge Representation and Reasoning, Morgan Kaufmann, San Mateo, pp. 305–316.

Jannink, J., Mitra, P., Neuhold, E., Pichai, S., Studer, R., and Wiederhold, G. (1999). An algebra for semantic interoperation of semistructured data, Proceedings of IEEE Knowledge and Data Engineering Exchange Workshop (KDEX' 99), Chicago, Nov. 1999.

Jennings, N.R., and Wooldridge, M. (1998). Applications of Intelligent Agents, Agent technology: Foundations, applications and markets. Springer.

Jennings, N.R., Sycara K., and Wooldridge, M.J. (1998). A roadmap of agent research and development, Autonomous Agents and Multi-Agents Systems, Vol. 1(1), pp.7-38.

Karnik, N.M. and Tripathi, A.R. (1998). Design Issues in Mobile-Agent Programming Systems, IEEE Concurrency, Vol. 6(3), pp. 52-61.

Koller,D., Levy, A. and Pfeffer,A. (1997). P-CLASSIC: A tractable probabilistic description logic, Proceedings of the AAAI Fourteenth National Conference on Artificial Intelligence, Providence, Rhode Island, Aug. 1997, pp. 390–397.

Kosko, B. (1993). Fuzzy Thinking. Hyperion/Disney Books.

Lakshmanan, V.S., Leone, N., Ross, R. and Subrahmanian, V.S. (1997). ProbView: A flexible probabilistic database system, ACM Transactions on Database Systems, Vol. 22(3), pp. 419-469.

Lange, D.B. and Oshima, M. (1999). Seven good reasons for mobile agents, Communications of the ACM, Vol. 42(3), pp.88-89.

Lam, T.H.W. and Lee, R.S.T. (2006). iJADE FreeWalker - An Ontology-based Tourist Guiding System, Knowledge-Based Intelligent Information and Engineering Systems (LNCS 4252), pp. 644- 651.

Lee, C.S., Chen, Y.J. and Jian, Z.W. (2003). Ontology-based fuzzy event extraction agent for Chinese e-news summarization, Expert Systems With Applications, Vol. 25(3), pp. 431–447.

Lee, C.-S., Jiang, Z.-W., Hsieh, T.-C. (2006). A Genetic Fuzzy Agent Using Ontology Model for Meeting Scheduling System, Information Sciences, Vol. 176, no. 9, pp. 1131-1155.

Lee, C.-S., Jian, Z.-W., Huang, L.-K. (2005). A fuzzy ontology and its application to news summarization, IEEE Transactions on Systems, Man and Cybernetics (Part B), Vol. 35(5), pp. 859- 880.

Lee, R.S.T. (2002). iJADE Authenticator - An Intelligent Multi-agent based Facial Authentication System, International Journal of Pattern Recognition and Artificial Intelligence, Vol. 16(4), pp. 481-500.

Lee, R.S.T. (2003). iJADE Surveillant - An Intelligent Multi-resolution Composite Neuro-Oscillatory Agent-based Surveillance System, Pattern Recognition, Vol. 36, pp. 1425-1444.

Lee, R.S.T. (2004). iJADE Stock Advisor: An Intelligent Agent based Stock Prediction System using Hybrid RBF Recurrent Network, IEEE Transactions on Systems, Man and Cybernetics (Part A), Vol. 34(3), pp. 421-428.

Lee, R.S.T. (2005). Fuzzy-neuro approach to agent applications: From the AI perspective to modern ontology. Springer.

Lee, R.S.T. and Liu, J.N.K. (2004a). iJADE Web-miner : An Intelligent Agent Framework for Internet Shopping, IEEE Transactions on Knowledge and Data Engineering, Vol. 16(4), pp.461-473.

Lee, R.S.T. and Liu, J.N.K. (2004b). iJADE WeatherMAN: A Weather Forecasting System using Intelligent Multi-Agent Based Fuzzy-Neuro Network, IEEE Transactions on Systems, Man and Cybernetics (Part C), Vol. 34(3), pp. 369-377.

Lim, J. C. and Foo, S. (2003). Creating virtual exhibitions from an XML-based digital archive, Journal of Information Science, Vol. 29(3), pp. 143-157

Maedche, A. (2002). Ontology learning for the semantic web. Kluwer Academic Publishers.

Maedche, A. and Staab, S. (2001). Ontology learning for the Semantic Web, IEEE Intelligent Systems, Vol. 16(2), pp. 72-79.

Mazzierei, M. (2004). A fuzzy RDF semantics to represent trust metadata, Proceedings of the 1st Workshop on Semantic Web Applications and Perspectives (SWAP 2004), Ancona, Italy, Dec. 2004.

McGuinness, D.L., Fikes, R., Rice, J., and Wilder, S. (2000). The Chimaera Ontology Environment, Proceedings of the 17th National Conference on Artificial Intelligence (AAAI 2000), Texas, USA, July 2000, pp. 1123-1124.

Morin, E. (1999). Automatic acquisition of semantic relations between terms from technical corpora, Proceedings of the 5th International Congress on Terminology and Knowledge Extraction (TKE 1999), Innsbruck, Austria, Aug. 1999, pp. 268-278.

Navigli, R., Velardi, P. and Gangemi, A. (2003). Ontology learning and its application to automated terminology translation, IEEE Intelligent Systems, Vol. 18(1), pp.22-31.

OWL (2004). OWL Web Ontology Language Overview, http://www.w3.org/TR/owl-features

Patel, C., Supekar, K., Lee, Y., and Park, E.K. (2003). OntoKhoj: A semantic web portal for ontology searching, ranking and classification, Proceedings of the 5th ACM International Workshop on Web Information and Data Management, Louisiana, USA, Nov. 2003, pp. 58-61.

Patel-Schneider, P., Horrocks, I., and van Harmelen, F. (2002). Reviewing the Design of DAML+OIL: An Ontology Language for the Semantic Web, Proceedings of the 18th National Conference on Artificial Intelligence (AAAI 2002), Alberta, Canada, July 2002, pp. 792-797.

Patel, M., and Duke, M. (2004). Knowledge discovery in an agents environment, Proceedings of the First European Semantic Web Symposium (ESWS 2004), Crete, Greece, May 2004, pp. 121-136.

Pechoucek, M., Thompson, S.G., Baxter, J.W., Horn, G.S., Koen Kok Warmer, C., Kamphuis, R., Marik, V., Vrba, P., Hall, K.H., Maturana, F.P., Dorer, K., Calisti, M. (2006). Agents in Industry: The Best from the AAMAS 2005 Industry Track, IEEE Intelligent Systems, Vol. 21(2), pp. 86-95.

Peral, J. (1988). Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann.

Poslad, S., Laamanen, H., Malaka, R., Nick, A., Buckle, P. and Zipf, A. (2001). CRUMPET: creation of user-friendly mobile services personalized for tourism, Proceedings of 3G 2001-Second International Conference on 3G Mobile Communication Technologies, London, UK, pp. 62-74

RDF (2004). Resource Description Framework (RDF): Concepts and Abstract Syntax, http://www.w3.org/TR/rdf-concepts/

RDFS (2004). RDF Vocabulary Description Language: RDF Schema, http://www.w3.org/TR/rdf-schema/

Rubin, D.L., Hewett, M., Oliver, D.E., Klein, T.E. and Altman, R.B. (2002). Automating data acquisition into ontologies from pharmacogenetics relational data sources using declarative object definitions and XML, Proceedings of the 7th Pacific Symposium on Biocomputing, Hawaii, USA, Jan. 2002, pp. 88-99.

Sadeh, N., Chan, E., Shimazaki, Y. and Van, L. (2002). MyCampus: an agent-based environment for context-aware mobile services, Proceedings of AAMAS02 Workshop on Ubiquitous Agents on Embedded, Wearable, and Mobile Devices, Bologna, Italy, July 2002, available at: http://autonomousagents.org/ubiquitousagents/2002/papers/papers/29.pdf

Salton, G., and Buckley, C. (1998). Term-weighting approaches in automatic text retrieval,' Information Processing & Management, Vol. 24(5), pp. 513–523.

Schreiber, A.T., Dubbeldam, B., and Wielemaker, J. (2004). Ontology-based photo annotation, IEEE Intelligent Systems, Vol. 16(3), pp. 66-74.

Semantic Web (2001). W3C Semantic Web, http://www.w3.org/2001/sw/

Soo, V.W., Lee, C.Y., and Li, C.C. (2003). Automated semantic annotation and retrieval based on sharable ontology and case-based learning techniques. Proceedings of the 2003 Joint Conference on Digital Libraries, Texas, USA, May 2003, pp. 61-72.

Stoilos, G., Stamou, G., Tzouvaras, V., Pan, J.Z., Horrocks, I. (2005). Fuzzy OWL: Uncertainty and the Semantic Web, Proceedings of International Workshop OWL: Experiences and Directions, Galway, Ireland, Nov. 2005, available at: http://image.ntua.gr/papers/398.pdf

Stoilos, G.; Simou, N.; Stamou, G.; Kollias, S. (2006). Uncertainty and the Semantic Web, Intelligent Systems, Vol. 21 (5), pp. 84-87.

Stojanovic, L., Stojanovic, N. and Volz, R. (2002). Migrating Data-Intensive Web Sites into the Semantic Web, Proceedings of the ACM Symposium on Applied Computing (SAC), Madrid, Spain, March 2002, pp. 1100-1107.

Tho, Q.T., Hui, S.C., Fong, A.C.M., Cao,T.H. (2006). Automatic Fuzzy Ontology Generation for Semantic Web, IEEE Transactions on Knowledge and Data Engineering, Vol. 18(6), pp. 842- 856.

Velez, B., Weiss, R., Sheldon, M.A. and Gifford, D.K. (1997). Fast and effective query refinement, Proceedings of the 20th ACM Conference on Research and Development in Information Retrieval (SIGIR 97), Philadelphia, USA, July 1997, pp. 6-15.

Widuantoro, D.H., and Yen, J. (2001). A Fuzzy Ontology-Based Abstract Search Engine and Its User Studies, Proceedings of the 10th IEEE International Conference on Fuzzy Systems, Melbourne, Australia, Dec 2001, pp. 1291-1294.

Wooldridge, M. (2002). An Introduction to Multiagent Systems. Wiley.

XML (2004). Extensible Markup Language (XML) 1.1, http://www.w3.org/TR/xml11/

Zadeh, L. (1965), Fuzzy sets, Information and Control, Vol. 8, pp. 338-353

Zadeh, L. (1975), Fuzzy logic and approximate reasoning, Synthese, Vol. 30, pp. 407-428

Zadeh, L.A. (2003). Web Intelligence and Fuzzy Logic – The concept of Web IQ (WIQ), Invited talk at the 2003 IEEE/WIC Int. Conference on Web Intelligence (WI 2003), available at: www.comp.hkbu.edu.hk/IAT03/InvitedTalk1.htm

Zadrozny, S. and Kacprzyk, J. (1998). Implementing fuzzy querying via the internet/WWW: Java applets, activeX controls and cookies, Proceedings of the 3rd International Conference on Flexible Query Answering Systems, Roskilde, Denmark, May 1998, pp. 382-392