INTEGRATING AND CONCEPTUALIZING HETEROGENEOUS ONTOLOGIES ON THE WEB

GOH HAI KIAT VICTOR

NATIONAL UNIVERSITY OF SINGAPORE

2006

INTEGRATING AND CONCEPTUALIZING HETEROGENEOUS ONTOLOGIES ON THE WEB

GOH HAI KIAT VICTOR

(B. Comp. (Honours), NUS)

A THESIS SUBMITTED

FOR THE DEGREE OF MASTER OF SCIENCE

DEPARTMENT OF COMPUTER SCIENCE

NATIONAL UNIVERSITY OF SINGAPORE

2006

Acknowledgements

The author is indebted to many people for their kind support of this research thesis. In particular, the author is extremely grateful to Prof Chua Tat Seng for his unwavering support and caring supervision. His countless occasions of sacrificing his own free time to provide advice and guidance to the author are greatly appreciated. His feedback about each phase of the research is more than just pointing out flaws and strengths of methodologies. He is able to analyze issues at a very comprehensive level and provide good suggestions. Moreover, his friendly and caring attitude has allowed the author to feel a balance between research and daily life. Under his supervision, the author has become a much better researcher, motivated and well prepared for any future endeavors.

The author would also like to thank Dr Ye Shiren for the numerous meetings to exchange ideas and resources. This research has been greatly hastened with his support and sharing of resources. Additionally, the author is grateful for the brainstorming of research issues with Mr Neo Shi Yong, Mr Tan Yee Fan, Mr Sun Renxu, Mr Mstislav Maslennikov, Mr Xu Huaxin, Mr Qiu Long, Mr Goh Tze Hong, Mr Seah Chee Siong and Mr Lim Chaen Siong. Their help in participating some of the research experiments together with many kind participants are also deeply appreciated.

Last but not least, the author would like to express his sincere thanks to Prof Ng Hwee Tou and Prof Lee Wee Sun for their constructive comments about the progress paper, which forms a basis of this thesis.

Table of Contents

Summary

The World Wide Web (WWW) has evolved to be a major source of information. The great diversity and quantity of information is growing each day. This has brought about an overwhelming feeling of having too much information or being unable to find or interpret data. In addition, since online information in HTML format is designed primarily for browsing, it is not amendable to machine processing such as database style manipulation and querying. Thus to obtain valuable information on the web, the data must first be organized and indexed. This can be done by performing some form of web structuring through discovering and building an ontology which describes the organization of specific web sites. By building good ontologies from the web, data can be easily shared and reused across applications and different communities. This research aims to develop techniques to analyze the inherent structure and knowledge of the web in order to build good ontologies and utilize them to perform information extraction, information retrieval and question answering. In particular, we extract data models from the web using an existing system and perform ontology integration based on their semantic meanings obtained from web searches, online guides, WordNet and Wikipedia. The integrated ontology is further utilized together with the contextual information on the web to discover latent user preferences and summarize information for users. In this thesis, we tested our system on I3CON, TEL-8 and online shopping data. The results obtained are promising and demonstrate a viable aspect towards future web information processing.

List of Tables

List of Figures

1. Introduction

The World Wide Web (WWW) has evolved to be a major source of information. The great diversity and quantity of information is growing each day. This has brought about an overwhelming feeling of having too much information or being unable to find or interpret data. In addition, since online information in HTML format is designed primarily for browsing, it is not amendable to machine processing such as database style manipulation and querying. Thus to obtain valuable information on the web, the data must first be organized and indexed. This can be done by performing some form of web structuring, such as storing data into a relational database or building an ontology. By building good ontologies from the web, data can then be easily interpreted, shared and reused across applications and different communities. The task of building ontologies and making effective use of them is thus a valuable research topic to be studied upon.

1.1 The Deep Web and Semantic Web

Although a lot of information may be seen on the "surface" web, there is still a wealth of information that is deeply buried or hidden. The main reason for this is that a substantial amount of information on dynamically generated sites is not collected by standard search engines. Bergman (2001) estimated that this substantial amount of information on the "Deep Web" is approximately 400 to 550 times larger than the commonly defined WWW. Traditional search engines are neither able to identify hidden links or relationships among "Deep Web" data, nor are they able to detect any underlying data schema. They create indices by spidering or crawling "surface" web pages. In order to retrieve any information, the data presented in a page must be static and linked to other pages. They are thus incapable of handling pages that are dynamically created as the result of a specific search or time. An example would be a search for recent sales of desktops and their prices, such as "Give me the most expensive brand of desktops and their configurations?". The hidden information among "Deep Web" sources is often stored in searchable databases that are not detected by traditional search engines. One solution to this problem is to identify all possible hidden information and store them appropriately.

Another problem which arises from the WWW is that data that is generally hidden away in HTML files is often useful in some given contexts, but not in others. For example, computer configurations, soccer statistics or election results are often presented by numerous sites in their own HTML format. It is thus difficult to integrate such data on a large scale. Firstly, there is no global system for publishing the data in a fixed format that can be easily processed by anyone. Secondly, it is difficult to organise and present the data from a global view. The solution to this is to define a format for presenting data, and also an automatic way of organising existing data. The Semantic Web is a major effort towards making this a success. The Semantic Web currently comprises of the usage of standards and tools like XML (Extensible Markup Language), XML Schema, RDF (Resource Description Framework), RDF Schema and OWL (Web Ontology Language). However, one major obstacle towards the realization of Semantic Web is in developing "standardized" ontologies for different domain, and in discovering such ontologies in many existing domains with vast amount of data in HTML formats. Thus, research into transforming and organising existing data into ontology-based formats are essential. Such research however, is still very much in the infancy period.

1.2 Motivation for this Research

With respect to the problems faced in Deep Web and Semantic Web, this research aims to utilize freely available web information to mine hidden knowledge in existing HTML-based web pages and store the extracted semantic information for shared use in various applications. In particular, ontologies are automatically extracted from various web sites, integrated into a "global" ontology, which can be used effectively to summarize or conceptualize information for presentation to the end users. Two important applications for this research include Question Answering and Semantic Web.

In Question Answering, an ontology provides a good framework that is useful in supporting queries. First, it allows us to better understand a given query. Second, it allows us to return better formulated results. Take for example a simple web query such as: “What are the best available desktops and their configurations?” Normal search engines would extract the keywords “best, available, desktops, configurations” and do a simple word matching in the database. This returns a set of possibly irrelevant documents which the users have to manually check through for his answer. However by looking into an ontology, one can know “desktops” means computer and “configurations” for computers include central processing unit (CPU), memory, storage, etc. Using this information, the retrieval system will thus be able to return the required answers effectively. At the same time, we can provide different views for different aspects of a query, for example all possible “configurations”. In short, by building and integrating ontologies, we can achieve a knowledge representation or better understanding of the available web.

In Semantic Web, we need a form of standardization that allows data to be shared and reused across applications, enterprises, and community boundaries. Due to the complicated format of data posted on the web, it is a difficult task to extract semantic information from the web or share any existing information. One possible way towards Semantic Web sharing is the re-publishing of every web site using the standards introduced, such as in XML, RDF or OWL. However such a process is infeasible on a large scale and many communities may disagree on doing so due to business secrets or security issues. Hence we need an automatic way of uncovering this information from the Deep Web and bridge this gap of information sharing. A good solution is to utilize existing web knowledge to assist in building or integrating a good ontology, ideally an exact replica of the available public information. By mere transferring of a global ontology across applications, we are able to facilitate ease in sharing and reuse. Moreover, the ontology allows users to have a “bird’s eye-view” about different key perspectives of available knowledge. For example in an ontology about Computers, when users want to know about Computers, they are also able to know different aspects of computers, such as its’ hardware components, history or brands. This ability to share, reuse and have a “bird’s eye-view” is especially useful for prospective commercial or educational applications.

The task of building and integrating ontologies on the web has tremendous growth potential. Even though Semantic Web communities are actively trying to promote a standardized way of publishing information, it will take a long time (or never, due to security issues) before the public or individual communities make any compromises. As information publicly available continue to explode every minute, ontologies research and maintenance will eventually be mandatory. This research project will therefore be focusing on using existing web knowledge to build and integrate ontologies. Furthermore, we hope to demonstrate the power of ontologies and how they can be used to generate better results for users. With the growing popularity in online shopping, we have decided to use online shopping websites as a test-bed for our research together with the public corpus of I3CON and TEL-8.

1.3 Contributions

The major bottleneck in ontology building is the integration or mapping between data models (Noy, 2004). Henceforth, this research focuses on ontology integration and advanced techniques to handle it. In particular, web knowledge in the form of online guide books, Wikipedia, and web search results will be used to improve the overall performance. Existing researches on extracting data model shows reasonable performance on certain specific domains (Ye and Chua, 2004). This can be done using wrappers or automatic identification via analysis of web page structures. In our research, we utilize the system for mining data model as discussed in (Ye and Chua, 2004). Furthermore, we build upon the Diamond Model framework presented in (Ye et al, 2006) to overcome its drawbacks in modeling semantic information for ontology integration. The results of ontology integration are further utilized to provide users with a summarized view of the available information. The main contributions for this research are: 1) resolve the problems of ontology integration due to the lack of semantic information, 2) provide a complete model for ontology usage and reusability, and 3) structure and conceptualize important information from the web for layman users or knowledge seekers.

The first part of this research analyzes existing works and proposes a good framework for ontology integration and usage. In particular, we identify how we can utilize existing external knowledge from the web to provide accurate contextual evidence for ontology integration, which is mostly missing in past researches. The second part of this research involves analyzing the effects of different proposed techniques in using web knowledge for ontology building. Finally, the last part of this research examines the different possibilities of conceptualizing information from the web and presenting them to end users in a summarized view. As online shopping information is of interest to most users, our research will use them as a preliminary test-bed together with the public corpus of I3CON and TEL-8.

The experimental results obtained for ontology integration shows that we can achieve an improvement of up to 21.8 in F1-measure when we incorporate external web knowledge for web ontologies. Subjective evaluation on the information returned through our ontologies also shows that majority of the users preferred our results as compared to information returned through other search engines or online shopping sites. The overall results show promising signs of how ontologies can be automatically mined, integrated and then presented to the users.

1.4 Thesis Outline

This thesis serves both as a critical survey for the existing works and as a research report for the general framework and experiments involved. Chapter 2 introduces the main differences in ontologies and how they exist in the real world. Chapter 3 describes existing related work and compares the benefits together with the drawbacks for them. Chapter 4 examines the main issues in ontology integration and how they may be tackled or improved. Chapter 5 discusses the main framework in our research and each sub-component for our system. Chapter 6 presents the testing and evaluation results obtained for ontology integration. Chapter 7 investigates how to perform ontology conceptualization and reports on the evaluations done. Finally, Chapter 8 concludes the thesis.

2. Types of Ontology

Ontology was first introduced by (Gruber, 1993) as an "explicit specification of a conceptualization". They are used to describe the semantic contents of any given information. When several information sources are given, ontology can also be used for associating or identifying semantically related concepts among the information. Besides being a form of explicit content, ontology are additionally used as a global query model or for verification during information integration (Wache et al, 2001). However, many ontologies that are existing or to be built are different. They are not only different in content, but there are also significant differences in their structure, languages and implementation. This section serves to provide a brief analysis to how ontologies may differ. For the rest of this report, we will use the terms *Concept*, *Element*, *Node* and *Object* interchangeably to mean the part of an ontology which is to be matched or merged.

2.1 Ontology Specification Language

At the current level of ontology research, there is no standardized way of building or designing an ontology. There exists a large variation of possible languages which can be used to describe an ontology. The native languages used to describe ontologies in early researches include mostly logic programming languages like Prolog. As ontology research evolves, there are languages that have been specifically designed to support ontology construction. The Open Knowledge Base Connectivity (OKBC) model and languages like Knowledge Interchange Format (KIF), or Common Logic (CL) are some of the specifications that have become the basis of other ontology languages. Several languages based on logics, known as description logics, have been introduced to cope with the demands of ontology description (Corcho, 2000). These include Loom

(MacGregor, 1991), DARPA Agent Markup Language (DAML), Ontology Interchange Language (OIL), and lately Web Ontology Language (OWL). In all ontology languages, there is a definite tradeoff between computation costs and the language expressiveness. The more expressive a language is, the higher the computation costs when evaluating or accessing the data in an ontology. Therefore, we should always choose a language which is just rich and expressive enough to represent the complexity of the ontology for its targeted purposes. Word Wide Web Consortium (W3C) have come to acknowledge this fact and many ontologies are increasing reliant on technology or specification like RDF schema as a language layer, XML schema for data typing and RDF to assert data. Henceforth, this research project will be handling ontologies mostly in RDF/XML format. An example of RDF/XML format for music soundtracks is shown in Figure 2.1. From the example, we can clearly see that the data is restricted to a certain format which is easy to check for consistency. In the example, a music soundtrack must contain an artist, price and year. Computers can then use these resource declarations to assert that any valid soundtrack listing should have these fields and their respectively data type.

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:cd="http://www.sample.music/cd#">

<rdf:Description
 rdf:about="http://www.sample.music/cd/Empire Burlesque">
  <cd:artist>Bob Dylan</cd:artist>
  <cd:price>10.90</cd:price>
  <cd:year>1985</cd:year>
</rdf:Description>

<rdf:Description
 rdf:about="http://www.sample.music/cd/Hide your heart">
…
</rdf:Description>
…
</rdf:RDF>
```

Figure 2.1 An example of RDF/XML format

2.2 Semantic Scope

Besides differences in language specifications, ontologies also differ in their purpose and meaning of their contents. There are two main levels of ontology scope, domain-specific (lower level) or global (upper level). Domain specific ontologies describe specific fields of information about a selected domain, for example in electronic products or medicine. Conversely, global ontologies describe basic concepts or relationships about information with respect to any domain. WordNet (Miller et al, 1993) which is used widely by natural language researchers is one example of a global ontology. The main drawbacks of a global ontology are the sparseness of data involved and the ambiguities present when referencing an object. For example when searching for "windows" in the global ontology, one may either refer to it as "*Microsoft Windows*", "*glass windows*" or "*time windows*". The scope is often too wide and there is no definite way of resolving the ambiguities unless some context information is provided. In contrast, domain specific ontologies are capable of handling specific queries directed to their domain, but are not sufficient since the scope may be too narrow. A hybrid way of using ontology is to create many domain specific ontologies and overlay it with a global ontology or global classifier. Any given information is first classified or matched to a particular domain specific ontology before further processing. This research adopts this hybrid approach for efficiency and coverage purposes.

2.3 Representation Level

Different ontology builders adopt different methodology when describing or creating ontologies. There are several levels of representation which can be used to describe an ontology. The simplest is the use of a set of lexicons or controlled vocabularies. For example, "food" concept may comprise of “edible, vegetables, and meat”. Slightly more advanced representations include categorized thesauri which groups similar terms together or taxonomies where terms are hierarchically organized. Other representations may also involve complex descriptions about distinguishing features or named relationships with different concepts. The SUMO ontology (http://www.ontologyportal.org/), for instance, contains axioms which define relationships such as “have molecular structure of” and “sub-region of country”. The level of representation required depends mainly on the purpose of the final ontology.

2.4 Information Instantiation

One major difference in all ontologies is their terminological component. This is specifically known as the schema for a relational database or XML document. Each schema defines the structure of the ontology and the possible terms or identifiers used. Some schemas include an assertion component which describe the ontology with example instances or individuals that is evident for the terminological definition. This extra assertion component can often be separated from the main ontology and maintained as a branched knowledge base. The main issue in whether a given object can be classified as a concept or an individual instance is usually an ontology specific decision. For example, “Sony MP3 player” can be an instance of electronic products, while “Walkman Bean” (a type of Sony MP3 player) can be an instance of electronic products or as an instance of the subclass of “Sony MP3 player”. The definitions may vary across multiple different ontologies, but all of them are still considered valid.

3. Review of Related Works

Ontology integration is a widely discussed topic among Database communities, Semantic Web researchers and Knowledge Engineering groups. As described in the previous Chapter, ontologies come in many different forms and variations. Thus the main purposes of ontology integration may be simplified into these few categories:

a) To obtain a common specification. Integration is done based on the differences in their *specification languages*. This is usually done assuming the context of the ontologies are the same and only the expressiveness/expression is different. An example would be ontologies about "Cars" where ontology *A* is written in Prolog syntax while ontology *B* is written in RDF.

b) To achieve a standard compromised *scope*. This can only be done with understanding of the context for concepts under different ontologies. An example is "windows" in ontology *A* refers to Microsoft Windows because it can install software, and in ontology *B* also refers to Microsoft Windows because virus can attack it.

c) To obtain similar level of *representation* or a more complete global representation. An example would be to define the "food" concept in simpler ontology *A* with logic statements (instead of pure lexicons) and merge it with concepts under the more comprehensive ontology *B*.

d) To establish agreement between different information *instantiation* level or create links/new nodes between them. For example, "Walkman Bean" in ontology *A* is linked as a subclass of "Sony MP3 player" in ontology *B*.

Furthermore, ontology integration are usually done either on merging the taxonomy and concept hierarchy (Halkidi, 2003), or merging the data model in the form of schema integration. With respect to these objectives in mind, this Chapter provides a review of existing works in ontology integration and gives a brief analysis for them.

3.1 Database-styled Integration

Ontology integration under this category follows the ideas of database schema integration which is actively researched under the Database Communities. Many databases which contain catalogues, records, indexes or even classification systems are often also considered to be an ontology. The problems that arise due to difficultly in database schema integration were discussed in depth by many database experts (Batini, et al 1986), (Wache, 1999), (Noy, 2004). The main issues in this form of integration is in 1) removing data heterogeneity conflicts between the many different databases, 2) resolving the schema differences between two or more heterogeneous databases, and 3) creating a global schema that encompasses the smaller schemas for integration. Ideas discussed under such integration may provide good insight to the direction for general ontology integration or merging. Most definitions used under ontology integration were also first introduced here. Some examples include semantic relevance, semantic compatibility and semantic equivalence.

A good survey of different schema integration techniques was first given in (Batini et al. 1986). They proposed that schema integration should include at least five main steps: pre-integration processing, comparison, conformation, merging and finally restructuring. The main idea in database integration techniques was to do integration by utilizing expert systems or agents (Bordie, 1992). InfoSleuth (Fowler et al, 1999) and Retsina (Sycara et al, 2003) are two examples of such systems. Most of such agents are based on the concept of mediators which provide intermediate response to users by linking data resources and programs across different sources. However, one major drawback of such systems is that they require all domain knowledge to be given in a controlled vocabulary.

Two other techniques were given by (Palopoli et al, 2000) to abstract and integrate database schema. They assumed that there is an available collection of existing inter-schema properties which describes all semantic relationships among different input database objects. The first technique uses these inter-schema properties to produce and integrate schemas. The second technique uses a given integrated schema as the input and outputs an abstract schema with respect to the given properties. The main problem they faced in achieving a good schema integration is the absence of semantic knowledge embedded in the underlying schemata. It is conjectured that complete integration can only be achieved with a good understanding of the embedded semantics in the input databases. Consequently, the use of meta-level knowledge is investigated by (Srinivasan et al, 2000). They introduced a conceptual integration approach which measures similarity on database objects based on meta-level information given. These similarities are then used to create a set of concepts which provide the basis for abstract domain level knowledge. We must note, however, that the meta-level knowledge given beforehand must be sufficiently reliable or in most cases, composed manually. In summary, the main idea in most database-styled integration is to focus on integrating schemas on a semantic level or based on the understanding of meanings.

3.2 Rule-based Integration

This form of ontology integration makes uses of logic, rules or ontology algebra. The main idea of such approaches is to derive a set of rules for integration. For example in (Wiederhold, 1994), the system utilizes ontology algebra to perform three main operations for integration: difference, intersection and union. The algebra also provides a way to create rules (or articulations) to link information across different domains or disjoint knowledge sources. The rules written in algebra form presumably enable one to create knowledge interoperability. All mappings or semantic information are expressed in mathematical terminologies which may provide ease in inferences and knowledge portability. However, one main disadvantage is that such rules are often hard to find or create, and have to be tuned towards each given domain.

Another example which uses rules for integration is (Mitra et al, 2000). With the support of a basic set of articulation rules, they used ontology algebra to create more specific rules for linking of information between ontologies. The ontology graphs of the ontologies are given as input for the creation of such rules. The main operations in their algebra involves producing new articulation ontology graph, which consists of the nodes and the edges added to the rule generator using the basic articulation rules supplied for the two ontologies. The main drawbacks in their work include the need of a set of well-formed articulation rules and also the difficultly in crafting them for different ontology pairs.

Other similar researches in this field include (McCarthy, 1993), CYC (Guha, 1991) and (Hovy 1998). McCarthy used simple mathematical entities to represent context information which can be used during situations when certain pre-defined assertions are activated. In addition, there is a notion of lifting axioms to state that a proposition or assertion in the context of an ontology is also valid in another. Similarly in CYC, the proposed use of “micro-theory” is designed to model some form of context information. Each micro-theory is a set of simple context assumptions about the knowledge world. One interesting point to note is that micro-theories are organized in an inheritance hierarchy whereby everything asserted in the super micro-theory are also true in the sub-class lower level micro-theory. On the other hand, Hovy went back to basics and used several heuristic rules to support the merging of ontologies, namely the *Definition*, *Name* and *Taxonomy* heuristics. *Definition* compares the natural language descriptions for two concepts using linguistic techniques; *Name* compares the lexical names of two concepts; and *Taxonomy* compares the structure proximity of two concepts. As in all rule-based systems, the difficulty for such forms of integration arises from the fact that rules are often hard to craft and maintain for each given domain or each ontology pair.

3.3 Cluster-based Integration

The focuses of this type of ontology integration is to pre-group similar objects together and present them as results. When given large ontologies where it is hard to perform the integration process, this may be a possible choice. Concepts or nodes across ontologies are clustered by finding the similarities between them under different situations, applications or processes. (Visser and Tamma, 1999) proposed this idea for "integrating" heterogeneous ontologies in 1999. They clustered concepts based on their similarities given by information from different agents (or humans in their context). Each cluster in the "final" ontology is described by a subset of concepts or terms from the WordNet (Miller et al, 1999). A new ontology cluster is a child ontology that defines certain new concepts using the concepts already contained in its parent ontology. Using WordNet as the root ontology, concepts are described in terms of attributes, inheritance relations, and are hierarchically organized. They tested this approach on a small scale for the domain of coffee. Since they do not consider the existing schemas of given ontologies, it is doubtful this approach can be used for perfect schema integration of ontologies. However, the simplicity in presentation of results to the users may be useful for querying multiple ontologies at once where full ontology integration is not required.

Another research under this category was proposed by (Williams and Tsatsoulis, 2000). They used an instance based approach for identifying candidate relations between diverse ontologies using concept clusters. Each concept vector represents a specific web page and the actual semantic concept is represented by a group of concept vectors judged to be similar by the user based on their web page bookmark hierarchies. Their approach uses supervised inductive learning to learn their individual ontologies and output semantic concept descriptions (SCD) in the form of interpretation rules. The main idea of their system DOGGIE is to apply the concept cluster algorithm (CCI) and identify candidate relations between ontologies. Each concept cluster may contain one or more candidate relations for the concepts. The experimental results looks promising, but since they only consider candidate relations in the form of "is-a" relation, it is uncertain if they will perform well for other forms of relations, such as "part-of", "sub-class".

3.4 Specific Methods and Systems Review

3.4.1 InfoSleuth

InfoSleuth by (Fowler et al, 1999) is designed to support construction of complex ontologies from smaller component ontologies. They believed that tools tailored for one component ontology can be used in many application or domains. Two examples of reusable ontologies are units of measure, and geographical or country data. All the mappings among the ontologies are explicitly specified as relationships between terms in an ontology and related terms in another ontology. They used a special class of agents called "resource agents" to perform these mappings. A resource agent encapsulates a set of information about the ontology mapping rules, and presents that information to the agent-based system in terms of one or more ontologies. It acts as a wrapper for the underlying data source and presents only part of the overall domain ontology that it supports. This can be seen as a projection and usage of the projection to create complex ontologies. This early work depends mostly on manually created templates/wrappers and does not seem to be very scalable.

3.4.2 RDF-Transformation

(Omelayenko and Fensel, 2001) presented an approach to the integration of product information on the web through the use of RDF (resource description framework) data model. It is based solely on the directed labeled graphs of the RDF data model. They assume that all product catalogs from different organizations or domains are already well specified in XML documents. The only problem that may exist is in different representations about the same product. To resolve this, they proposed a two-layered method whereby one layer handle the product information presented in XML, and the other layer handle the transformation or translation between different representations in RDF. The main idea is that XML document contains a structure defined by their schema and can be transformed into an RDF data model graph using XML transformation language (XSLT). This RDF data model can then be compared or merged to output an ontology. In a later paper by Omelyayenko (Omelayenko, 2002), a technique was proposed for discovering semantic correspondence between two different product data models. A naïve-bayes classifier was used to identify the semantic group based on instance information for the each data model. This approach stresses on the importance of RDF structure as a basis for comparison. One main problem that exists in the real world is that data are often not well structured for such an approach to be workable.

3.4.3 ConceptTool

The ConceptTool developed by (Compatangelo and Meisel, 2002) is based on a description logic approach to formalize a domain specific, enhanced entity relationship model. Their work aims to facilitate knowledge sharing through an interactive analysis tool for ontology experts. The tool assists experts in aligning two ontologies through several enhanced entity relationship models augmented with a description logic reasoner. The core of the system involves the use of linguistic and heuristic inferences to compare attributes between concepts of two ontologies. At each stage of the comparison, the expert will be prompted with relevant information to resolve conflicts between overlapping concepts. Overlapping concepts are linked to each other by way of "semantic bridges". Each bridge allows the definition of transformation rules to remove the semantic mismatches between these concepts. There are 6 main steps in ConceptTool: analysis of schemas to derive taxonomic links; analysis of schemas to identify overlapping entities; prompting the expert to resolve overlapping entities; automatic generation of entities in the articulation schema after resolving each pair of entities; prompting the expert to define the mapping between attributes of entities; and finally summarization or analysis of the articulated schema. Though this is a useful system for domain experts, the amount of manual work and heuristic rules involved makes it very restrictive and not scalable.

3.4.4 ONION

As a follow up to the rule-based integration system described in section 3.2, Mitra and Wiederhold (Mitra and Wiederhold 2002) developed the Ontology compositION system (ONION) which provides an articulation generator for resolving heterogeneity in different ontologies. It is an architecture based on algebra or rule formalism to support ontology integration. One special feature of this system is that it separates the logical inference engine from the representation model of the ontologies as much as possible. This allows the accommodation of different inference engines when necessary. The basic system contains a data layer which manages the ontology representations, the articulations or rule sets involved and the rules required for query processing. The authors argue that ontology merging into one global source is inefficient and costly. They claim that one global information source is not feasible due to too many inconsistencies among the ontologies. Hence they tried to resolve the semantic heterogeneity by using articulation rules which express the relationship between two or more concepts. These rules are manually created which take into account relationships such as "attribute of", "instance of", "subclass of", "part of", and "value of". In their experiments the ontologies used were constructed manually from two commercial airlines websites. One interesting point to note is that they included a learning component in the system which takes advantage of users' feedback to generate better articulation in the future.

3.4.5 IT-Talks

(Prasad et al, 2002) proposed the use of text classification techniques for ontology integration in their web-based system for notification of information technology talks. Their system uses text-based classification (as in information retrieval), and Bayesian reasoning for resolving uncertainty. The text classification technique they used generates scores between concepts in the two ontologies based on their tagged relevant documents. Bayesian reasoning is then used to check for subsumption (coverage). If a new concept is partially matched with majority of the children of a higher level concept, then this higher level concept is chosen over (or subsumes) the direct match with its children. The authors also tried an alternative algorithm for subsumption by considering the best mapping as 1) the concept that is the lowest in the hierarchy and, 2) the posterior probability is greater than 0.5. They experimented with two hierarchies, namely the ACM topic ontology and a relatively small ontology about IT talks. In general, their use of a classification based approach seems reasonable but is yet to be tested on a large corpus.

3.4.6 GLUE

As a port of the original proposed system of LSD in (Doan et al., 2001), GLUE (Doan, 2002) is a system aimed at detecting schema mappings for semi-automatic data integration. GLUE is among one of the systems that uses machine learning techniques to find mappings. It first applies statistical analysis to the available data to compute a joint probability distribution. Based on the distribution, it generates a similarity matrix for the data and use relaxation labeling to obtain the mappings. GLUE also tries to exploit information in the data instances or in the taxonomic structure by employing different learners. The author developed two learners, a content learner and a name learner. The content learner uses a Naïve Bayes text classification method, while the name learner does the same but uses the full name of the instance instead of its content. A meta-learner is then used to combine the results or set the weights. The main algorithm in GLUE works in three basic stages: 1) learn the joint probability distribution for classes of each ontology, 2) compute the similarity between pair-wise classes as a function derived from their joint probability distributions, and 3) employ heuristic rules for constraint relaxation to choose more likely mappings. The author tested the system on university course catalogs and showed promising results of above 70% accuracy. However, more experiments should be done to evaluate GLUE in other domains and test its scalability.

3.4.7 CAIMAN

Another system which uses machine learning for ontology integration is CAIMAN (Lacher and Groh, 2001). The system aims to maintain different perspectives or views on the ontologies for different users. The use of bookmarks or folder structure is assumed to be a form of individual ontology for the users. The authors then tried to integrate these individual ontologies with the directory structure of CiteSeer (http://www.researchindex.org/). They measure the probability of two concepts being the same through text classification methods. For each concept node in the ontology to be integrated, a corresponding node in the community ontology is identified through classification. It is assumed that repositories stored on both the user and community portal contain some actual documents (for context information mining), as well as links to their physical locations. The authors claimed that information has to be indexed in an understandable way by each user and thus do not provide support to formal community ontologies in their system. It is therefore doubtful when applying this system on a broader scope. However, the idea of classification for integration is still worth looking into.

3.4.8 CUPID

The CUPID system, which was co-developed with Microsoft by (Madhavan et al, 2001), implements a generic schema matching algorithm. The system combines linguistic and structural schema matching techniques, and computes normalized similarity coefficients based on a predefined thesaurus. The input to the system is a set of schemas represented in the form of graphs. Each node represents a single schema element. The graphs are traversed in both a bottom-up and a top-down manner. The matching algorithm consists of three stages. The first stage computes the linguistic similarity coefficients between schema element names based on morphological normalization, string-based matching, categorization, and a simple thesaurus look-up. The second stage computes structural similarity coefficients which measure the similarity between contexts. The main idea in structural matching is to compute similarity between non-leaf nodes based strongly on leaf node matches instead of the immediate descendents or intermediate substructures. The third and final stage of CUPID computes weighted similarity coefficients and generates the final mappings by choosing pairs of schema elements with weighted similarity coefficients which are higher than a threshold. The CUPID system demonstrates that automatic ontology integration may be feasible and should be further investigated.

3.4.9 FCA-Merge

FCA-Merge (Stumme and Mädche, 2001) uses formal concept analysis (FCA) techniques (Ganter and Wille, 1999) to merge two ontologies sharing the same set of instances. The general idea in Formal Concept Analysis is to use a formal context defined as a triplet K:=(G, M, I), where G is a set of objects, M is a set of attributes, and I is a binary relation between G and M. The algorithm in FCA-merge first extracts instances from text documents which represents the concepts and assigns them to the ontologies to be merged. Second, it creates a boolean table indicating which instance belongs to which concept. They use lexical analysis to associate single words or composite expressions with a concept from the ontology if a corresponding entry in the domain-specific part of the lexicon exists. Third, it computes a lattice based on the ontologies and instances belonging to each of them. The contexts or instances are merged in the lattice during this process by means of classical formal concept analysis (coverage principle). In short, the final lattice contains only concepts that are general, and those that are not more general than these concepts from the ontologies are removed. Fourth, the final stage, requires the help an expert to further simplify the lattice and generate the final taxonomy of the ontology. This last stage in deriving the merged ontology from the concept lattice strongly requires human interaction. There are two assumptions to be made under FCA-merge: 1) the documents should be representative of the domain to be merged and should be closely related to the ontologies, and 2) the documents have to cover all concepts from both ontologies as well as being able to differentiate them. This idea of using context information is sound, but it forgoes the benefits of using available structure already present in the ontologies, such as hierarchy of the concepts or nodes. A good system should be able to make use of both structural and context information.

3.4.10 IF-Map

IF-Map by (Kalfoglou and Schorlemmer, 2003) is another system similar to FCA-merge. It is an automatic method for ontology mapping based on the Barwise-Seligman theory of information flow (Barwise and Seligman, 1997). Their method draws on the proven theoretical ground of Barwise and Seligman's channel theory. The basic principle of IF-map is to merge two local ontologies by looking at how they are mapped from an external reference ontology. The local ontologies are assumed to contain many instances which may be mapped to the reference ontology (where there are usually few or no instances). There are 3 major steps in IF-Map: 1) ontology harvesting; 2) translation of ontologies in different languages to the same format, Horn logic for their Prolog engine; 3) logic info-morphisms: the task of generating all possible mappings between the unpopulated reference ontology and each of the populated local ontologies. This is done by considering how each local community classifies instances within their local ontology. By mapping concepts to the same node on the reference ontology, one could then decide if the concepts should be merged. As with FCA-merge, IF-Map lacks a good use of both structural and context information.

3.4.11 PROMPT, ANCHOR-PROMPT, PROMPT-DIFF

These systems are collectively developed by Noy and Musen as part of their Protégé-2000 package. PROMPT (Noy and Musen, 2000) or SMART (Noy and Musen, 1999) was developed around 1999, and added an extension to become Anchor-PROMPT (Noy and Musen, 2001). Anchor-PROMPT is an ontology merging and alignment tool with a complex prompt mechanism to handle possible matching terms when they are encountered. The input to the system consists of two ontologies and a set of pre-defined anchors-pairs of related terms. These anchors pairs can either be defined manually or identified with the help of string comparison methods. It first uses the ontologies to construct directed labeled graph from a hierarchy of concepts and relations. Each node in the graph represents a concept and the edges represent the relations between nodes. Then it analyzes paths in the graphs between nodes specified as anchor-pairs. Following a graph perspective, it collects a set of paths that connects the terms of an ontology which are related to terms of the other one. The frequency of concepts or terms appearing in similar position is then used to decide if two nodes are semantically similar to another. The results show that the accuracy of Anchor-PROMPT is directly proportional to the length of the paths considered. For instance, with paths of length 2 can achieve an accuracy of 100% while path of length 4 has only to 67%. The latest in their proposed system is PROMPTDIFF (Noy and Musen 2002), an algorithm which integrates different heuristic matchers for comparing ontology versions. One general point we should note is that these systems tries to model the structure integrity that is present across ontologies. However, structure itself may not be fully sufficient for merging (as can be seen by the decrease in accuracy when path length increases). There is a need to understand the concepts semantically using some form of additional context information, for example external web knowledge.

3.5 Overall Analysis of Related Work

Most of the systems or related work in ontology integration relies, to different extent, on the help of human experts to accomplish the task. Despite the fact that tools have been developed to assist ontology integration through suggestion or checking, there is no good unsupervised method to perform ontology integration automatically. The task of ontology integration is not just a simple pair-wise object comparison. It requires understanding of semantic meanings for each object. Moreover, the fact that objects can have many-to-many, many-to-one, or one-to-many relationships within a single domain makes the task even more difficult. Another major flaw in most current works is that they do not effectively capture the benefits of context and structure existing within the ontologies. Most works usually focus on using only one source of information. For effective ontology integration, both semantic (the meaning of objects) and structural information (hierarchy of objects, their relationships) need to be investigated. However, this type of information is hard to obtain in previous works. This is because they generally focus on a very broad scope of general domain. Instead of that, by identifying specific domains to work on and combining them later, we can effectively rely on domain specific structures or knowledge to automate the whole integration process. Nonetheless, not much work has been done to evaluate this effectiveness and it is worth researching into.

From another perspective, ontology integration can also be seen as a projection of ontologies from different points of view, either according to the needs of different applications or tasks (as in cluster-based integration). Regardless of the form in ontology integration, there are still many research issues in the area of semi-automatic or automatic integration. In the next two chapters we will discuss and propose some methods to do ontology integration automatically, which is one of the aims of this research project.

4. Heterogeneous Ontology Integration and Usage

4.1 Issues in Ontology Integration

One of the main issues in ontology integration (or semantic integration) is how a mapping between two ontologies can be derived. There exists many ontologies, either freely available or constructed by domain experts, and the shear number of these ontologies and schemas makes it extremely difficult to manually define correspondences, articulation or rule sets for each mapping. Moreover, in the world of World Wide Web, new information is published at an exponential growth rate. There is therefore a definite need for automatic information organization. This in turn, gave rise to a strong need for automated or semi-automated way to integrate existing or newly built ontologies. However, it is not an easy task for good ontology integration. At times, the ontology integration process can be extremely laborious and error-prone.

4.1.1 Difficulties in Ontology Integration

Given any two ontologies *A* and *B*, the task of most ontology integration is to be able to decide whether an element *a* of *A* and an element *b* of *B* are the same. The equivalence should depend on the real world representation or how real humans perceive them. This task is extremely difficult due to several reasons. First, due to the fact that an element *a* from Ontology *A* may map to more than one element in Ontology *B*, we need to compare element *a* with all elements in Ontology *B* before deciding the final best match. These processes of comparison with all elements are very costly and often cause a significant rise in the overall computation costs.

Second, matching between elements is often very subjective, even when they are very similar lexically. An element by the name of "Ford" under Ontology *A* about cars may map to model description in Ontology *B* about cars, or may only be suitable as popular brand names. The matching depends mainly on the required application and also the context which the element occurs in. This is one of the foremost reasons that many existing systems require some form of human invention. In some extreme cases, this ambiguity between elements may even require a collective agreement by expert users before confirmation of a match. In order to solve this problem, external knowledge or some form of contextual information need to be present.

Third, there are sometimes too few sources of information to provide enough evidence for matching of the elements. There may not be enough contextual information, no schema documentation, or no references to identifiers or complex terms. For instance, given only two element names, such as “order” and “command”, should we map both together (using “order” as a form of authoritative command), or should we map it to an action of “buying”? The process of obtaining additional information is often very difficult. Ontology builders who created them may have changed jobs, forgotten about the schema, retired or perhaps even passed away. Any documentations or descriptions are also likely to be brief, outdated, incorrect or non-existent. Some available information may also be incomplete. For example, the element name “light-cars” implies that the element is something involving cars, but it does not tell us whether it refers to lightweight cars or lightings for cars. As with the second problem, one solution is to use an external source of information to compensate for the lack of evidence. Examples of such information sources are the WordNet, or web search results.

The last problem in ontology integration is the reliability of the information source comparisons. Many existing systems measure element similarities based on the given schema and data information. These usually include element names, data values, data types, schema structures, imposed constraints or element descriptions. However, comparison based on such information may not be reliable. For example, two elements with the same name may refer to different things (such as feet for human feet or unit of length), or elements with different names may refer to the same thing (such as drinks and beverages). The proposed solutions to this problem may be the use of a confidence measure to determine the confidence for each type of comparison, or more reliable methods for comparing between information sources.

4.1.2 Rule Crafting vs. Machine Learning

Though not distinctively pointed out in the previous sections, there is a distinct difference between ontology integration using rule crafting (for example, InfoSleuth) or machine learning methods (for example, FCA-merge). Both methods have their own benefits and drawbacks. Rules rely on expert knowledge of a domain, and are relatively inexpensive to craft if the given domain is small enough to work on. They do not require any form of training when compared to machine learning. In addition, they run quite fast since they are just direct application onto the schema without any major computation. In some cases, rules can also be fine-tuned to the effect that it can work quite well for a given domain. Some experts also pointed out that rules can provide a quick way to capture valuable user knowledge, especially in the form of regular expression. For instance, a regular expression that detects phone number formats can be written easily or a list of local phone numbers can be downloaded from phone directories to be used.

Machine learning methods may be difficult to learn these rules if they do not have sufficient well selected training data. On the other hand, machine learning is beneficial over rule crafting because they can exploit data redundancy to capture a series of information that may not easily be captured or thought of through knowledge crafting. Examples of such "data unveiled knowledge" include highest co-occurring words, popular context descriptions, different value range or perhaps inherent patterns in structure. Rules crafted by experts may not be able to cover all these variety of information and as the domain or extend of scope expands, rules may sometimes simply be impossible to craft. An example for such cases is to find the dissimilarity between articles describing cars and those describing car companies. There is no definite way of writing rules for a binary classification but machine learning that incorporates simple probabilistic or frequency analysis of words can often do the trick. More advanced machine learning ideas, such as neural networks (Li and Clifton, 2000), may also help to improve the results. Furthermore, machine learning methods can easily make use of feedback or past matching results to assist in future matches (sometimes simply by passing it as new training data). Rule crafting methods in contrast will not be able to do so unless an expert constantly review and modify the rules whenever new information is given at every iteration. Weighing the tradeoffs between both approaches and the need for constant future improvements, this research project has chosen to take the latter approach.

4.2 Matching Methods

During the process of mapping one element in Ontology *A* to another element in Ontology *B*, there are several common methods which are used for similarity measures. This section discusses some matching techniques which are commonly used in ontology integration.

4.2.1 Term Matching

This level of matching can be considered as one of the most basic. The main component is to compare term differences. Terms in such cases are usually lexical tokens or a word. Therefore this method is usually applied on the names, labels or titles of elements. Recent researches also extended this form of matching to compare class names, URL (Uniform Resource Locator) and URI (Uniform Resource Identifier). There are two main methods for this level of matching. They are namely 1) Lexical String Matching and 2) Linguistic Feature Matching.

Lexical String Matching. This method compares the difference between terms in the form of lexical strings. It considers the structure of strings as a sequence of characters or literals.

There are several ways to compare strings depending on what format the string is given. For example, Substring or Edit Distance matching for shorter strings; Word Distance or Word Sequence matching for text description.

Linguistic Feature Matching. This method utilizes techniques in Natural Language Processing (NLP) to extract linguistic features for the given terms and perform matching on these features. The features can be inherent in the term itself by parsing it through standard NLP tools (such as Part-of-Speech tagging, morphological analysis) or extracted based on external resources (such as synonyms, multilingual translation). The main aim of this method is to make use of natural language to formalize the meaning for the term during comparison. In majority of the cases, it can also be classified as a form of term variation detection. Terms can vary morphologically, semantically, or syntactically. Variations to this form of matching include the use of Soundex (ie. an index to how a term is pronounced verbally).

One main problem which is encountered in this form of matching is that terms can refer to more than one concept, or reversely, many terms can represent a same concept. Though many researchers will hope not to have this kind of ambiguity, it is a fact that most linguists have come to accept, even across different human languages. This problem does not only occur at a general level for any data instance, it also exists when the ontology is domain specific. In cases of web URIs, identical names must refer to the same web page or object, but there may be two similar objects with different URIs. If not effectively taken care of, this problem of ambiguity for term matching may be propagated when many ontologies are integrated together. The chief reason for this error propagation is the inconsistencies that may exist across different sub-ontologies, such as naming conflicts or relation conflicts.

4.2.2 Structure Matching

Any ontology that is constructed will contain some form of structure, either in the way concepts are organized or the possible representations within a concept. This form of matching leverages on the existing structures among ontologies for their integration. Instead of performing basic terms matching as in the previous section, the structures between two ontologies are compared.

This method takes into consideration the whole structure of the given ontologies. In particular, similarity is computed based on the positions of element in the ontology tree and possibly the surrounding or neighbour nodes of the element to be matched. The idea of using neighbour elements is based on the assumption that two elements about the same concept should have relatively similar neighbours in different ontologies. Henceforth, the decision of matching depends usually on their position, their siblings, descendants, ancestors or some sort of combinations.

Despite the novelty of structure comparison, we have to avoid the pitfall of comparing two structurally different ontologies or giving too much weight to the results. The structures of ontologies may be widely varied even when they are about the same domain. For example, "Animals" in Ontology *A* may be specialized with descendants "Cold-blooded", "Warm-blooded" while in Ontology *B* it may be "Land", "Air", "Sea". In such cases structure matching will add noise instead of improving the results.

4.2.3 Attribute Matching

This method makes use of attributes and properties that exists in the way an element or concept is presented. The most common comparisons for this form of matching are on value range, cardinality and inherent relations. Since there may be many concepts or elements with the same attributes or properties, this form of matching is often not very accurate. They are usually used to create initial clusters or groups to be further refined using other available matching techniques. (Li and Clifton, 2000) proposed to use this as a filtering step to remove elements with obvious attribute incompatibility.

4.2.4 Kernel Function/Validity Matching

Most ontologies may be constructed differently and presented in a different manner. The main idea of this form of matching is to transform ontologies using a kernel function and compare them on a common space, or check for their validity. Most of such transformations involve the use of logic such as the Boolean Satisfiability (SAT), modal SAT, or Description logic. Shvaiko (2004) demonstrated the use of modal SAT for extending matching methods related to propositional SAT and binary predicates. The key idea is to enhance the propositional logics with modal logic operators and map ontologies using them, thereby effectively transforming the matching problem into a modal logic formula. The formula is then checked for its validity using sound and complete satisfiability search procedures. An invalid formula would then mean a failing match. In most cases, this level of matching procedure will either return only a true or false for a match. For good results, this matching procedure usually also require some form of human interaction to define seed pairs for formula construction. The main benefit of such a procedure is the expressiveness of similarity between ontologies or the elements. For example there can be a logic statement that states two particular elements are equivalent since they share all similar instances. The drawback however, is the difficulty in creating a good translation formula and evaluating the output. Since each match can only be true or false, all cases of partial matching between ontology elements are totally removed and this in turn, may reduce the overall accuracy.

4.2.5 Instance Matching

This final form of matching depends not on the element or concept to be matched, but on the instances that they cover. The key idea is to have a collection of instances or examples which is covered by each concept or element to be compared. In order to decide whether two elements are similar, the collections of instances from the two elements are compared. Generally, the more overlap or similar the instances are, the higher the similarity score for the two elements. The similarity among instances can be computed based on any of the earlier discussed methods or text-based classification methods if instances are in plain text. A simpler way of using instance matching is to test collection intersection. When collection A_c intersect B_c is equivalent to A_c and to B_c at the same time, we infer that they are similar (and vice-versa if they are not equivalent). By and large, this method of matching provides a good basis for comparison between entities based on real life data, or instances which satisfy the ontologies. The only drawback is the difficult task of obtaining a good collection of instances.

4.3 Frameworks for Ontology Usage

The aim of ontology integration is influenced by the way ontologies are to be used or presented to the users. Ontologies are commonly assumed to be the basis for explicit description of information from semantic sources. This explicit description needs to be effectively organized before we can use them. This section serves to discuss 3 main ways which we can organize ontologies and utilize their information. They are namely: 1) single global ontology 2) multiple local ontologies 3) fusion approach. Figure 4.3.1 illustrates some of these frameworks.

Single Global Ontology. This approach for using ontology requires all existing ontologies to be merged into a single ontology. The global ontology provides a form of shared vocabulary and integrated global knowledge to the world. SIMS by Arens et al (1996) is an example for this kind of approach in ontology integration. Their global ontology model consists of a hierarchical knowledge base with a tree like structure, having nodes represent different states, actions and objects. All information from existing ontologies is represented in the global ontology by creating relations between objects from each ontology source and the global ontology. The relations created aims to define the semantics of the source objects and helps to group semantically related objects in the global ontology. In general, the global ontology can be either a combination of many related ontologies or several specialized ontologies. In the case of specialized ontologies, there may be no direct overlap between objects in semantic meanings. The main purpose of such integration is to provide a common space for knowledge representation.

For global ontology which integrates many related ontologies, there is often a need to define the level of granularity. This is because not all information sources have an identical view about the same domain. A slightly different view would mean a completely different level of description or thus granularity. As first person, Gruber (1995) stated that finding the minimal ontology commitment becomes a rather difficult task in such cases. The coverage of information sources may be different and thus it is difficult to obtain a compact global ontology after integration.

Another issue in the usage of a single global ontology is the need of monitoring each local ontology. Whenever there is a change in the local ontology, the conceptualization in the global ontology may be affected. This often gives rise to the costly need of rebuilding the global ontology whenever there is a minor change. The rebuilding is needed to reflect the correct information published and also to provide a correct mapping to other related ontologies. This major disadvantage of using a single global ontology directs researchers to consider the use of multiple local ontologies.

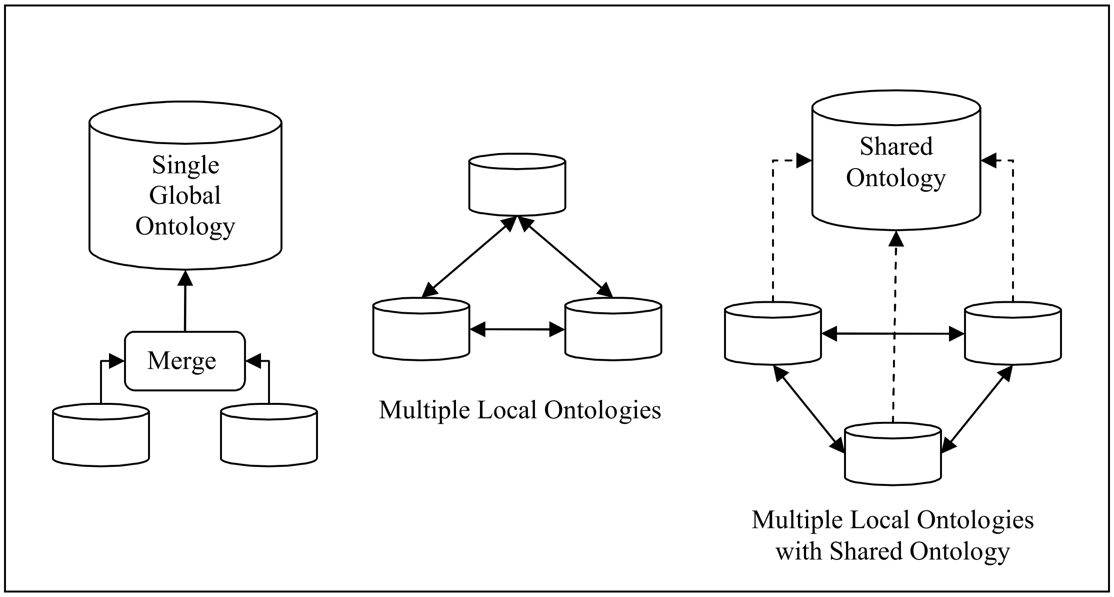

Single Global Ontology
Merge
Multiple Local Ontologies
Shared Ontology
Multiple Local Ontologies with Shared Ontology

Figure 4.3 Frameworks for Ontology Usage

Multiple Local Ontologies. The main idea of using multiple local ontologies is to maintain the original structure of information described by each local ontology. Each local ontology has their own definition of knowledge representation and vocabulary. They may also consist of a combination of several other smaller ontologies, as in single global ontology. The original consideration as compared to a single global ontology is that no centralized merging and constant monitoring is needed. Each local ontology can be modified or developed without major consideration of other ontologies which will eventually be used together. However, a lack of common vocabulary makes it rather difficult to compare several local ontologies at once. One solution to this problem is the use of mappings between ontologies (Weinstein and Birmingham, 1999). The mappings should be able to link semantically related objects together from different local ontologies. Though this solution is theoretically workable, the amount of mappings based on different views and granularity may be huge. The main reason for this is that many semantic heterogeneity problems that exist for a single global ontology are still here. Thus this may be a more costly process than that of a single global ontology (Wache et al, 2001).

Fusion Approach. This approach is designed to overcome the drawbacks of both the single global ontology and multiple local ontologies. It maintains the local ontologies, as in multiple local ontologies approach, but builds or uses an additional layer of shared ontology over them. The local ontologies are still described using their individual descriptions and structures. However, a new shared ontology is built or used over them to allow for comparison between the individual local ontologies. This shared ontology can be a vocabulary of terms obtained from the local ontologies (Wache et al., 1999) or an overview ontology by itself (Stuckenschmidt, 2000).

In the case of a shared vocabulary, the terms are obtained from the local ontologies by applying some operators on the primitives across ontologies and keeping their mappings. Primitives are categories of an ontology that cannot be defined in terms of other categories in the same ontology. An example of a primitive is the concept "Point" under Euclid's geometry. Since the shared vocabulary contains primitives and their mappings, the comparison between local ontologies becomes a much easier task than before. The way primitives under the shared vocabulary points to terms of local ontologies are dependent on individual implementation. Goh (1997) used an attribute value vector to represent the description of information from the local ontologies or context. Terms for the local ontologies is extracted from the pre-built shared vocabulary and data itself. Wache (1999) used labels for each type of information to represent its semantic meaning. The labels are then combined with existing primitives in the shared vocabulary by means of description logics operators.

In the case of an overview ontology, an external ontology is used to map or define the local ontologies. Very general ontologies, such as WordNet, can be used to assist mappings. Stuckenschmidt et al (2000) used a general ontology to cover all possible properties of the local ontologies. The general ontology consists of definitions and descriptions for properties, such as attribute value range for the concepts. The local ontologies are simply an instance or specification derived from the general ontology. This method allows local ontologies to be easily compared on the basis of the general ontology.

The main advantage of this approach is that new local ontologies can easily be added without the need of much modification. The use of a shared ontology or vocabulary also makes the local ontologies easily comparable. However, existing ontologies can sometimes be hard to reuse if they have to refer to a pre-defined shared ontology. This is because every term in the existing ontologies needs to be swap or replaced with terms used in the shared ontology.

5. Proposed Framework for Ontology Integration and Usage

From the reviews in the previous sections, we can clearly see that there are two important issues faced by existing frameworks for ontology integration and usage. Firstly, many existing works require human experts to provide help in knowledge integration. This can be in the form of interactive feedback for recommended integrations or crafting of domain knowledge for the systems to use. Therefore we need a way to automatically build this domain knowledge to overcome the reliant on human labor. Secondly, most works usually focus on using only one source of information, that is, they individually try to capture either the structure of the ontologies, meanings of concepts, or patterns among concepts. They do not effectively combine all sources of information or clearly differentiate between the importances for each piece of evidence. An existing core framework, introduced as part of a recent internal research (Ye et al, 2006), tries to solve this problem by considering the importance of different evidences in a hybrid approach. We further build upon this core framework to automatically extract different evidences from the web and perform automated ontology integration.

5.1 Existing Core Framework for Integration

The core framework for ontology integration follows from the observations that a combination of semantic and structural analysis is required for good results. Our main objective is to extract data models about the same domain from different web sources and integrate them under a global ontology. The idea in this framework follows that of a model which we term as the basic Diamond Model or IEC Model. This model has been designed as part of an internal research project (Ye et al, 2006). An overview of the model which looks like a diamond layout is given in Figure 5.1.

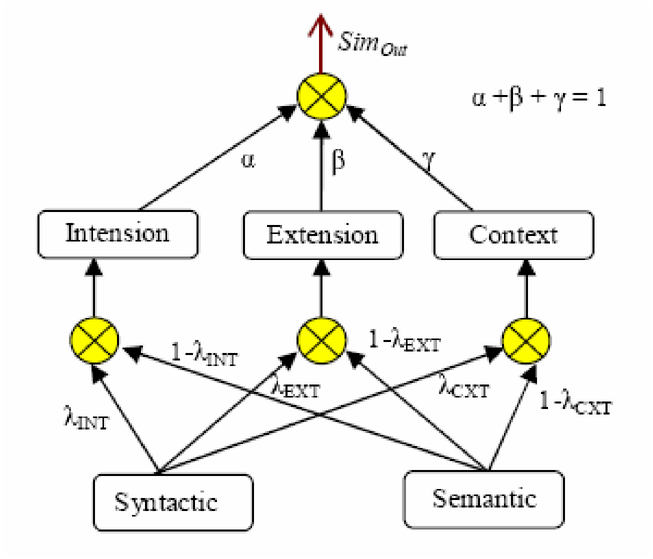


Figure 5.1 Overview of Core Framework for Integration (Diamond / IEC Model)

The key focus of this model is to capture different aspects of evidence which is present in the data instances or ontologies. The three main aspects of evidence are classified as follows:

1) The meaning or characteristics of a concept. This is usually given in the form of a definition, class label, or specific descriptions such as data types and constraints. We term this as the ***Intension*** (INT) or aim of the concept to be expressed.

2) The available properties of the concept. This include all the possible features of a concept which is not found under INT. The features are obtained from data instances that cover the concept, such as possible value range, cardinality, unit of measure, or enumeration of possible strings. We term this as ***Extension*** (EXT) or extended features from the concept.

3) The neighbourhood characteristics for the concept. This is obtained from the characteristics of surrounding concepts or nodes with respect to the target concept. The path from the root of the ontology to the target concept is considered and the features of surrounding neighbours are taken into account. The simplest way to use this is to consider the concept names along the path. We term this the ***Context*** (CXT).

The dividing of evidences into these categories allows us to use different methods for analyzing each piece of evidence and assign confidence to each method if necessary. It also allows us to measure any form of overlap among evidences and select the best evidence under different situations. This is more scalable and efficient in the long run. An example of INT, EXT and CXT is given in Table 5.1.

Intension (INT)	**Extension (EXT)**	**Context (CXT)**
Desktop	IBM, Acer, Toshiba, Dell, …	< Root >
CPU	Intel, Celeron, AMD, …	Desktop
Speed	G, M, GHz, MHz, …	Desktop -> CPU
Video Card	GeForce, Radeon, ATI,	Desktop
Desktop RAM	Meg, MB, G, GB, SDRAM, DDR, RD, …	Desktop
Video RAM	Meg, MB, G, GB, …	Desktop -> Video Card

Table 5.1 Example of INT, EXT, CXT

Since the evidences are available in different formats and variations, we can further analyze them at different levels. As such, we further divided evidences into two classes: semantic evidence and syntactic evidence. Each piece of evidence that we obtain either for INT, EXT, or CXT, can have a semantic meaning and also a syntactic pattern. Analysis at such levels provides us with the capability to compare between ontologies at different granularities. With these two views on evidence and 3 categories, we derive six meta-matchers for similarity measure:

1) *Semantic INT Matcher* 2) *Syntactic INT Matcher*

3) *Semantic EXT Matcher* 4) *Syntactic EXT Matcher*

5) *Semantic CXT Matcher* 6) *Syntactic CXT Matcher*

Using these 6 matchers, we can either perform a parallel one layer score combination or divide them into a two layer combination: (a) for each evidence category, and (b) for all categories. We have chosen the latter due to two reasons. First, each category of evidence may provide different predictions for matching. We will lose too much information if we were to consider them all together. Second, it is more intuitive for users to evaluate or improve the results based on different categories rather than all at once. Hence, we perform two levels of similarity score combinations. The first combination is done at the lowest level to combine semantic and syntactic scores for each category of evidence. A second combination is then done to combine similarity scores from different categories of evidence. In both levels of combinations, scores are combined using a linear weighting scheme. At the lowest level of matching, we combine semantic matching scores and syntactic matching scores for concept C_1 and C_2 using this formula:

$$\begin{aligned} Sim_x(C_1, C_2) = \lambda_x * Semantic\_Sim_x(C_1, C_2) \\ + (1 - \lambda_x) * Syntactic\_Sim_x(C_1, C_2) \end{aligned} \quad \ldots (1)$$

where $X \in \{EXT, INT, CXT\}$, λ_x is the weight for the combination where $0 \leq \lambda_x \leq 1$.

After obtaining the respective scores for each categories of evidence, we further combine the similarity scores using another linear weighted sum as follows:

$$Sim_{out}(C_1, C_2) = \alpha * Sim_{INT}(C_1, C_2) + \beta * Sim_{EXT}(C_1, C_2) + \gamma * Sim_{CXT}(C_1, C_2) \quad \ldots (2)$$

We normalized the weights and ensure that $\alpha + \beta + \gamma = 1,\ 0 \leq \alpha \leq 1,\ 0 \leq \beta \leq 1,\ 0 \leq \gamma \leq 1$.

5.2 Existing Similarity Matchers

5.2.1 String Matcher

This matcher is used to match strings on the lexical level when the concepts are defined using common labels with only very few variations. This matcher is designed to handle commonly misspelled word, singular-plural problems, and minor language styles, for example British and

American English. It is based solely on the idea of String Edit Distance. The String Edit Distance between two strings is the minimal cost of operations to transform one string into another. The operations include insertions, deletions, and substitutions of characters. Due to the fact that most strings in natural languages are short and the number of errors rarely exceeds 2, we have chosen to use the cost of less than or equal to 2 to indicate a match. Our experimental results show that this is quite reasonable.

5.2.2 Syntactic Matcher

This matcher utilizes the string matcher and an additional 3 sub-modules for matching. They are namely: Expression Template Matcher; Acronym Matcher; and Soundex Matcher. All these modules under the syntactic matcher are boolean matches (either a hit or a miss). The purpose of the syntactic matcher is to capture the similarity in generic patterns and some form of isomorphism. Most of the modules are based on simple heuristic rules or checking with respect to lists or domain independent knowledge. Users or experts may create better or new rules based on their required domain to further improve the results.

Expression Template Matcher. This module depends strictly on a list of predefined generic templates such as phone numbers, codes, dates and time, enumeration, noun phrases and some specific patterns. For example, “digits + units” matches “55 kilograms”, and time does not match with phone numbers. The expression template also utilizes generic lists which are mined from the web. Specifically, unit of measures (UOM) and some domain specific words are collected from the web.

Acronym Matcher. This module basically matches phrases or strings which can be represented in the form of acronyms. Acronyms are essentially words formed from the initial letters of a multi-word name. As such, the use of a simple set of heuristic rules is quite sufficient. However, for completeness, we also supplemented the acronym matching module with lists of common abbreviations and acronyms. For example, we can match "ACL with Association for Computational Linguistics" and "IBM with International Business Machines Corporation".

Soundex Matcher. This module handles the inconsistency in writings when elements name are written differently but pronounced similarly and mean the same thing. Users may have misspelled words based on their own perception of pronunciation or used abbreviation for terms. For example, "Send 2" with "Send to" and "Smyth" with "Smithe". The module keeps a set of term pairs for simple transformation, but also uses a modified Soundex algorithm (Holmes, 2002) to match unseen pairs. The key idea is to map input strings to a new string that represent their phonetics, and compare those strings for an exact match. Our observations have shown that Soundex algorithm only works well for medical terms, names, or unknown words.

In the case of a match for any syntactic module, a score δ is added to the overall score for the matcher. The final score is then normalized before using for score combination in the IEC model.

5.2.3 Semantic Matcher

WordNet Matcher. This matcher takes into account the semantic closeness of elements based on the ratio of their term co-occurrence in WordNet relations. The assumption we make here is that terms which are semantically related to one another are often defined using similar sets of words. For example, "notebook" is defined in WordNet as "a small compact portable computer" while "laptop" is defined as "a portable computer small enough to use in your lap". By removing stop-words (a list of common or general terms, for instance prepositions and articles), we can easily observe that "notebook" and "laptop" have quite a number of overlapping words. Hence it is reasonable for us to use term co-occurrence for measuring semantic relatedness (Lesk, 1986). In WordNet, we measure term co-occurrence using the following relations:

1) Gloss: an explanation or definition of a word in a text. Examples may be included.

2) Hypernymy (is-a relation): the semantic relation of being super-ordinate or belonging to a higher rank or class

3) Meronymy (has-a relation): the semantic relation that holds between a part and the whole or part to whole relation

The term co-occurrence is computed by tabulating the number of common words as in Baziz et al (2004). An alternative of using WordNet package in Perl for measuring semantic relatedness seem to perform equally well. The package was developed based on papers by Hirst and St-Onge (1998), Jiang and Conrath (1997), Leacock and Chodorow (1998), Banerjee and Pedersen (2002), Lin (1998) and many others. Further details of the package are available at http://search.cpan.org/dist/WordNet-Similarity/

5.3 Drawbacks of Existing Core Framework

Though the existing core framework is reasonably comprehensive, there are three main drawbacks. These drawbacks are very serious in that it limits the system in performing to its full potential due to its inability to capture semantic or contextual information effectively. In the existing basic diamond model, contextual information depends mainly only on the usage of WordNet. As such, it encounters the following problems:

1) There is a restriction on the possible usage of words to obtain semantic-inherent data. Though WordNet is rather comprehensive, it is still a limited dictionary based system. There are many examples where a word cannot be found in WordNet. Just to name a few (as of Version 2.1), "Video Card", "biometric", and "chlorinator" all could not be found. In particular, it is very difficult to use WordNet for phrases or strings with more than one word. Unless the meaning of each word is considered separately and merge again, you could not obtain any semantic information from them. Moreover it

should be noted that combining meanings of words separately may not represent the meaning of the words as they are given. For example, “Rock and Roll”. A solution to this problem is the use of web information. The web consists of a wealth of data which is increasing everyday. People are writing new guide books, publishing new articles on Wikipedia or creating new web sites for search engines to index. This frequently updated large source of information allows more semantic data to be obtained.

2) WordNet is too general and at times we are not sure which senses can give the true meaning of a given word. Similar to the first problem, web information in the form of guide books, search results and Wikipedia is one possible solution. There is usually more relevance for ontology integration with such information. This is especially true for guide books which mention specifically related things for a particular domain.

3) WordNet is a closed dictionary and unable to utilize data redundancy freely available on the web. Data redundancy can assist in performing a much better match in most cases. In addition to the many articles in Wikipedia, there are also many guide books available online. These documents if describing things under the same domain have a tendency to repeat an important point or issue several times. For example, “CPU” and “Intel Processor” may be used exchangeably across several guide books about computers. In addition, they are commonly mentioned together with other important terms, for instance “Memory”, “Video Card”. By exploiting this redundancy, we can effectively capture the correct semantic context for the terms to be matched. An additional consideration may be to exploit the redundancy across web information together with WordNet.

In addition to these drawbacks for the existing framework, the contextual matching is also too simplified and may not be accurate. This is because context here is considered only in the form of term matching from the concept node up to the root. This simplified matching has problems and will be discussed more in depth in Section 5.5.

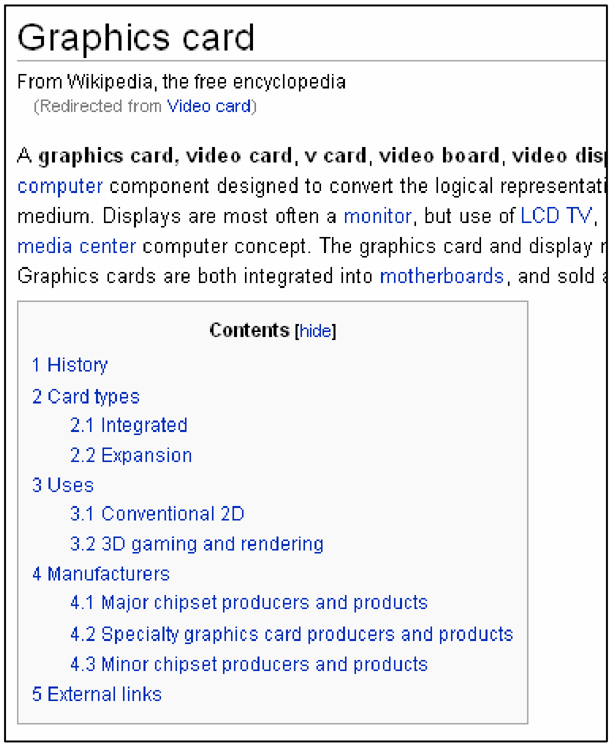
5.4 Web-Based Similarity Matchers

In order to fully capture the semantic meanings of the concepts to be matched, WordNet itself is not sufficient. Hence, we introduce 2 other semantic matchers. They are, namely, Guides-Wikipedia Matcher and Web Instance Matcher. The main purpose of these semantic matcher is to capture the meaning or semantic relatedness between two given concepts or elements. The key idea is to utilize external resources to assist in measuring semantic relatedness. In particular, as complement to the semantic network (WordNet), online guides and resources, and web search results are used. One of the major aims of this research is to investigate the effect of using these readily available external resources, specifically web knowledge, to aid in ontology integration. The scores obtained from all the semantic matchers are combined using a linear weighted sum as in Equation 2.

5.4.1 Guides-Wikipedia Matcher

Online guide books and Wikipedia (http://www.wikipedia.org/) are good sources of external information which can be utilized in helping us measure the semantic relatedness. The key idea is similar to WordNet matcher where terms overlaps are considered. This matcher makes use of online guide books and Wikipedia results obtained for multiple domains such as Computers, Diamonds and Cameras. The matching for guide books and Wikipedia follows 3 procedures: a) collecting of instances corresponding to the concepts, b) matching key features from the instances, and c) matching full text based on LSA classifier.

Collection of Instances. Instances from the Wikipedia are collected simply by posting a query into the Wikipedia website. Guides books on the other hand are downloaded beforehand and stored on the local machine. This stage involves tagging the corresponding documents or instances with the concepts. As for Wikipedia, it is obvious that the returned results will belong to the submitted concept. However for guide books, we need to determine which guide books or which section belongs to the concept to be matched. First,



Graphics card

From Wikipedia, the free encyclopedia
(Redirected from Video card)

A **graphics card**, **video card**, **v card**, **video board**, **video dis**
computer component designed to convert the logical representati
medium. Displays are most often a monitor, but use of LCD TV,
media center computer concept. The graphics card and display r
Graphics cards are both integrated into motherboards, and sold a

Contents [hide]
1 History
2 Card types
2.1 Integrated
2.2 Expansion
3 Uses
3.1 Conventional 2D
3.2 3D gaming and rendering
4 Manufacturers
4.1 Major chipset producers and products
4.2 Specialty graphics card producers and products
4.3 Minor chipset producers and products
5 External links

Figure 5.4.1 Wikipedia Result for "Video Card"

we extract the table of contents or possible sections from the guide books using a set of simple heuristic rules, for example extra large fonts, title of page or specific tags such as headers <H1>, <TH>, listings <OL>, <UL> and links ".htm#video". Then we perform the matching of the concepts or terms with the table of contents or section headlines. The matching is done using a modified version of the syntax matcher as discussed in Section 5.2.2. Whenever there is a match with the headlines, that particular section or page will be assigned to the concept. The procedure of detecting sections applies to Wikipedia as well in cases where a table listing is available (see Figure 5.4.1).

Matching Key Features from Instances. Instead of performing full text document matching between instances for two concepts, this step aims to leverage on any key features we have from the instances. In particular, the headlines or words from the table of contents or section headings that may provide some form of contextual semantic meaning to be match (see Figure 5.4.2 on

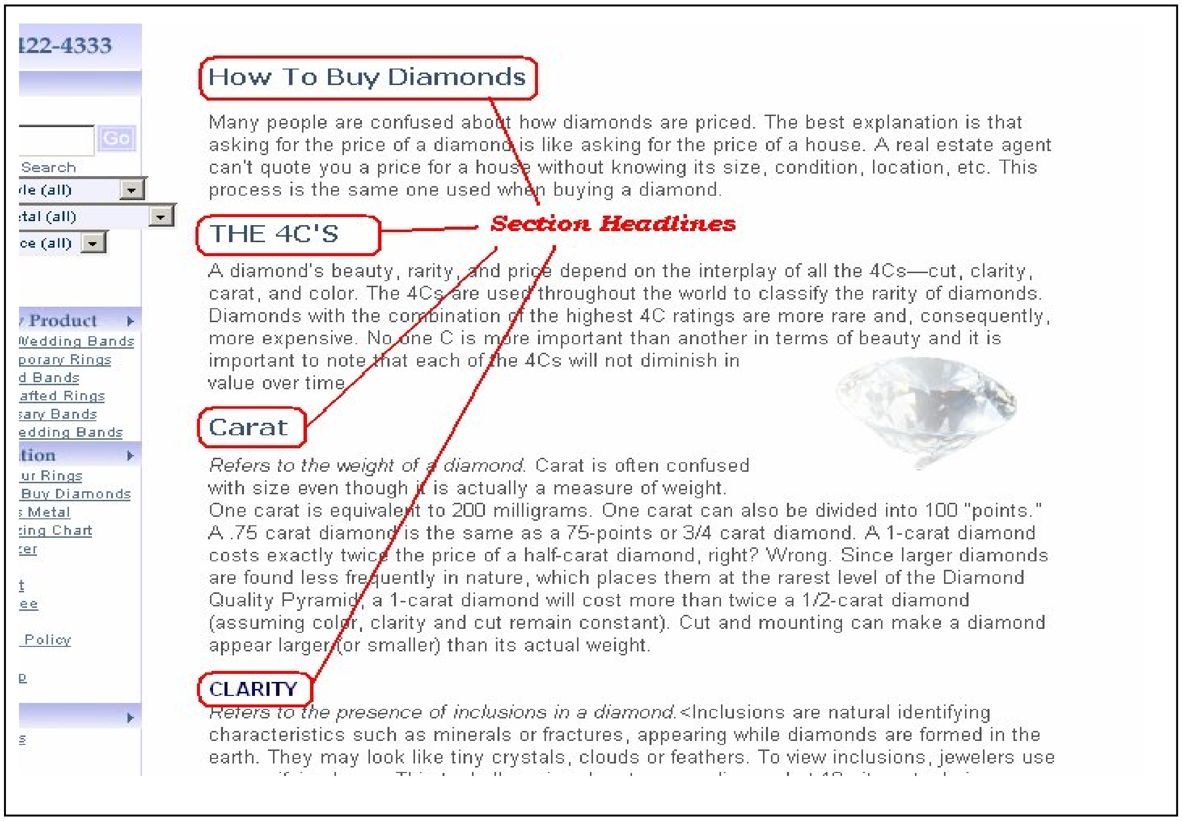
Diamond concept). The intuition is that important points or very related concepts may be mentioned together with our target element as headlines. Authors tend to divide concepts into key sections and use headlines to distinguish them. For example, one desktops guide has processors, memory, hard disk, video cards mentioned as sub-sections. Similarly in another guide we may have processors, memory, hard disk and graphic cards. In such cases, we can effectively measure the similarity between “video cards” and “graphic cards” to perform a mapping. We measure the amount of overlap in this case by using the idea of mutual information (MI) on the terms for each concept. Given two random variables X and Y, MI measures the information about X that is shared by Y. If X and Y are independent, then X contains no information about Y and vice-versa, thus their mutual information is zero. If X and Y are identical then all information conveyed by X is shared with Y (ie. knowing X reveals nothing new about Y and vice versa), therefore the mutual information is the same as the information conveyed by X (or Y) alone, namely the entropy of X (or Y).

The mutual information is based on the following mathematical formulas:

$$MI(x,y) = \log\frac{P(y \mid x)}{P(y)} = \log\frac{P(x \mid y)}{P(x)} = \log\frac{P(x,y)}{P(x)*P(y)} \qquad \ldots (3)$$

$$I(X,Y) = \sum_{y\in Y}\sum_{x\in X} P(x,y)*MI(x,y) = \sum_{y\in Y}\sum_{x\in X} P(x,y)*\log\frac{P(x,y)}{P(x)*P(y)} \qquad \ldots (4)$$

where *x* is a term in X, y is a term in Y, and I(X, Y) = 0 if and only if X and Y are independent. A high value here would mean a likely match in semantic relatedness for the two concepts.



How To Buy Diamonds

Many people are confused about how diamonds are priced. The best explanation is that asking for the price of a diamond is like asking for the price of a house. A real estate agent can't quote you a price for a house without knowing its size, condition, location, etc. This process is the same one used when buying a diamond.

THE 4C'S

A diamond's beauty, rarity, and price depend on the interplay of all the 4Cs—cut, clarity, carat, and color. The 4Cs are used throughout the world to classify the rarity of diamonds. Diamonds with the combination of the highest 4C ratings are more rare and, consequently, more expensive. No one C is more important than another in terms of beauty and it is important to note that each of the 4Cs will not diminish in value over time.

Carat

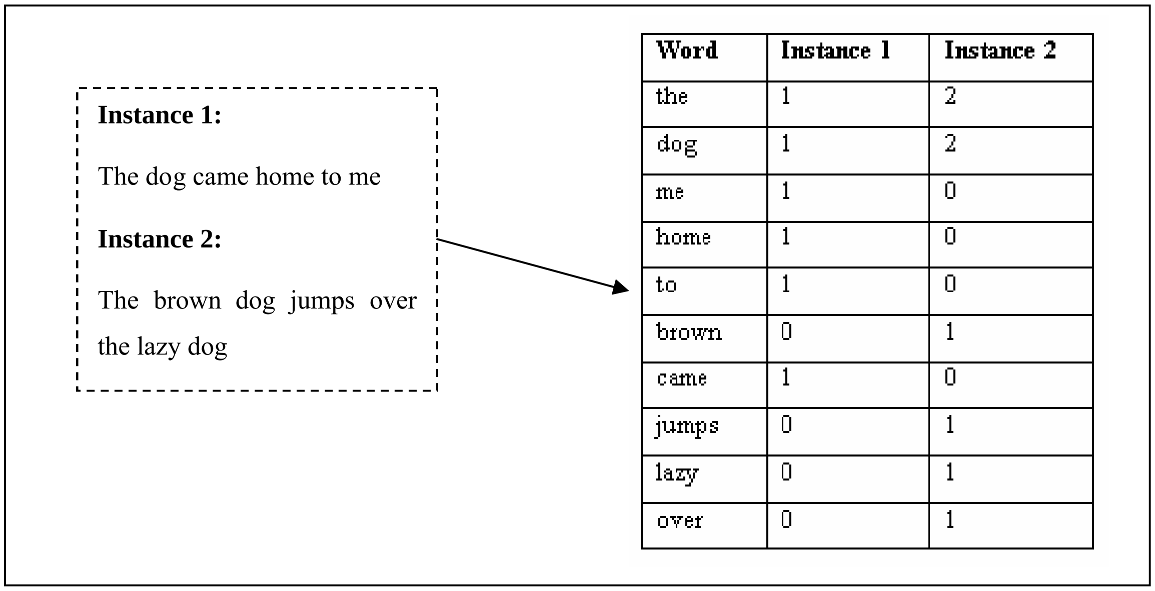
Refers to the weight of a diamond. Carat is often confused with size even though it is actually a measure of weight. One carat is equivalent to 200 milligrams. One carat can also be divided into 100 "points." A .75 carat diamond is the same as a 75-points or 3/4 carat diamond. A 1-carat diamond costs exactly twice the price of a half-carat diamond, right? Wrong. Since larger diamonds are found less frequently in nature, which places them at the rarest level of the Diamond Quality Pyramid, a 1-carat diamond will cost more than twice a 1/2-carat diamond (assuming color, clarity and cut remain constant). Cut and mounting can make a diamond appear larger (or smaller) than its actual weight.

CLARITY

Refers to the presence of inclusions in a diamond.<Inclusions are natural identifying characteristics such as minerals or fractures, appearing while diamonds are formed in the earth. They may look like tiny crystals, clouds or feathers. To view inclusions, jewelers use

Figure 5.4.2 A Guide Book for "Diamonds"

Matching full text based on LSA. Latent Semantic Analysis (LSA) is a statistical model of word usage that permits comparisons of the semantic similarity between pieces of textual information (Foltz, 1996). LSA was originally designed to improve the effectiveness of information retrieval methods by performing retrieval based on the derived "semantic" content of words in a query as opposed to performing direct word matching. This approach avoids some of the problems of synonymy, in which different words can be used to describe the same semantic concept. The primary assumption of LSA is that there is some underlying or "latent" structure in the pattern of word usage across documents, and that statistical techniques can be used to estimate this latent structure. We have thus chosen to use LSA to help model the semantic content on the full text document, in particular the web information. Before applying LSA, web pages are passed through a preprocessing stage to replace HTML tags with default labels (using a transformation table) and removal of stop-words.



Instance 1:

The dog came home to me

Instance 2:

The brown dog jumps over the lazy dog

Word	Instance 1	Instance 2
the	1	2
dog	1	2
me	1	0
home	1	0
to	1	0
brown	0	1
came	1	0
jumps	0	1
lazy	0	1
over	0	1

Figure 5.4.3 Example Input Matrix for LSA

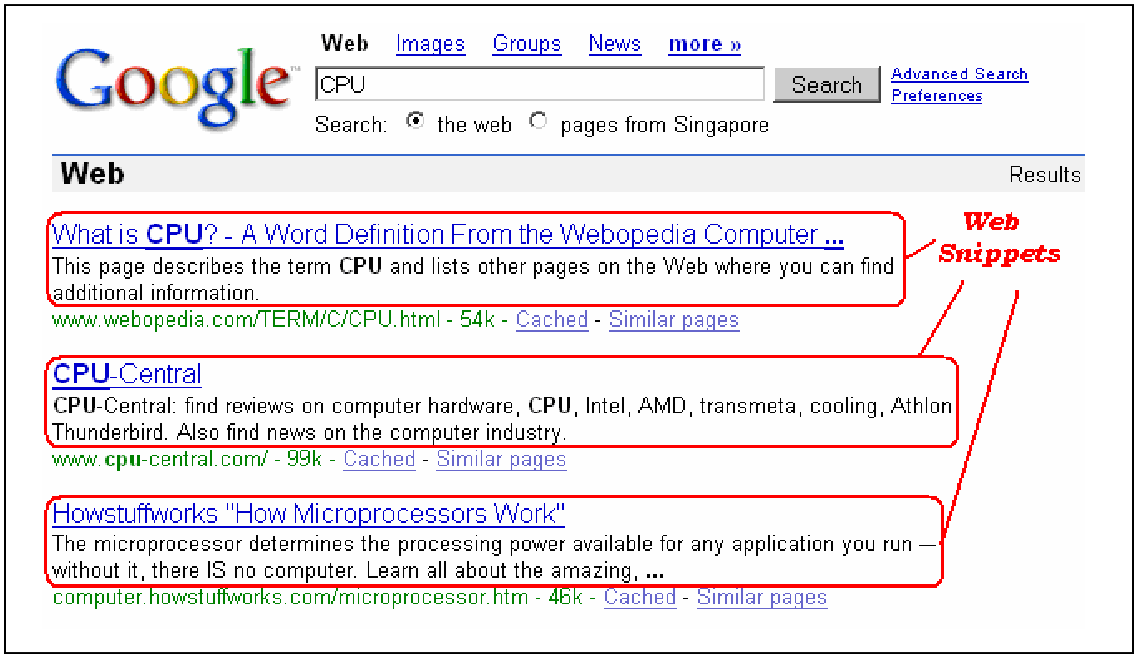
LSA first generates a matrix of occurrences of each word in each document. Then it applies singular-value decomposition (SVD) on the matrix. The SVD scaling decomposes the word-by-document matrix into a set of *k* orthogonal factors from which the original matrix can be approximated by linear combination. Instead of representing documents and terms directly as vectors of independent words, LSA represents them as continuous values on each of the *k* orthogonal indexing dimensions derived from SVD analysis. Since the number of factors or dimensions is much smaller than the number of unique terms, words will not be independent. For example, if two terms are used in similar documents, they will have similar vectors in the reduced-dimensional LSA representation. One advantage of this approach is that matching can also be done between two pieces of textual information, even if they have no words in common. The result of SVD is a *k*-dimensional vector space containing a vector for each term and each document. The location of term in the vectors reflects the correlations of their usage across documents. Similarly, the location of document in the vectors reflects correlations of the terms used in the documents. We then compare the semantic similarity of the documents using their dot products in this new space. Dot product provides a measure of similarity between two vectors, with 1 being a perfect match. Thus, by determining the vectors of two pieces of textual information, we are able to determine the semantic similarity between them. An example of a possible input matrix to LSA is given in Figure 5.4.3.

5.4.2 Web Instance Matcher

This matcher is similar to the third procedure for Guides-Wikipedia Matcher, matching full text based on LSA. The main difference is in the source of information used. Here we use search engines to provide us with semantically related documents. A web instance is a collection of web snippets for a given concept. We separated this matcher from the above for two reasons:

1) Guides and Wikipedia provide vertical search results, ie. answers correspond almost directly to the query, while normal search engines return very general answers.
2) Guides and Wikipedia are usually better structured, while normal search engines may return answers that are not in standard natural language structure. For example, a search for cars on Google returns "Lightning McQueen, a hotshot rookie race car driven to succeed, discovers that life is about the journey, not the finish line, when he finds himself…"

The instances for this matcher are currently obtained from the top 100 document snippets returned by the Google search engine. As search engine results starts to diverge, we may consider using results across search engines in the future (Sherman, 2005). Figure 5.4.4 shows an example of Google web snippets for a search on "CPU".



What is CPU? - A Word Definition From the Webopedia Computer ...
This page describes the term CPU and lists other pages on the Web where you can find additional information.
www.webopedia.com/TERM/C/CPU.html - 54k - Cached - Similar pages

CPU-Central
CPU-Central: find reviews on computer hardware, CPU, Intel, AMD, transmeta, cooling, Athlon Thunderbird. Also find news on the computer industry.
www.cpu-central.com/ - 99k - Cached - Similar pages

Howstuffworks "How Microprocessors Work"
The microprocessor determines the processing power available for any application you run — without it, there IS no computer. Learn all about the amazing, ...
computer.howstuffworks.com/microprocessor.htm - 46k - Cached - Similar pages

Figure 5.4.4 Google Snippets for "CPU"

5.5 Enhanced Concept Matching

Given all concepts pairs from two ontologies, we perform concept matching for integration after computing the overall score for each pair obtained from Eqn 2 (under Section 5.1). This matching can be simplified into a graph matching problem where we need to find the corresponding nodes or vertices that match, incorporating any form of isomorphism if present. The main objective is to find a *fitness function* which maps possible vertices of Ontology *A* to Ontology *B*. The common sub-graph isomorphism problem, which is to determine the largest graph *H*, that is a sub-graph of both *A* and *B*, has been shown to be NP-complete (Brooks, 2003). However, in our context, there is no actual need of finding a complete common graph. Some vertices are impossible to match, for example concept "CPU" with "Animals". Our main focus is thus to compare the nodes to be matched and the scores contributed by their neighbouring nodes (nodes from the root). One point to note is that scores contributed by neighbouring nodes is already taken into consideration by the CXT evidence matching. We shall go in depth on how we actually model this.

5.5.1 Measure of Context

When measuring syntax and semantic information for CXT, we transverse each node from the root up to the concept to be matched. For example, in order to match concept *E* and *Q* in two given concept trees of "*A->B->C->D->E*" and "*M->N->O->P->Q*", we first match *A* with *M*, then *B* with *N*, *C* with *O*, *D* with *P*, and finally *E* with *Q*. Besides the targeted concepts themselves, all nodes along the path are matched as well. This is relatively "*easy*" to compare when the two ontology paths are of the same length. However, an issue arises when the paths' lengths are different. Moreover, a direct comparison of nodes along the path, at a fixed position, can miss out context information which may be misplaced by just by one or two position. Take for example the trees to match "Lion" given in Figure 5.5.1. If we were to do a direct bottom-up comparison of nodes along the path, *Cats* at L-4 for *A* will match with *Mammals* at L-3 for *B*, and *Mammals* at L-3 for *B* will match with *Animals* at L-2 for *A*. If we perform a top-down matching, the first two nodes will match, but we will lose out the information provided by *Cats*. An alternative is to permutate all possible pairs and combine their context scores. However, a) there is no good way to combine the scores, and b) permutating through the pairs is not reasonable as it would seriously increase computation time. Furthermore, noise level can increase since many combinatory pairs may not be related in the full permutation. A simple solution is just to choose matching only for nodes which exist on the same level, for example both *Mammals* at L-3. However, this may not be a complete solution. This is because the issue is further complicated by the fact that there may be more than 1 "misplaced" position along the path (insert node "*Vertebrates*" before *Mammals*). Many ontologies are not defined in the same way and the granularity levels may vary quite a lot. Hence, the levels of node may be altered in such a way that the sequence or chain does not present a sufficiently good overlap.

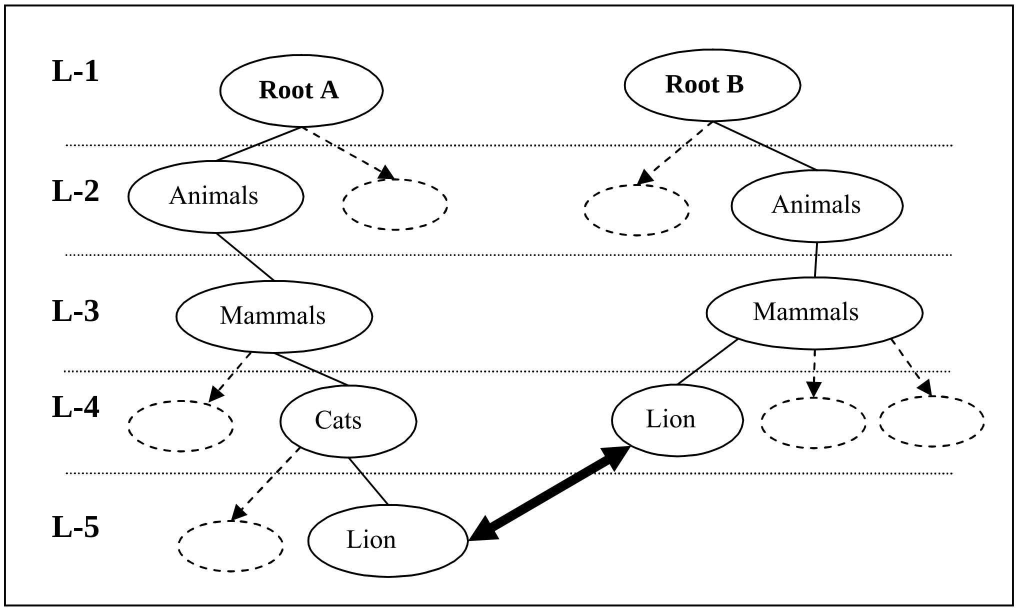


Figure 5.5.1 Ontology Trees about Animals

As such, we propose a sliding window approach for matching elements at each level. This solution is inspired by the problem in DNA sequence matching (Dunham, 1999). The sliding window approach is chosen for three reasons: 1) It allows us to handle "misplacement" of nodes within the window. 2) It is much more efficient and reasonable than full permutation. 3) It allows us to capture more semantic meaning inherent in the ontology hierarchy. For example in Figure 5.5.1, *Mammals* from *B* can match with *Mammals* from *A* and also with *Cats* from *A*. The additional match with *Cats* will provide more overlap information to boost the semantic meaning for the mapping. Conversely for the case when two ontology hierarchies are totally different, for example Ontology about Animals and Computers, this additional non-overlap information will ensure a mismatch, even if there is an accidental match in the targeted concept pairs.

Since node "misplacement" rarely occurs too far away, we selected a sliding window of size 3 for the matching. In practice, this size can be altered depending on the ontologies to be matched. To illustrate how we perform the matching, we use the example in Figure 5.5.1 again. First, *Lion* from A is matched with that of *B*. Next, *Cats* from *A* is matched with *Lion* and

Mammals from *B*. Then, *Mammals* from *A* is matched with *Lion*, *Mammals* and *Animals* from *B* (window of size 3). And finally *Animals* from *A* is matched with *Animals* and *Mammals* from *B*. It does not matter if the match is top-down or bottom-up. In view of the fact that nodes further away from the targeted concept does not provide as much information as those nearby, we compute and combine the scores along the path using a linearly decaying weight function. The overall formula for scoring is given as follows, with ℓ , $0 \leq \ell \leq 1$, being the weighting for neighbourhood scores:

$$\begin{aligned} Context\_Sim(C_1, C_2) &= \ell * Sim\_Exact(C_1, C_2) \\ &\quad + (1-\ell) * \sum_{i=2}^{m} [(1 - \frac{(i-1)*(1-\theta)}{i}) * Sim\_Window(C_i)] \quad \ldots (5) \end{aligned}$$

where *i* is the number of levels away from the target concept, *m* is the maximum depth of the tree starting from the target concept, and θ is the restricting value between 0 and 1. If θ is set to 1, it becomes equivalent to summing up all the scores for the sliding window under the each concept C_i. *Sim_Exact(.)* here is the function which does matching on the concept node itself, without considering any neighbours or the path (see Eqn 2, under Section 5.1). *Sim_Window(.)* is the function which computes the similarity based on the 3-window, using *Sim_Exact(.)* for each pairing match and averaging. θ is currently set to 0.35 based on empirical results. Note that the depth of path starts from 1.

5.5.2 Ontology Mapping

After computing all the similarity scores for each pair of concept nodes to be matched, we select the best mapping between ontologies {O$_i$} using the algorithm in Figure 5.5.2. First we initialize the resultant matching set *S* as empty. Second, we select the best matching pair of concepts among the ontologies and add to set *S*. This process is repeated until the similarity score for the best matching pair falls below a predefined threshold. Third, we remove any conflicts between match pairs in set *S* by keeping only the pairs with a larger similarity score. Conflicts in the last stage are mainly mappings of *N* nodes from the same ontology pointing to a common node in the other ontology.

```
● Initialize the resultant matching set, $S$. Set $S$ = null. Let $A$ be an arbitrary ontology.
● for each $A$, $A \in \{ O_i \}$ perform:
    Let $e$ = root of $A$ , $M = \{ O_i \}$, E = set of elements to be match = null
    while(true) do{
        - Let $e_k \in \{ K \mid K \in \{ M - A \}$, where K is an ontology}
        - Compare $e$ with all elements $\{ e_k \}$ from other ontologies $\{ M - A \}$ and obtain
          the best match element $e_k^m$ for each ontology, ie. find all $e_k^m$ such that:
            $e_k^m = \arg\max \{ Sim_{out}(e, e_m) \mid Sim_{out}(e, e_m) > \text{threshold } \omega,$
                    match pair $< e, e_m > \notin S$, $e_m$ is not in any previous match pair of S }
        - Add all match pairs $\{ < e, e_k^m > \}$ into S.
        - Add direct children of $e$ to E. ie. $E = E \cup \{$ children of e $\}$
        - Set $e = \{ c \mid c \in E \}$, $E = E - \{c\}$
        - if($e$ = null) break;
    }
● for(each match pair $< a_j, b_j >$ in S){ // to remove conflicts with $N$ to 1 mapping
    for(each match pair $< a_i, b_i >$ in S, where $< a_i, b_i > != < a_j, b_j >$){
        if($a_i = a_j$ & $b_i, b_j$ are from same ontology, $A$, OR
          $b_i = b_j$ & $a_i, a_j$ are from same ontology, $A$) // 2 element map from $A$ to same
          point
          - Remove match pair with smaller score from S
  } }
● Return S
```

Figure 5.5.2 Ontology Mapping

5.6 New Framework for Ontology Integration and Usage

This section serves as an outline for targeted research and possibly a final program which enables ontology creation, integration and also usage. Figure 5.6.1 shows the targeted framework for our research.

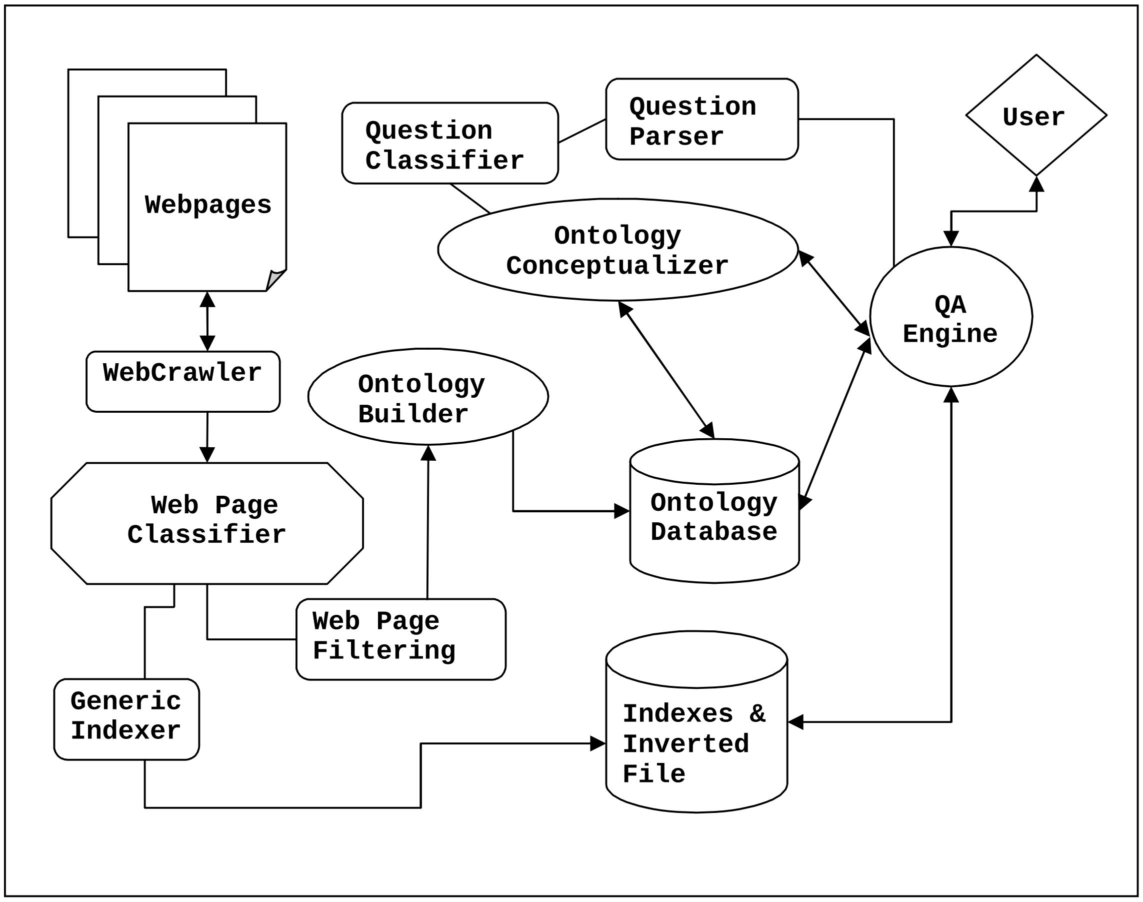


Figure 5.6.1 Overview of Targeted Framework

The first stage of this framework involves automatically collecting data from online web pages. This is to be done by an automatic WebCrawler. The web pages collected will then be classified into categories belonging to different domains or possibly contain different data models. At the same time, web pages will be indexed into the system using a generic indexer. This is to support simple search engine queries and is optional for our system. The key feature will be our Ontology

Builder and Ontology Conceptualizer. The Ontology Builder first takes in web pages from different domains and tries to mine data models from them. Then it performs ontology integration for the different data models and organizes data instances together with their merged schemas into an Ontology Database. At the end of the research project, the Ontology Database is structured using a fusion model for different domains (see Section 4.3). This Ontology Database is read by the Ontology Conceptualizer to select key features for comparison and provides an effective summary for the user. The user interacts with the system through the Question Answering (QA) Engine, which is supported by the Question Parser and Question Classifier. We are currently in the process of evaluating some of the sub-components.

6. Testing and Evaluation for New Framework

This chapter documents the testing and evaluation on various components of our overall targeted framework. The test data is obtained from 3 main corpora: a) I3CON ontology dataset (Hughes, 2004); b) TEL-8 dataset of UIUC Web Integration Repository; and c) Online web pages collected over several base websites. Table 6.1.1 shows the data distribution from different sources.

Corpora Name	**Ontology (Number of Ontologies)**	**Number of elements per ontology**	**Ontology Depth**
I3CON	Animals (2), Soccer (2), Hotel (2), Network (2), People (2), Pets (2), Russia (2)	Min (4), Max (>200), Average (46.7)	2-4 layers
TEL-8	Books (66), Airfare (45)	Min (5), Max (32), Average (11.5)	2-4 layers
Online	Computer (6), Hard Disk (6), Digitial camera (6), MP3 player (6), Video Card (6), Diamond (6), Others (6)	Min (6), Max (>500), Average (50.3)	2-5 layers

Table 6.1.1 Data Distribution across corpus sources

The I3CON data (Hughes, 2004) is a typical corpus designed for standard ontology research. It is not a very large data set. The TEL-8 dataset (at http://metaquerier.cs.uiuc.edu/repository) consists of data models from 2 specific domains, namely Books and Airfares. They are extracted using a web query interface for the repository. These two corpora are converted into XML format using a preprocessing module before testing. The last corpus is a collection of 83.5 GB of web pages collected over several base websites, a total of 3,833,179 web pages. The websites used and the distribution of web pages are as shown in Table 6.1.2. This is collected using an automated WebCrawler. The data in the last corpus are selectively classified into several main categories and effectively only a small collection out of these is used. In our preliminary experiments, only 6 main categories are extracted into XML format for the testing of ontology integration, they are namely Computers, Hard Disk, Digital Camera, MP3 players, Video Cards and Diamonds. The extraction is done based on the system described in Ye and Chua (2004). These models are extracted from the structured content pages and have much more variation in structure, granularity, and terminology than the previous two corpora.

Websites	**Total Number of Webpages**	**Size (MB)**	**Defined Categories**
Price.com	531696	11942.39063	33
Pricerunner.com	549976	11815.89063	56
Pricegrabber.com	688248	14786.57813	79
Shopping.com	650547	15882.49512	43
Yahoo Shopping	656901	14113.10742	21
MSN Shopping	755811	16976.22363	93

Table 6.1.2 Data Distribution across web sources

Domain	**Number Guide Books**
Animal	2
Airfare	1
Book	2
Computer	9
Hard Disk	1
Digital Camera	7
MP3 Player	6
Video Card	3
Diamond	8
Total	39

Table 6.1.3 Data Distribution for Guide Books

Main Guide Book Sources
www.pcworld.com
www.about.com
www.buyerzone.com
www.viewz.com/shoppingguide/
pages.ebay.co.uk/buy/guides/
www.thediamondbuyingguide.com/
www.hsamuel.co.uk/webstore/static/guides/
www.epinions.com/buyers_guide/
www.pcworld.com/howto/bguide/

Table 6.1.4 Main Sources for Guide Books

In addition to the previous datasets mentioned, we collected a total of 39 online guide books to support evidence matching. These serve as a form of publicly available web knowledge or global ontology which we can utilize. It is one of the main components used for evaluating whether web knowledge will help for ontology integration. Table 6.1.3 shows the distribution of online guide books for the different domains which we have chosen to work on, while Table 6.1.4 lists the sources where we obtained the guide books.

In all the experiments which require parameters to be estimated, data are divided into two sets, the training set and the testing set. The parameters to be estimated in these various experiments are set based on ten-fold cross validation of the training set in an active learning approach. The parameters are first randomly fixed. Then validation of results on the training set is done by randomly dividing it into two subsets (80% for training, 20% for testing), and repeating the process ten times. This is to ensure consistency in the results using the chosen parameters. The parameters are then changed manually according to past observations and evaluated again. The whole process is repeated until we deem the results obtained as satisfactory. Subsequently, the corresponding parameters are used for our actual testing.

6.1 Query Classification

This section documents the results obtained for classifying an input query belonging to a given domain, for example "Computer, Hard Disk or Video Cards". The set of web pages collected from different sources are filtered and pre-assigned with different category names. These category names are what the users may be interested in. Each category contains 100 web pages randomly selected from different web sources. There are a total of 6 categories or domains, which are as listed in Table 6.1.3. The objective of the query classifier is to map the query to a correct domain or category the user wants. The experiments done in this section are used to evaluate the basic methods used in query classification and recommend one to be used in an actual ontology query system.

Unlike plain text, web pages contain markup tags which may or may not be useful. In order to ease the analysis of web pages, we replace HTML tags with a transformation table and represent each web page as a bag-of-words. The frequency of words in these web pages are tabulated and collected for each category. In addition, we use a simple heuristic weighting scheme to boost the importance of words when they occur in important places. This is because web document has a richer structure compared to a plain text document. It has many important features like Title, Headings or Meta-Data. Words that have such features are given a weight boost for each occurrence, for example the word "Computer" under the Title is given a 2.5 weight boost, while the same word under common text is only having 1.0 weight boost (or no boost). The list of weights for several important feature types is given in Table 6.1.5. They are based on the results obtained in Golub and Ardo (2005).

Data Type	Headings	Meta-Data	Title	Text Contents
Weight Boost	2.1	1.5	2.5	1.0

Table 6.1.5 Weight Boost for different HTML elements

Three experiments are done for the query classification: 1) classify using the *basic* original input query or keywords, 2) classify using basic original input query and terms which are *synonyms* of the keywords, and 3) classify using basic original input query and co-occurring terms obtained from Google search results, a form of *web expansion*.

The main difference between the 3 experiments for this section is the query expansion input passed to the classifier. For *Basic*, only the parsed keywords are used. For *Basic + Synsets*, both the keywords and a list of top expanded synonyms from WordNet are used (top 5 synonyms for each keyword). For *Basic + Web Expansion*, the keywords and a list of top 10 most frequently co-occurring terms from Google search results are used. The evaluation is done using queries formed with a random selection of 50 unrelated keywords/phrases from a dictionary, for example "fire", "transport ship", "mechanical devices", "dynamic programming"; and 50 manually selected assignable/classifiable keywords/phrases, for example "video card", "digicam", "music player", "diamond selection". We measure the results according to whether it is correctly assigned.

In all 3 experiments under this section, the main classifier uses a vector space model to represent the web documents and measure the similarity of the keywords with the documents. This is done corresponding to their relative co-occurrence with terms in the documents for a particular category. The basis for scoring of terms is given by a modification to the classic weight scheme used in Salton's Vector Space Model (Salton, 1971):

$$w_i = B_i * tf_i * \log(\frac{D}{df_i}) \qquad \ldots(6)$$

where:

- B_i=weight boost for the term *i* depending on its HTML element type
- tf_i=term frequency (term counts) or number of times a term *i* occurs in a document
- df_i=document frequency or number of documents containing the term *i*
- *D*=number of documents in the database

The $log(D/df_i)$ term is also known as the "inverse document frequency", IDF_i. This is a measure of the shear volume of information (and entropy) associated to a term within a set of documents.

The category with the highest summed scores, above a threshold γ, across all its instances (web documents) will be the assigned category for the input keywords. The main reasons for the choice of a vector space model are its simplicity and the proven effectiveness in general categorization tasks (Becker and Kuropka, 2003).

Classifier Input	**Precision**	**Recall**	**F1**
Basic	88.4	76.0	81.7
Basic + Synsets	68.9	84.0	75.7
Basic + Web Expansion	79.3	92.0	85.2

Table 6.1.6 Results for Query Classification

As we can see from Table 6.1.6, the results obtained using a basic keywords input can achieve a rather high precision of 88.4. The main reason for this high precision is perhaps due to the fact that some manually selected keywords overlap with the category keywords. However this precision also comes with a cost of significantly lower recall. This first method returns the fewest assignments out of the 3 methods, only 43 assignments, among which 38 are correct. The second method which uses synsets is relatively the opposite, obtaining a high recall of 84.0 but a low precision of only 68.9. By adding an additional number of terms, we can effectively see an increase in recall for the second method. The core problem with the second approach however, is that synonyms added to the main keywords often add noise instead of assisting the match. The problem is worse in the case of phrases where there may be more words. For example "Graphics Card" returns "*artwork, art, graphics, nontextual matter, computer graphic, identity card, wag, wit, poster, posting*", out of which only "*computer graphic"* may be the most relevant. In order to effectively use synonyms, we have to further consider the context or senses of Synset terms chosen from WordNet. However, we cannot identify the context since we are usually given only a short query in most web searches. The third method works sufficiently well, attaining an overall F1-measure of 85.2. The main reason for this overall increase is that the web search results obtained from the top 100 Google snippets contains some context which the keywords are usually defined together with. For example, Diamonds will usually return *rings*, *polish*, *carat*, and Graphics cards will usually return *video card*, *computers*, or *memory*. These contextual terms, which are what most users are interested in, helps to boost the performance significantly. At the current stage, a query classification using original input query and co-occurring terms from web search results may be sufficient for an ontology query system. Further work could involve investigating a good query processing algorithm to combine available knowledge and perform a more accurate query mapping.

6.2 Web Page Classification

Before we are to pass the information to the ontology builder for model extraction and later ontology integration, we need to pre-classify web pages into general classes. The system in Ye and Chua (2004) is smart enough to learn object model automatically using a set of similar web pages. However if we do not group similar web pages together or pre-classify them, we may introduce noise and affect the performance of overall data model extraction. As such this section discusses the results obtained for web page classification according to the categories or domains which are also used in Section 6.1. The results in this section will serve as an indicator if automatic web page classification is feasible and indirectly allowing us to learn object models automatically using the system in Ye and Chua (2004). The experimental setup for this section is similar to Section 6.1, with the exception of the evaluation procedure. The features used in this classification is also identical to that in Section 6.1, a "bag-of-words" approach using vector space model with boosting in weights for different element types. The reasons for choosing such a model are in its simplicity and the availability of enough training data. The amount of data redundancy in the term overlap is expected to be sufficient for most classification purposes. The effectiveness of this simple approach is demonstrated in Becker and Kuropka (2003). In our future work, we will investigate more advanced techniques to be used in the classification process.

Classification Category	**Precision**	**Recall**	**F1**
Computer	68.4	70.9	69.6
Hard Disk	67.1	87.6	76.0
Digital Camera	78.8	84.3	81.5
MP3 Player	82.9	88.7	85.7
Video Cards	71.2	75.6	73.3
Diamonds	89.2	82.3	85.6

Table 6.2.1 Results for Web Page Classification

The objective of web page classification is to correctly assign a category to a given web page. In this experiment, we attempt to classify 300 web pages into 6 domains. Thus there are approximately 50 pages for each domain. The training data used are similar to that of Section 6.1, but with 200 web pages for each category. From the results in Table 6.2.1, it is clear that *Digital Camera, MP3 Player,* and *Diamonds* achieve a much higher F1-measure as compared to *Computer, Hard Disk*, and *Video Cards.* The foremost reason for the poor performance of the latter 3 categories is the contextual overlap of the data. More often than not *Computer* is mentioned along with *Hard Disk* components, *Video Cards* details and other hardware equipments such as RAM or DVD-Drive. Similarly, web pages for *Hard Disk* and *Video Cards* mention almost the same thing as *Computer* does. This contextual overlap causes ambiguities for the classifier and thus a lower F1-measure. The highest possible F1-measures are obtained by classification for MP3 Player and Diamonds. They achieved the F1-measures of 85.7 and 85.6 respectively. The main reason for such a high accuracy lies in the uniqueness of their web pages. Web pages for Diamonds for example, are usually structured and layout in a way much different than that of *Computer* and *Hard Disk.* Furthermore, the terms used to describe them are often very much different. In general, we can also notice that the recall are usually slightly higher than precision. One possible explanation for this is that the threshold used by the classifier, to determine a class, is set slightly towards favoring recall (ie. many instances are above the threshold). Manual tuning may tilt this scale again, but sometimes it is hard to determine a best trade-off for an actual application usage. At the current stage, it seems feasible to implement an automated web page classification system and hence making it possible for us to learn object models automatically using the system in Ye and Chua (2004). Future work in web page classification can involve advanced analysis of web page structure or web page summary for classification (Shen et al, 2004). It should also look towards an automatic classification without the need of training data, perhaps through some form of clustering.

6.3 Ontology Model Extraction and Integration

This section documents the findings for the main topic in this research project, Ontology Integration. The experiments are performed on a three-tier process: 1) Baseline System 2) Basic Diamond System 3) Enhanced Diamond Model System with Web Knowledge.

The Baseline System is based on summing up all the syntactic and semantic similarities between all the terms from INT and EXT as is done in Ye et al, (2006). CXT evidence is not measured here as we need to analyze if CXT plays an important role in the normal Diamond Model. Moreover, INT, EXT and CXT are considered separately only in the Diamond Model as in Ye et al (2006), and not considered separately in the Baseline System. The third system differentiates from the second by the usage of web knowledge (online guides, Wikipedia and Google search results). While the second system uses only WordNet and a simple INT, EXT, CXT measure through the ontology trees, the third system uses all of these and in addition provide additional context through the use of web knowledge or extra instances. In short, the second system is a basic one without the complex model of contextual information on the web as in the third system.

Corpus	System	Precision	Recall	F1
I3CON	Baseline	67.7	63.5	65.5
	Basic Diamond	86.1	89.3	87.7
	Enhanced Diamond + Web	89.2	91.0	90.1
TEL-8	Baseline	71.3	78.5	74.7
	Basic Diamond	92.1	91.6	91.8
	Enhanced Diamond + Web	93.6	93.7	93.6
Online	Baseline	55.4	61.2	58.2
	Basic Diamond	78.7	71.9	75.1
	Enhanced Diamond + Web	81.3	78.7	80.0

Table 6.3.1 Results for Ontology Integration

From the findings in Table 6.3.1, there is a major difference in all three results. The difference is especially significant between the Baseline and any of the Diamond Systems. There is about an average of 28% increase in all measures: precision, recall, or F1. This shows that the CXT or a measure of contextual information can be helpful. In addition, we notice that the errors in the Baseline System are usually caused by instances which have a large variation in structure, granularity and noise. The Baseline System is also especially prone to local spurious evidences, such as homonyms (words with same pronunciation or spelling but have different meanings). By considering each evidence (INT, EXT, CXT) separately, the Diamond Systems are more robust in coping with such instances. In many situations, the lack of context causes the Baseline system to perform very poorly. For example, the concept "name" may refer to the name of an airline company or name of a book, ie. the title. This ambiguity often can only be resolved when we consider the context, such as the surrounding concept nodes of "ISBN", "subject" and "publisher". The weights used for combining the matchers may also play a part, but we will not investigate that in this thesis. The weights are currently set based on empirical tests.

System	**Average F1**
Baseline	66.1
Basic Diamond	84.9
Enhanced Diamond + Web	87.9

Table 6.3.2 Average F1

Table 6.3.2 presents the average F1-measures given in Table 6.3.1. The average F1 for the Basic Diamond model far exceeds the Baseline system, and the Enhanced Diamond System with web knowledge performs even better. It achieves an increase in around 3% to 5% in F1-measures in each corpus as compared to the Basic Diamond System. This conforms to the previous finding that contextual information plays an important part in Ontology Integration. In particular, WordNet helps to connect the semantic relatedness between terms and web knowledge provides an extra source of contextual evidence. In cases where ambiguities cannot be clearly resolved using only the basic evidences, the Enhanced Diamond model effectively exploits data redundancy over the web to resolve them. For example, the concept "resolution" for computers and digital cameras has a major overlap. One refers to the "resolution of monitor screen" while the other refers to "resolution of image taken", or in some cases "resolution of embedded LCD screen". In most of these cases, the pure surface evidence found in the ontologies themselves or WordNet is not enough. The only way to resolve such ambiguities is to gather more evidence via other external sources. The World Wide Web which provides a huge source of readily available knowledge is therefore very useful. This is further validated through our findings with an achieved average F1 measure of 87.9 by utilizing web knowledge.

The results from our Enhanced Diamond Model with Web Knowledge significantly outperform the best F1-measure of 77.0 previously reported in I3CON (Hughes, 2004), and are comparable to that reported on the TEL-8 corpus (He et al, 2004). The amount of variability and the size of the ontologies are strictly proportional to the improvement in F1-measure for our new model over existing systems. There is not much improvement in TEL-8 corpus as it is a relatively simple and small corpus. It contains only two domains, namely, Books and Airfare, and a very fixed naming convention for most concepts. In many cases, a simple lexical matching of the concept names will suffice. As such, a high F1-measure of above 90.0 is easier to achieve. On the other hand, the I3CON data consists of 7 domains and more variation in naming conventions. Thus simple methods for ontology integration are not sufficient. Under such situations, our new model achieved an improvement of around 13% in F1-measure over existing systems. This shows that our new model is more robust with large ontologies or those with more variation. Results from online corpus also indicate that our new model can perform much better over simple methods as in the Baseline system. The effective use of web knowledge and better modeling of different evidences appear to be essential in robust, automatic, large scale ontology integration.

6.3.1 Roles of Different Types of Web Knowledge in Ontology Integration

In order to be comprehensive in our studies, we further divided our web knowledge into three types to be used together with the Enhanced Diamond Model. This follows from Section 5.4 where three types of web knowledge are used in the web-based similarity matchers. They are namely: 1) Guides. 2) Wikipedia. 3) Google Search Results. The main purpose of separating them is to identify which source of web knowledge is likely to contribute most in the performance of ontology integration. The secondary objective is to verify which source of web knowledge is consistent in improving the overall results and if they are too erroneous to be used.

Corpus	System	Precision	Recall	F1
I3CON	Basic Diamond	86.1	89.3	87.7
	Enhanced Diamond	88.0	88.2	88.1
	Enhanced Diamond + Google	88.2	89.5	88.9
	Enhanced Diamond + Wiki	89.1	90.3	89.7
	Enhanced Diamond + Guides	88.6	88.3	88.4
	Enhanced Diamond + Web Knowledge	89.2	91.0	90.1
TEL-8	Basic Diamond	92.1	91.6	91.8
	Enhanced Diamond	92.9	91.7	92.3
	Enhanced Diamond + Google	92.9	93.6	93.2
	Enhanced Diamond + Wiki	92.7	92.8	92.7
	Enhanced Diamond + Guides	91.6	92.7	92.1
	Enhanced Diamond + Web Knowledge	93.6	93.7	93.6
Online	Basic Diamond	78.7	71.9	75.1
	Enhanced Diamond	75.8	75.2	75.5
	Enhanced Diamond + Google	75.9	80.3	78.0
	Enhanced Diamond + Wiki	81.5	77.8	79.6
	Enhanced Diamond + Guides	78.7	77.9	78.3
	Enhanced Diamond + Web Knowledge	81.3	78.7	80.0

Table 6.3.3 Results using Different Types of Web Knowledge

Table 6.3.3 shows the results obtained from the Enhanced Diamond Model with the use of each individual type and also their full combination. These results are additionally compared against the Basic Diamond Model and Enhanced Diamond Model which uses only the enhanced context matching as described in Section 5.5. These additional results are identical to those in Table 6.3.1.

From the results, we can see that the Enhanced Diamond Model itself provides only a slight increase in F1-measure of about 0.5 when compared to the Basic Diamond Model. The main reason for the marginal improvement in F1 measure is the shallow depth of the ontology trees involved. For simplicity, most of the ontologies used in this study were restricted up to only a depth of five levels. This indirectly lessened the effect of enhanced contextual matching across different levels of an ontology tree. Though the current results do not show any strong significance for the proposed contextual matching method, we believe that given the task of integrating bigger or deeper ontologies, the improvements will be noteworthy.

From Table 6.3.3, we can also clearly see that the use of a combination of web knowledge helps to boost the results up by around 5%. We are able to achieve a F1-measure of 80.0 for the online corpus as compared to 75.1 in the Basic Diamond Model. This shows that web knowledge indeed can help in boosting performance. As for each individual type of web knowledge, online guides do not seem to work equally well across all corpus. It improves the performances on F1 measure for I3CON by 0.3 and online corpus by 2.8, but decreases the performance for TEL-8 corpus by 0.2. One of the main reasons for such a low improvement in I3CON is that the online guides are not tuned towards the domain required. The majority of the online guides relevant to I3CON are too general and do not discuss about any specific concepts in the I3CON domains. For example, guides about animals discuss how to choose a good pet, while two of the main concepts in the animal domain are “two legged things” and “biological mother”. The generality of the guides with respect to the domain does not help especially in the TEL-8 corpus. For example, in the airfare domain, the two concepts “arrival city” and “departure city” have a high similarity score based on the guides, but which in fact mean two different concepts. This results in more spurious matches, thereby causing a decrease in overall F1 measure. On the other hand, guides work reasonably well for the online corpus. This is basically due to the fact that majority of the chosen online guides were written to cater for online shopping data. The topics and section headings in the guides are often directly the concept itself, for example "Size of RAM" in a heading and the concept "Computer RAM". In general, we conjecture that guides work well only if they are specific enough and are available in sufficient numbers. The current 39 guides are only specific enough for online shopping data and there are not enough of them to cover concepts mentioned in some domains.

The second type of web knowledge, Wikipedia, seems to perform reasonably on the three corpuses. It improves performance in F1 measure by 1.6 for I3CON, 0.4 for TEL-8, and up to 4.1 for online corpus. This general improvement can be explained by the fact that if a concept can be found under Wikipedia, it often contains articles which are very specific and detailed, for example "Graphics Card". This allows us to capture more relevant evidences describing the concept. Another benefit in using Wikipedia is that new information may be added by users or domain experts to further help ontology integration in the future. The main drawback however, is that some articles may contain phony, inaccurate, or duplicated information as Wikipedia is open to public for editing. Furthermore, some concepts, such as "LCD", may have more than one articles retrieved and by considering all articles, we may be adding noise to the overall similarity score.

The third type of web knowledge, search engine results such as Google, improved the F1-measure for the model by 0.8 for I3CON, 0.9 for TEL-8, and 2.5 for online corpus. On average, search engine results do not perform as well as Wikipedia. The main cause for this is that many snippets from the search results may contain advertisements or sites selling some products, but not describing the concept at all. For example, a submitted query on "Diamond" returns "*Diamond womens car insurance … Visit now*". The major issues in using search engine results are: 1) how should we do filtering of noise, and 2) how can we submit a clever query. It is worth mentioning that query expansion and query reduction may help in the process, but we are not investigating into them as it is out of the scope of this thesis.

In summary, different enhancements to the basic Diamond Model bring about different aspects of improvements for ontology integration. The enhanced matching method for the Diamond Model enables us to perform better ontology integration if the ontology trees are deep or extensive. Guides can perform very well if they describe the specific domains we are working on. However, they need to be selected carefully and available in bigger numbers before they can become truly useful. A preprocessing step to select many good guidebooks for multiple domains should be investigated. Wikipedia provides consistent performance as a source of external knowledge for discovering hidden semantics. Nevertheless, problems occur when the terms cannot be found or many related articles are returned. The current assumption of taking only the top returned article may not be valid across all cases. Further work should look into differentiating between which articles are more relevant, and perhaps how to combine similarity scores across all articles returned. Web search results, such as Google, are sufficiently consistent for use as external knowledge or general purpose tasks in ontology integration. The main drawbacks are the frequent occurrences of spurious snippets with advertisements, and snippets which are too short. We conjecture that a filtering process to remove such snippets will increase the reliability of web search results as a good source of external knowledge. In general, a good ontology integration system requires good sources of external knowledge to provide contextual evidences or semantics for the concepts. This can be done manually by human experts, or in our case, automatically discovered using various types of web information. The results from our experiments show promising signs that the automated process may indeed be feasible. By leveraging data redundancy from web knowledge sources, our new automated model is able to outperform most existing systems. This may further be extended to include other knowledge sources such as textbooks or published papers.

7. Usage of Ontology for Information Retrieval

There are two main issues involved when retrieving information from merged or aligned ontologies. Firstly, the number of nodes or concepts in the combined ontologies may be huge. For example, a simple ontology about computer can contain up to 38 concepts, such as Processors, Operating Systems and Video Cards. It is not effective for users to view all the concepts and analyze the information by themselves. They need a way to select important information for viewing automatically. Secondly, the number of instances belonging to the ontologies is usually quite large. A search on Books may return thousands of instances which belong to the ontology when only one or two books are of utmost interest to the users.

Henceforth the preliminary steps towards conceptualizing ontology information, involves two steps. The first step is Latent User Preference Detection: a process whereby user preferences are detected through a combination of different information sources, without being informed by the user beforehand. This process enables us to automatically select only concepts which are of greater importance to the users. The second step is Ontology Instance Summarization: a process whereby instances of an ontology is ranked and only a distinct top N of them is returned to the users, while showing only the minimal set of information users are most interested in. This process effectively helps to filter out unnecessary information and present only details of interest to users. In this chapter, Section 7.1 and Section 7.2 discusses about Latent User Preference Detection, and Ontology Instance Summarization respectively. At the end of this chapter, experimental results obtained for each process are reported.

7.1 Latent User Preference Detection

The growth of information especially on the web has led to a common phenomenon whereby users spent majority of their time using search engines. Search engines often retrieve an enormous number of results where the users are left helpless in finding their desired products or information. This not only causes frustration for the users, it also results in a reduced turnover for many businesses. Often users cannot find their desired product even though they have the money to spend and the product is readily available. Adaptive web sites and personalized web applications has thus become one interesting research topic in recent years (Rossi et al, 2001). Since manual customization is very time consuming and tedious, users may not be willing to do them. Hence, we need an automatic way to detect user preferences whether hidden or declared. We shall call this process Latent User Preference Detection.

Current State-of-the-art techniques for user preference detection uses scores to describe preferences or distinguish between liked and disliked values as in Kohavi (2001), Delgado and Ishii (1999). An expressive and mathematically grounded framework was proposed in Kiebling (2002). It uses strict partial order for semantic preferences such as “I like X more than Y”, and special cases to describe numeric, negative or complex preferences. Joachims (2002) analyzed click-through information to boost the results of search engines. He recorded the search results that are selected by the users and used them as opinions of the users to find better rankings for them in future searches. The proposed method for measuring latent user preference follows from the above works by utilizing recall and centroid scores (Radev et al, 2000) found for each group of concept terms. A group of concept terms refers to the set of terms or keywords representing a given concept. The main assumption we have is that web knowledge contains an inherent user preference model. The top results from search engines for example, are in fact derived through some form of user preference selection such as the most frequent searches, click-through data and page rank (Hansen and Shriver, 2001). Wikipedia articles contain what the users would like to discuss or share, and guide books contain even more specific information about what users would like to know. Using this web corpus as a basis, we measure the importance of each concept in our ontologies by first computing the cosine scores for the centroid vectors based on the modified Salton's Vector Space Model in Eqn 6. To cater for the need of modeling click-through data information, we added a weight boost of 1.5 for text appearing in anchor links. The computational complexity of this process is linear based on the number of documents and the total number of terms for all concepts. The overall score for each concept as a model of user preference, under a given domain, is tabulated by a linear weighted sum of the normalized centroid score and normalized recall score. The recall score is currently given by the average number of documents retrieved by each concept when using Google and Wikipedia search. Eqn 7 shows the recall score function and Eqn 8 the score function for a concept as a model of user preference.

$$R(C) = \frac{\sum_{i}^{k} r_i}{k} \qquad \ldots(7)$$

where *C* is the concept, *k* is the number of terms for *C*, and r_i is the recall for term *i*

$$I(C_x) = \theta * \frac{V(C_x)}{Max\{V(C_1),...,V(C_n)\}} + (1-\theta) * \frac{R(C_x)}{Max\{R(C_1),..., R(C_n)\}} \qquad \ldots(8)$$

where $V(C_i)$ is the centroid score, $R(C_i)$ the recall score, and θ a weight, $0 \leqq \theta \leqq 1$

7.2 Ontology Instance Ranking & Summarization

In order to effectively summarize information for the users, we need to retrieve only instances which are of relevance to the users based on their preferences, and show only the more important concepts users would like to know. From Section 7.1, we are able to rank concepts which are important to users in each given domain. Using these results, we further rank the instances by combining the scores for each of the concepts appearing in the instances. For example, we first compute the score for "Hard Disk" concept in an instance *Y*, then we compute the score for "CPU" concept in *Y* again. These scores are then combined through RankBoost (Freund et al, 1998) to achieve a final ranking for instance *Y* to appear in the results. Each of the concept score is normalized and computed based on a simple set of heuristic rules which include: 1) "the greater the number the better" for any numerical data, 2) "the smaller the number the better" for price, and 3) "the higher the rank of a brand the better". The rank of brands is obtained from Business Week's Top 100 Global Brands. In our work, we use *N* rank lists corresponding to the *N* most preferred concepts found in Section 7.1, and another rank list which measures the scores for all other concepts combined. The reason for combining the rest of the concepts into the last rank list is because after a certain number of concepts, for example *N*=5, the rest of the concepts contribute significantly lesser importance during comparisons (from a user point of view). For example the top 5 concepts may have a preference score of around 0.9 while the next 10 concepts only have a preference score of only 0.05. In addition, since the heuristic scoring methods are tuned towards the most preferred concepts, they may not work well across all concepts, especially the lesser preferred concepts. The rank for the last rank list is computed based on a weighted preference boost to the summation of scores across the lesser preferred concepts (See Eqn 9 and 10).

$$WB_i = \frac{\mathrm{Pref_i}}{\mathrm{Max}(\{\mathrm{Pref_1} \ldots \mathrm{Pref_n}\})} \qquad \text{... (9)}$$

$$Rank\_Score(Y) = \sum_{j=1}^{n} (WB_j * Q_j^Y) \qquad \ldots \text{(10)}$$

where WB_i is the weight boost for Concept *i*, $Pref_i$ is the preference score for Concept *i*, *Y* is the instance to be scored or ranked, and Q_j^Y is the normalized score for Concept *j* given instance *Y*.

After obtaining all the rank lists for each instance, we combine the rank lists using RankBoost. The choice of RankBoost to combine the scores is because different scores for different concepts may have very different meanings. Moreover, RankBoost has been successfully employed in information retrieval, natural language processing and shape localization. It is also shown to be an effective algorithm for combining rank lists or preferences in Freund et al (1998). Figure 7.1 shows the algorithm for RankBoost.

The training data for RankBoost is a manually labeled set of 30 pair-wise ordered instances randomly selected among all instances. For example, <A,B> indicates that instance A is preferred or has higher rank than B, <B,C> indicates that B is of higher rank than C, etc. Transitivity rule holds for the ranking order, hence <A,C> is implicit. In addition to the training data, the main parameter required for RankBoost is the number of boosting iterations to undergo. Intuitively, the larger this number is, the better it fits the training data. However, there may be a risk of over-fitting if the number is too large. We set this number to 30 based on the empirical results obtained in Zhao et al (2006).

```
Let $X$ be the full instance space and $x \in X$ be the instances we are interested in.
Given the initial distribution $D_0$, initialize $D_1 = D_0$
For $t = 1$ up to T, perform:
    - Train weak learner using distribution $D_t$
    - Get weak ranking $h_t : X \rightarrow R$
    - Choose $\alpha_t \in R$
    - Update
        $D_{t+1}(x_0, x_1) = \frac{D_t(x_0, x_1)\exp(\alpha_t(h_t(x_0) - h_t(x_1)))}{Z_t}$
      where $Z_t$ is a normalization factor, chosen so that $D_{t+1}$ is a distribution
Output the final ranking:
        $H(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$
```

Figure 7.1 RankBoost Algorithm

As shown in Figure 7.1, RankBoost builds a strong ranking function *H* from *T* numbers of weak ranking functions h_t. The weak ranking function h_t is derived from a rank list f_n by comparing the score of f_n on a given instance to a threshold σ. For instances unranked by f_n, the weak ranking function assigns a default score *s* which is very low. With respect to our task, the rank lists f_n corresponds to the rank lists obtained for each concept in the given domain. The weak ranking functions h_t, are subsequently acquired using these rank lists f_n. At the final stage, the ranking function H combines all weak ranking functions h_t to form a final rank list.

Among the final rank list of instances to be returned and shown to the users, we further attempt to summarize the information in the instances by returning only ranked concepts above a predefined threshold, π (as computed in Section 7.1). This threshold is currently set at 0.65 based on empirical tests. A higher threshold value would mean lesser concepts returned and thus a more summarized view for the users, while a lower value would mean more concepts returned. A value of zero will simply be a non-summarized version where the users have to scan through all the concepts by themselves.

7.3 Subjective Evaluation

In order to evaluate our proposed idea for steps towards ontology conceptualizing, we performed subjective evaluation based on the results obtained from Section 7.1 and Section 7.2. We invited a total of 50 volunteers to help rate the performance of our system. Each volunteer is asked to rate on a question based on a scale of 1 to 5, with 1 denoting “Very Unreasonable” and 5 denoting “Very Reasonable”. There are two main experiments performed: 1) user preference on top five concepts selected by the system, and 2) user preference on system summarized ranked results compared to that of MSN Shopping, PriceGrabber.com, Froggle and Yahoo Shopping. We have restricted the first experiment to 3 domains, namely, “Book”, “Computer” and “Digital Camera”;

and the second experiment to 2 domains, namely, "Computer" and "Digital Camera", due to time constraints. There are two main reasons for choosing and using these domains for the experiments. Firstly, the domains encompass a spectrum of products which users are interested to buy on online shopping sites. Secondly, the domains are varied enough and have quite a large number of instances. Through the two experiments, we hope to obtain results to verify the feasibility and perceived usefulness for the combined usage of ontology and web knowledge for information retrieval. The evaluations in our experiments are done based on Likert-type item analysis as discussed in Clason and Dormody (1994).

Concepts \ Scale Ratings		1	2	3	4	5	Mean	Std. Dev
Book	Author	0	2	18	16	14	3.84	0.126
	Title	0	5	19	16	10	3.62	0.131
	Subject	0	1	16	21	12	3.88	0.113
	Description	0	7	14	18	11	3.66	0.139
	Price	0	4	18	18	10	3.68	0.126
Computer	CPU	0	6	12	19	13	3.78	0.138
	Display	0	5	17	16	12	3.7	0.135
	Price	0	3	13	16	18	3.98	0.132
	Graphic Card	0	4	9	18	19	4.04	0.134
	Hard Drive	0	4	12	17	17	3.94	0.135
Digital Camera	Price	0	9	15	16	10	3.54	0.143
	Resolution	0	5	16	17	12	3.72	0.134
	Zoom	0	3	14	22	11	3.82	0.120
	Battery Life	0	5	15	18	12	3.74	0.133
	LCD	0	3	15	19	13	3.84	0.126
Average							3.79	0.131

Table 7.3.1 User Preference on Selected Top 5 Concepts

Table 7.3.1 shows the user preference distribution for the top five concepts returned for each domain. The aim of this experiment is to verify if our system can automatically detect user preferences based on web knowledge. As shown in the results, most users give a rating of at least 3 and above. Only 10% of the users rated 2 and none gave the rating of "Very Unreasonable".

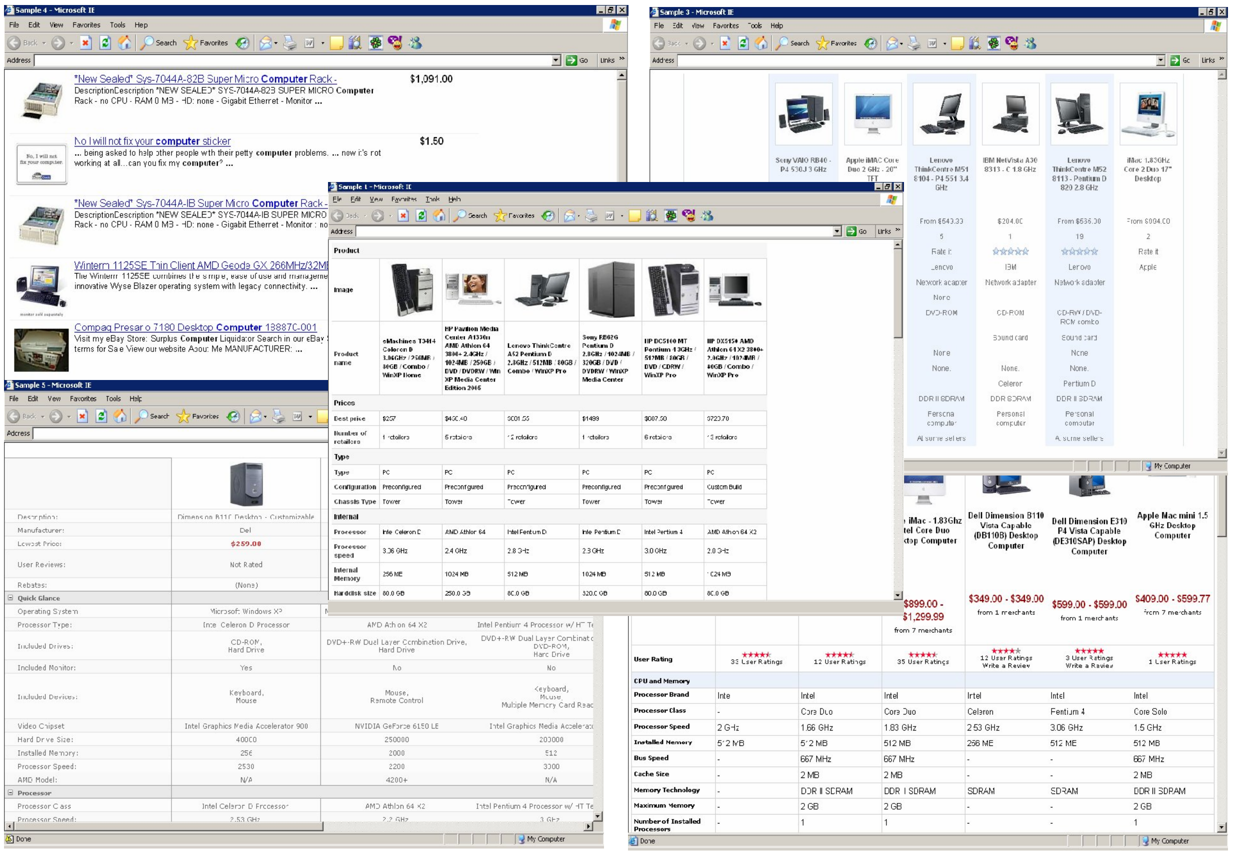
The average mean across all results is 3.79, indicating that most of the users feel that the returned results are much above average. The average standard deviation is 0.131, indicating that there is not much fluctuations with this agreement. A pair-wise percentage agreement was also computed using the results and the average percentage agreement was found to be 71.4%, indicating that most users agree with one another during their judgments. We should note that for simplicity, we assumed that all raters are honest and unbiased in their ratings. However, since raters may know the source of this experiment, there may be bias towards the ratings. The raters may be giving a rating according to their preference or dislike for the author or the author's affiliations.

Data Set \ Scale Ratings		**1**	**2**	**3**	**4**	**5**	**Mean**	**Std. Dev**
Computer	Set 1	0	0	19	21	10	3.82	0.748
	Set 2	0	0	24	17	9	3.7	0.763
	Set 3	0	0	23	17	10	3.74	0.777
Digital Camera	Set 1	0	0	18	20	12	3.88	0.773
	Set 2	0	0	20	18	12	3.84	0.792
	Set 3	0	0	16	22	12	3.92	0.752
Average							3.82	0.767

Table 7.3.2 User Preference on Returned Results

The second experiment involves returning summarized ranked results of ontology instances to the users based on their detected user preference in the previous experiment. The results are returned together with the results from MSN shopping, PriceGrabber.com, Froogle and Yahoo shopping. Only the top 6 results from each source are returned for each set of the data. Set 1 involves "Finding the cheapest", Set 2 involves "Finding the best configuration" and Set 3 involves "Finding the overall best". These results are filtered for advertisements, javascripts and any possible indication of their source. They are then formatted using random templates and passed to the 50 users for rating. In this experiment, users are asked to assign a rank scale of 1 to 5 for each of the 5 sources: Our System, MSN, PriceGrabber.com, Froogle and Yahoo. They are allowed to assign a rank only once for each data set. Hence if they give our system a rating of 5, no other sources can have a rating of 5. Figure 7.2 shows the screenshots of the returned results from different sources for Computer Data Set 2. Table 7.3.2 shows the distribution obtained for the returned ranked results by our system.

From the results, we can clearly see that most users think that our answers are reasonable or at least comparable with other sources. We achieved a mean rating of 3.82, indicating that most users like our system. However, the standard deviation is rather high at 0.767. This indicates that users are not very consistent in giving us such a high rating. The average mean rating fluctuates between 3.053 and 4.587, indicating that users believe that our answers are at least reasonable, if not better. By computing the pair-wise percentage agreement and averaging them, we obtained a measure of 73.8%. This shows that most users agree with one another during the judgments.



New Sealed Sys-7044A-82B Super Micro Computer Rack - $1,091.00
DescriptionDescription *NEW SEALED* SYS-7044A-823 SUPER MICRO Computer Rack - no CPU - RAM 0 MB - HD: none - Gigabit Ethernet - Monitor ...

No I will not fix your computer sticker $1.50
... being asked to help other people with their petty computer problems. ... now it's not working at all...can you fix my computer? ...

New Sealed Sys-7044A-IB Super Micro Computer Rack -
DescriptionDescription *NEW SEALED* SYS-7044A-IB SUPER MICRO
Rack - no CPU - RAM 0 MB - HD: none - Gigabit Ethernet - Monitor : no

Winterm 1125SE Thin Client AMD Geode GX 266MHz/32M
The Winterm 1125SE combines the simple, ease of use and manageme
innovative Wyse Blazer operating system with legacy connectivity. ...

Compaq Presario 7180 Desktop Computer 18887C-001
Visit my eBay Store: Surplus Computer Liquidator Search in our eBay
terms for Sale View our website About Me MANUFACTURER: ...

Product						
Image						
Product name	eMachines T3414 Celeron D 3.06GHz / 256MB / 80GB / Combo / WinXP Home	HP Pavilion Media Center A1330n AMD Athlon 64 3800+ 2.4GHz / 1024MB / 250GB / DVD / DVDRW / Win XP Media Center Edition 2005	Lenovo ThinkCentre A52 Pentium D 2.8GHz / 512MB / 80GB / Combo / WinXP Pro	Sony RB62G Pentium D 2.8GHz / 1024MB / 320GB / DVD / DVDRW / WinXP Media Center	HP DC5100 MT Pentium 4 3GHz / 512MB / 80GB / DVD / CDRW / WinXP Pro	HP DX5150 AMD Athlon 64 X2 3800+ 2.0GHz / 1024MB / 80GB / Combo / WinXP Pro
Prices						
Best price	$257	$450.40	$601.55	$1499	$607.50	$723.70
Number of retailers	1 retailers	5 retailers	12 retailers	1 retailers	6 retailers	13 retailers
Type						
Type	PC	PC	PC	PC	PC	PC
Configuration	Preconfigured	Preconfigured	Preconfigured	Preconfigured	Preconfigured	Custom Build
Chassis Type	Tower	Tower	Tower	Tower	Tower	Tower
Internal						
Processor	Intel Celeron D	AMD Athlon 64	Intel Pentium D	Intel Pentium D	Intel Pentium 4	AMD Athlon 64 X2
Processor speed	3.06 GHz	2.4 GHz	2.8 GHz	2.3 GHz	3.0 GHz	2.0 GHz
Internal Memory	256 MB	1024 MB	512 MB	1024 MB	512 MB	1024 MB
Harddisk size	80.0 GB	250.0 GB	80.0 GB	320.0 GB	80.0 GB	80.0 GB

Sony VAIO RB40 - P4 530J 3 GHz	Apple iMAC Core Duo 2 GHz - 20" TFT	Lenovo ThinkCentre M51 8104 - P4 551 3.4 GHz	IBM NetVista A30 8313 - C 1.8 GHz	Lenovo ThinkCentre M52 8113 - Pentium D 820 2.8 GHz	iMac 1.83GHz Core 2 Duo 17" Desktop
		From $543.33	$204.00	From $536.30	From $894.00
		5	1	19	2
		Rate it	☆☆☆☆☆	☆☆☆☆☆	Rate it
		Lenovo	IBM	Lenovo	Apple
		Network adapter	Network adapter	Network adapter	
		None			
		DVD-ROM	CD-ROM	CD-RW / DVD-ROM combo	
			Sound card	Sound card	
		None		None	
		None.	None.	None.	
			Celeron	Pentium D	
		DDR II SDRAM	DDR SDRAM	DDR II SDRAM	
		Personal computer	Personal computer	Personal computer	
		All some sellers		All some sellers	

Description:	Dimension B110 Desktop - Customizable				
Manufacturer:	Dell				
Lowest Price:	$259.00				
User Reviews:	Not Rated				
Rebates:	(None)				
Quick Glance					
Operating System	Microsoft Windows XP				
Processor Type:	Intel Celeron D Processor	AMD Athlon 64 X2	Intel Pentium 4 Processor w/ HT Te		
Included Drives:	CD-ROM, Hard Drive	DVD+-RW Dual Layer Combination Drive, Hard Drive	DVD+-RW Dual Layer Combinati DVD-ROM, Hard Drive		
Included Monitor:	Yes	No	No		
Included Devices:	Keyboard, Mouse	Mouse, Remote Control	Keyboard, Mouse, Multiple Memory Card Read		
Video Chipset	Intel Graphics Media Accelerator 900	NVIDIA GeForce 6150 LE	Intel Graphics Media Accelera		
Hard Drive Size:	40000	250000	200000		
Installed Memory:	256	2000	512		
Processor Speed:	2530	2200	3000		
AMD Model:	N/A	4200+	N/A		
Processor					
Processor Class	Intel Celeron D Processor	AMD Athlon 64 X2	Intel Pentium 4 Processor w/ HT Te		
Processor Speed:	2.53 GHz	2.2 GHz	3 GHz		

			iMac - 1.83Ghz Intel Core Duo Desktop Computer	Dell Dimension B110 Vista Capable (DB110B) Desktop Computer	Dell Dimension E310 P4 Vista Capable (DE310SAP) Desktop Computer	Apple Mac mini 1.5 GHz Desktop Computer
			$899.00 - $1,299.99 from 7 merchants	$349.00 - $349.00 from 1 merchants	$599.00 - $599.00 from 1 merchants	$409.00 - $599.77 from 7 merchants
User Rating	33 User Ratings	12 User Ratings	35 User Ratings	12 User Ratings Write a Review	3 User Ratings Write a Review	1 User Ratings
CPU and Memory						
Processor Brand	Intel	Intel	Intel	Intel	Intel	Intel
Processor Class	-	Core Duo	Core Duo	Celeron	Pentium 4	Core Solo
Processor Speed	2 GHz	1.66 GHz	1.83 GHz	2.53 GHz	3.06 GHz	1.5 GHz
Installed Memory	512 MB	512 MB	512 MB	256 MB	512 MB	512 MB
Bus Speed	-	667 MHz	667 MHz	-	-	667 MHz
Cache Size	-	2 MB	2 MB	-	-	2 MB
Memory Technology	-	DDR II SDRAM	DDR II SDRAM	SDRAM	SDRAM	DDR II SDRAM
Maximum Memory	-	2 GB	2 GB	-	-	2 GB
Number of Installed Processors	-	1	1	-	-	1

Figure 7.2 Screenshots of Returned Results

Systems	Overall Average Mean Rating
Our System	3.82
Froogle	1.46
MSN	3.26
Yahoo	3.28
PriceGrabber	3.16

Table 7.3.3 Average Mean Rating

In order to compare our results with other sources, we tabulated the overall average mean rating for each system. Table 7.3.3 shows the results obtained. From the results, we can see that most of the lowest ratings in this experiment were assigned to Froogle, the Google-based product search engine. One of the major reasons for this is that current Froogle results by itself do not have any feature comparison among the products. It only lists down the products in a descending order of relevance. In addition, Froogle often return results which are duplicates of one another. This causes more work for the users to look for a product or item they desire. Hence we have a general low rating for Froogle. On the other hand, MSN, Yahoo, and PriceGrabber have an average mean rating of around 3.16 to 3.28, indicating that most users feel quite neutral towards their results. In comparison, our system has a mean rating of 3.82, higher by 0.54 or about half a scale. This indicates that most users have a tendency to prefer our systems over others. Some possible explanations for the observed results are: 1) users indeed like our system much better than others since we provide a summarized view catering to their interests, 2) users are bias and somehow detected the source of each data set, and 3) the instances retrieved for each system are published on different dates and this may have affected the overall results.

In general, the experiments show promising signs that our framework is feasible. We should note however, that these experiments are very subjective and may not be representative enough of a much larger population. Future research should be directed towards a larger scale evaluation.

8. Conclusion

The World Wide Web (WWW) has evolved to be a major source of information. The great diversity and quantity of information is growing each day. This has brought about an overwhelming feeling of having too much information or being unable to find or interpret data. In addition, since online information in HTML format is designed primarily for browsing, it not amendable to machine processing such as database style manipulation and querying.

Thus to obtain valuable information on the web, the data must first be organized and retrievable. This can be done by performing some form of web structuring, such as storing data into a relational database or building an ontology. By building good ontologies from the web, data can also be easily shared and reused across applications and different communities. The task of building ontologies and making effective use of them is a valuable research topic to be studied upon.

This thesis reviewed some of the existing state-of-the-art systems and techniques used in recent researches for ontology building and integration. The main drawback of most systems is the requirement of a substantial amount of human invention. In addition, there is no effective use in the semantic information which can be found readily on the web and other resources. Most works also focus on either matching the structure or basic description of concepts during ontology integration.

In our work, we propose the consideration of different pieces of evidences, namely INT, EXT, CXT, and utilize available web knowledge to boost the results. Complementing the model is a proposed way to measure semantic context more effectively. Experiments on I3CON, TEL-8 and online shopping data have shown a promising sign that such techniques are feasible.

Additionally, we extended this work using web knowledge to perform latent user preferences detection, ontology instances ranking and summarization. Our work effectively demonstrates how web knowledge can be utilized to support ontology building, integration and question answering. The basis of this thesis can easily be extended to real life commercial applications which require automated building of knowledge bases or expert recommender systems.

8.1 Future Work

Future work for ontology integration should look into better modeling of contextual and semantic information from other sources of information, such as published papers or textbooks. The parameters and thresholds used in our ontology integration processes should also be investigated. Methods such as Expectation Maximization or Simulated Annealing may be used to set these parameters automatically. The work can further be extended to incorporate logic, constraints and axioms integration within the ontologies. As for ontology conceptualization, we have performed only a small step towards automated summarization of the ontology instances. More work can be done on investigating a complete conceptualization of ontologies from different angles, be it from ontology instances point of view, user preferences, ontology graphs, or ontology axioms discovery. A more in-depth analysis of the ontologies and their instances through some form of clustering or grouping should be examined as well.

Bibliography

[1] Y. Arens, C.N. Hsu, and C.A. Knoblock. Query processing in the sims information mediator. In Advanced Planning Technology. AAAI Press, USA, 1996.

[2] M. Baziz, M. Boughanem and N. Aussenac-Gilles, The Use of Ontology for Semantic Representation of Documents, Workshop Semantic Web in SIGIR, 43-51, 2004.

[3] J. Barwise and J. Seligman. Information flow: the logic of distributed systems. Cambridge University Press, 1997.

[4] C. Batini, M. Lenzerini and S.B. Navathe, A comparative analysis of methodologies for schema integration, Computing Surveys 18(4), 323-364, 1986.

[5] J. Becker, D. Kuropka, Topic-based vector space model. In Proceedings of the 6th International Conference on Business Information Systems, Colorado Springs, 7-12, 2003.

[6] M. Bergman. The Deep Web: Surfacing the hidden value. BrightPlanet.com, 2000.

[7] M. Bordie, The promise of distributed computing and the challenges of legacy information systems. In Proceedings of IFIP, Australia, 1992.

[8] P. Borst and H. Akkermans, An ontology approach to product disassembly. In EKAW, 1997.

[9] M.W. Bright, A.R. Hurson and S. Pakzad, Automated resolution of semantic heterogeneity in multidatabases, ACM Transactions on Database Systems 19(2), 212-253, 1994.

[10] R. Brooks. Exact Probabilistic Inference for Inexact Graph Matching, 2003. http://www.cim.mcgill.ca/~rbrook/graphs/graph.pdf

[11] Business Week Online. http://bwnt.businessweek.com/brand/2006/

[12] A.E. Campbell and S.C. Shapiro, Ontologic mediation: an overview. In: IJCAI95 Workshop on Basic Ontological Issues in Knowledge Sharing, 1995.

[13] D. L. Clason, and T. J. Dormody. Analyzing data measured by individual Likert-type items. Journal of Agricultural Education, 35 (4), 31-35, 1994.

[14] E. Compatangelo and H. Meisel. Intelligent support to knowledge sharing through the articulation of class schemas. In Proceedings of the 6th ICKES, 2002.

[15] O. Corcho, A. Gomez-Perez. A Roadmap to Ontology Specification Languages. In 12th International Conference on Knowledge Engineering and Knowledge Management (EKAW'00), 80-96. October 2000.

[16] J. Delgado and N. Ishii. Online Learning of User Preferences in Recommender Systems. In Proceedings of the IJCAI-99 Workshop on Machine Learning for Information Filtering, 1999.

[17] A.H. Doan. Learning to map between structured representations of data. PhD thesis, University of Washington, Seattle, 2002.

[18] A.H. Doan, P. Domingos, and A. Halevy. Reconciling schemas of disparate data sources: A machine-learning approach. In Proceeding of SIGMOD, 2001.

[19] I. Dunham, N. Shimizu, B.A. Roe, S. Chissoe, A.R. Hunt, J.E. Collins, R. Bruskiewich, D.M. Beare, M. Clamp, L.J. Smink. The DNA sequence of human chromosome 22. Nature 402: 489-495, 1999.

[20] P.W. Foltz. Latent Semantic Analysis for text-based research. Behavior Research Methods, Instruments and Computers. 28(2), 197-202, 1996.

[21] J. Fowler, M. Nodine, B. Perry and B. Bargmeyer, Agent based semantic interoperability in Infosleuth, Sigmod Record, 1999.

[22] Y. Freund, R. Iyer, R. E. Schapire, and Y. Singer. An efficient boosting algorithm for combining preferences. In Proceedings 15th International Conference on Machine Learning, 1998.

[23] B. Ganter and R. Wille. Formal concept analysis: mathematical foundations. Springer Verlag, Berlin (DE), 1999.

[24] F. Giunchiglia, P. Shvaiko, and M. Yatskevich. S-Match: an algorithm and an implementation of semantic matching. In Proceedings of ESWS 2004, 61-75, 2004.

[25] C. H. Goh. Representing and Reasoning about Semantic Conflicts in Heterogeneous Information Sources. Phd, MIT, 1997.

[26] K. Golub, and A. Ardo. Importance of HTML structural elements and metadata in automated subject classification. In Proceedings of ECDL, 368-378, September 2005.

[27] T. Gruber. A translation approach to portable ontology specifications. Knowledge Acquisition, 5(2):199-220, 1993.

[28] T. Gruber. Toward principles for the design of ontologies used for knowledge sharing, International Journal of Human-Computer Studies, 43(5):907-928, 1995.

[29] R.V. Guha, Contexts: Formalization and Some Applications. Ph.D. Thesis, Stanford University, 1991.

[30] M. Halkidi, B. Nugyen, I Varlamis, and M Vazirgiannis. THESUS: organizing web document collections based on link semantics, VLDB, 320-332, 2003.

[31] M. H. Hansen and E. Shriver. Using navigation data to improve IR functions in the context of web search. In Proceedings of the 10th international conference on information and knowledge management, 2001.

[32] B. He and K.C.C. Chang, J. Han, Discovering Complex Matchings across Web Query Interfaces: A Correlation Mining Approach, KDD, 2004.

[33] D. Holmes, M.C. McCabe. Improving Precision and Recall for Soundex Retrieval. In Proceedings of IEEE ITCC, 22-27, 2002.

[34] E. Hovy, Combining and standardizing large-scale: practical ontologies for machine learning and other uses. In Proceedings of 1st ICLRE, 1998.

[35] T. Hughes, Results for I3CON Ontology Alignment Experiment, available at http://www.atl.external.lmco.com/projects/ontology/i3con.html, 2004.

[36] T. Joachims. Optimizing Search Engines using Clickthrough Data. In Proceedings of the 8th International Conference on Knowledge Discovery and Data Mining (SIGKDD 2002), 2002.

[37] Y. Kalfoglou and M. Schorlemmer. If-map: an ontology mapping method based on information flow theory. Journal of data semantics, 1: 98-127, 2003.

[38] V. Kashyap and A. Sheth, Semantic and schematic similarities between database objects: A context-based approach, International Journal on Very Large Databases, 5(4), 276-304, 1996.

[39] W. Kiebling. Foundations of Preferences in Database Systems. In Proceedings of the 28th International Conference on Very Large Databases (VLDB 2002), 2002.

[40] R. Kohavi. Mining E-Commerce Data: The Good, the Bad, and the Ugly. In Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2001.

[41] M. Lacher and G. Groh. Facilitating the exchange of explicit knowledge through ontology mappings. In Proceedings of the 14th International FLAIRS conference, 2001.

[42] F. Lehmann and A.G. Gohn, The EGG/YOLK reliability hierarchy: semantic data integration using sorts with prototypes. In the 3rd International ACM Conference on Information and Knowledge Management, 1994.

[43] M. Lesk, Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from an ice cream cone. SIGDOC, 24-26, 1986.

[44] W.S. Li, Knowledge gathering and matching in heterogeneous databases. In AAAI Spring Symposium, 2000.

[45] W.S. Li and C. Clifton. Semint: a tool for identifying attribute correspondences in heterogeneous databases using neural networks. Data Knowledge Engineering, 33(1):49-84, 2000.

[46] R.M. MacGregor. Inside the LOOM description classifier. In ACM SIGART Bulletin, 2(3):88-92, 1991.

[47] J. Madhavan, P. Bernstein, and E. Rahm. Generic schema matching using Cupid. In Proceedings of 27th VLDB, 48-58, 2001.

[48] J. McCarthy, Notes on formalizing context. In Proceedings of the 13th IJCAI, 1993.

[49] D.L. McGuinness, R. Fikes, J. Rice and S. Wilder, An environment for merging and testing large ontologies. In Proceedings of 7th ICPKRR/KR, 2000.

[50] E. Mena, A. Illarramendi, V. Kashyap and A.P. Sheth, OBSERVER: an approach for query processing in global information systems based on interoperation across preexisting ontologies. In Proceedings of the First IFCIS-CoopIS, 1996.

[51] G.A Miller, R. Bechwith, C. Fellbaum, D. Gross, and K. Miller 1990. Introduction to WordNet: An On-line Lexical Database. International Journal of Lexicography, 1990 (Revised August 1993), 235-312.

[52] P. Mitra, G. Wiederhold and M. Kersten, A graph oriented model for articulation of ontology interdependencies. In Proceedings of Conference on EDBT, 2000.

[53] N.F.Noy. Semantic Integration: A Survey of Ontology-Based Approaches. SIGMOD Record, Special Issue on Semantic Integration, 33 (4), December, 2004

[54] N.F. Noy and M. Musen. SMART: Automated Support for Ontology Merging and Alignment. In Proceedings of the 12th Workshop on KAW, 1999.

[55] N.F. Noy and M. Musen. PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment. In Proceedings of the 17th AAAI, USA, 2000.

[56] N.F. Noy and M. Musen. Anchor-PROMPT: Using non-local context for semantic matching. In Proceedings of IJCAI 2001 workshop on ontology and information sharing, Seattle, 63-70, 2001.

[57] N.F. Noy and M. Musen. PROMPTDIFF: A Fixed-Point Algorithm for Comparing Ontology Versions. In Proceedings of the 18th AAAI, 2002.

[58] B. Omelayenko. Integrating vocabularies: Discovering and representing vocabulary maps. In Proceedings of the First ISWC2002, 2002.

[59] B. Omelayenko and D. Fensel. A two-layered integration approach for product information in B2B e-commerce. In Proceedings of the Second EC WEB, 2001.

[60] L. Palopoli, L. Pontieri, G. Terracina and D. Ursino. Intensional and extensional integration and abstraction of heterogeneous databases, Data and Knowledge Engineering, 201-237, 2000.

[61] M.F. Porter. An algorithm for suffix stripping, Program 14(3), 130-137, 1980.

[62] S. Prasad, Y. Peng, and T. Finin. Using explicit information to map between two ontologies. In Proceedings of the AAMAS 2002 Workshop on OAS, 2002.

[63] D. R. Radev, H. Jing, and M. Budzikowska. Centroid-based summarization of multiple documents. In ANLP/NAACL Workshop on Summarization, 2000.

[64] E. Rahm, and P. A. Bernstein. A survey of approaches to automatic schema matching, J. of VLDB, 10:334-350, 2001.

[65] G. Rossi, D. Schwabe and R. Guimaraes. Designing Personalized Web Applications. In Proceedings of the 10th World Wide Web Conference, 2001.

[66] D. Shen, Z. Chen, Q. Yang, H.J. Zeng, B. Zhang, Y. Lu, W.Y. Ma. Web-page Classification through Summarization, In Proceedings of SIGIR, 27th ACM ICRD in Information Retrieval, 2004.

[67] C. Sherman. Search Engine Results Continuing to Diverge, 2005. http://searchenginewatch.com/searchday/article.php/3524411

[68] P. Shvaiko. Iterative schema-based semantic matching. Technical Report, University of Trento, 2004.

[69] U. Srinivasan, A.H.H. Ngu and T. Gedeon, Managing heterogeneous information systems through discovery and retrieval of generic concepts, Journal of the ASIS, 51(8), 707-723, 2000.

[70] H. Stuckenschmidt, H. Wache, T. Vogele, and U. Visser. Enabling technologies for interoperability. Workshop on the 14th International Symposium of Computer Science for Environmental Protection, 35-46, 2000.

[71] G. Stumme and A. Mädche. FCA-merge: bottom-up merging of ontologies. In Proceedings of 17th IJCAI, 225-230, 2001.

[72] Suggested Upper Merged Ontology (SUMO). http://www.ontologyportal.org/

[73] K. Sycara, M. Paolucci, M. V. Velsen, and J. Giampapa, Autonomous Agents and Multi-Agent Systems, Springer, 2003.

[74] V.A.M. Tamma and T.J.M. Bench-Capon, Supporting different inheritance mechanisms in ontology representations. In Proceedings of the 1st Workshop on Ontology Learning and ECAI, 2000.

[75] TEL-8 dataset. http://metaquerier.cs.uiuc.edu/repository

[76] M. Uschold, Creating, integrating and maintaining local and global ontologies. In Proceedings of the 1st Workshop on Ontology Learning, 2000.

[77] P.R.S. Visser and V.A.M. Tamma, An experience with ontology clustering for information integration. In Proceedings of the IJCAI-99 Workshop, 1999.

[78] H. Wache, T. Scholz, H. Stieghahn, and B. Konig-Ries. An integration method for the specification of rule oriented mediators. In Proceedings of the International Symposium on Database Applications in Non-Traditional Environments, Japan, 109-112, 1999.

[79] H. Wache, T. Vogele, U. Visser, H. Stuckenschmidt, G. Schuster, H. Neumann and S. Hubner. Ontology-Based Integration of Information - A Survey of Existing Approaches. In IJCAI-2001 Workshop on Ontologies and Information Sharing, 108-117, Seattle, April 2001.

[80] P.C. Weinstein and P. Birmingham, Comparing concepts in differentiated ontologies. In Proceedings of the 12th Workshop on Knowledge Acquisition, Modeling and Management, Canada, 1999.

[81] G. Wiederhold, An algebra for ontology composition. In Proceedings of 1994 Monterey Workshop on formal Methods, USA, 56-61, 1994.

[82] Wikipedia, the free encyclopaedia. http://www.wikipedia.org/

[83] A.B. Williams and C. Tsatsoulis, An instance-based approach for identifying candidate ontology relations within a multi-agent system. In Proceedings of the 1st Workshop on Ontology Learning, ECAI, 2000.

[84] World Wide Web Consortium. http://www.w3.org/

[85] W. Wu, C. Yu, A. Doan, and W. Meng. An Interactive Clustering-based Approach to Integrating Source Query interfaces on the Deep Web. In SIGMOD, 2004.

[86] S. Ye, T.S. Chua, Learning Object Model from Product Web Pages, workshop on Semantic Web, SIGIR, 69-80, 2004.

[87] S. Ye, T.S. Chua, B.W.J. Ang, Adaptive Integration of Ontologies on the Web, Internal Report, 2006.

[88] S. Ye, T.S. Chua, J. R. Kei, Clustering Web Pages about Persons and Organizations, Journal of Web Intelligence and Agent Systems, Vol(3), 1-14, 2005.

[89] M. Zhao, S.Y. Neo, H.K. Goh, T.S. Chua, Multi-Faceted Contextual Model for Person Identification in News Video, In Multimedia Modeling (MMM), China, 2006.