Web Semantic Annotation Using Data-Extraction Ontologies

A Dissertation Proposal
Presented to the
Department of Computer Science
Brigham Young University

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

by
Yihong Ding
December 2004

1 Introduction

One major challenge for current web technology is to let machines understand and be able to manipulate web content. The volume of web pages increase tremendously each year. Generic search engines, like *Google* [18], do a phenomenal job of ranking billions of web pages and identifying potentially useful candidates, often presenting the page a user wants within the first few search results. Some domain-specific search engines can work even better on the specialized Web sites, like NEC's *CiteSeer* [8] which helps people find scientific literature. But in general, machines cannot understand the web well because there are too much data out there and they are heterogeneous. The semantic web technology is a way that the W3C (World Wide Web Consortium) recommended to make the whole web machine understandable [45]. Web semantic annotation is one of the important semantic web technologies.

Machines cannot understand web content unless the content is marked and the markups are formally defined, which is exactly the purpose of *web semantic annotation*. To clarify the meaning of semantic annotation, we must distinguish it from other annotations. Semantic annotation is not to add some personal comments or notes into the documents. Any note made by the semantic annotation process must be formally defined and its meaning must be precise, unambiguous, and traceable. These properties guarantee that the results of semantic annotation are machine understandable.

We can divide the web semantic annotation process into three stages. In the first stage, some human or machine agents must create a formal specification of conceptualizations; this is usually called an ontology. In the second stage, based on a defined ontology, some machine or human agents need to find the concept instances inside the target web documents. In the last stage, some machine or human agents need to create the semantic annotation through which machines can understand the content of target web documents.

The history of the semantic annotation can be traced back to ancient Greece when the great Greek philosopher Aristotle described ontology as "the science of being qua being" [3]. For many years after Aristotle, people have studied how ontologies can specify all facts in the world in an unambiguous way. In this sense, the history of semantic annotation is the history of ontologies. A new age of semantic annotation started in the year 1999 when Sir Tim Berners-Lee proposed the idea of the semantic web [6]. The semantic annotation becomes an important way to enable the transformation of current web to be the semantic web, which is also known as the machine-understandable web.

Researchers have proposed three ways to do web semantic annotation. First of all, people have tried to manually annotate web pages with the help of visualized interfaces and optimized annotation storage and sharing mechanism. Many of this type of approaches are not originally designed to do formal semantic annotation. Instead, they let users manually take notes on whatever the users want to keep or share with others. Despite the fact, these systems can be easily adopted to do semantic annotation work by strictly taking notes with respect to a pre-defined ontology. The main research purpose of this type of approaches is not the methods of annotating information but the methods of storing and sharing annotated information. Annotea is a representative example of this type. Annotea uses an RDF-based [44] annotation schema for describing annotations as metadata and XPointer [46] for locating the annotations in the annotated document [25]. The RDF-based annotation schemas are the simpler version of ontologies. Through them, users can apply Annotea system for semantic annotation tasks. [21] has surveyed many other manual annotation systems [11, 17, 22, 35, 36, 41].

Because of the huge volume of web pages, it is very unlikely that people can use manual annotation tools to convert current web to be the semantic web. Automatic semantic annotation tools are required. The second typical way to do semantic annotation is to combine automatic semantic annotation with automatic ontology generation. Much automatic ontology generation research uses machine learning techniques that automatic classify data instances to build the conceptual tree of the ontology [13]. In this paradigm, automatic ontology generation, if it would have worked, would directly lead to semantic annotation results by assigning the generated semantic categories to the corresponding data instances. Although researchers have surveyed many different automatic ontology generation studies, for example [13], SCORE (Semantic Content Organization and Retrieval Engine) is still the only one that has been introduced for semantic annotation research. SCORE employs multiple metadata extractor agents to extract entities using regular-expression-based crawling rules from both semi-structured and unstructured texts [38]. It then applies both classification-based methods and knowledge-base methods to identify entity hierarchies and relationships. The advantage is that it does not require pre-defined ontologies for target web pages. But because of the difficulties encountered by ontology generation research [13], the practicality of this type of approaches has not gotten proved to be successful and is still under study.

The third and final way to do semantic annotation, and the way which is currently the most popular, is to annotate web pages using pre-defined ontologies [12, 20, 26, 42]. This approach assumes that ontologies already exist so that it avoids the very difficult ontology generation problem. These annotation tools depend on the automatic information extraction (IE) tools to help locate concept instances. In order to use these IE tools, users usually have to specify mappings between the used IE tool and the domain ontology. Then users can employ the system to find and annotate data instances inside web documents.

Although results are encouraging for this third approach to semantic annotation, there are three main issues still under exploration. They are the matching problem between domain ontology and IE engine, the concept disambiguation problem and the annotation scalability problem [26, 39]. As we have known, many of the existing IE tools have not used ontologies. Since semantic annotation always requires domain ontologies, it leads to the matching problem between domain ontology and IE engine. During the semantic annotation procedure, the system must be able to distinguish closing concept instances. Because of the potential gigantic numbers, the concept disambiguation problem could be very difficult. As a web application, we must consider the ways to scale the semantic annotation to the size of the whole web. The sizes of application domains could be huge; the numbers of domains could be numerous; the numbers of related web pages could be gigantic. To build a practical semantic annotation system, the scalability problem is definitely an unavoidable one.

Current solutions for the three problems are not good enough to make the semantic annotation practical. The main focus this dissertation is to propose and study the new solutions of these three problems and produce a pure ontology-driven web semantic annotation tool with high accuracy and it is scalable to large size annotation tasks. Correspondingly, I will propose three methods to achieve the resolutions. First, the use of data-extraction ontologies [15] eliminates the matching problem between IE engines and domain ontologies because they directly use ontologies for data extraction purposes. The combination of data-extraction ontologies and semantic web ontologies may further improve the integration of IE tools and domain ontologies. Secondly, IE wrappers using HTML layout information perform fast and have high accuracy when the semantic categories are specified correctly. Ontology-based IE tools have more advantage on concept disambiguation because of their conceptual recognizers with the trade-off that they may perform slower because of the complicated comparisons. By combining the two of them together we can have an advanced resolution for concept disambiguation problem. Finally, with the observation that many large-size annotation applications are actually composed by multiple small-size annotation tasks, I propose an automatic divide-and-conquer mechanism to solve the scalability problem.

The rest of this dissertation proposal is scheduled as follows. Section 2 presents the thesis statement, which summarize the challenges I will accomplish in this dissertation work. Section 3 describes in details the research topics I am going to explore and their relationships to the work of others. After it, Section 4 continues the discussions by illustrating the approaches to be taken in solving the problems, the arguments supporting the chosen approach, and the methods to be followed. Inside Section 4, Section 4.3 lists the artifacts besides the dissertation to be produced and Section 4.4 presents the delimitations of the dissertation. After the discussion of research plans, Section 5 lists the potential research papers that would be generated based on this dissertation work. Section 6 gives a brief list of contributions that the thesis will make. Section 7 makes a schedule for completion of the dissertation. Finally, all the references are listed at the end of the proposal.

2 Thesis Statement

The purpose of this dissertation is to propose a new solution for the web semantic annotation problem. The proposed solution should be able to directly apply ontologies to annotate web pages with high accuracy, and it should be scalable to large domain ontologies. Further, it should also be resilient to the layout of web pages so that it is scalable to large numbers of web pages that may refer to the same application domain. The proposed solution will make the following contributions.

1. It will apply a successful ontology-based data-extraction technique that can automatically extract data instances from web documents according to a given domain ontology.

2. It will show how to compliment current web ontology languages with knowledge recognition power so that people can directly apply web ontologies for annotation work.

3. It will combine concept recognizers and layout-based wrappers so that it can achieve high accuracy and maintain resiliency at the same time.

4. It will use a divide-and-conquer technique that employs multiple small ontologies to solve large domain annotation tasks.

3 Research Description

There are four issues I want to address for this dissertation work: creation of automatic web semantic annotation tools, ontology representation and transformation, accuracy of semantic annotation, and scalability of semantic annotation. in my web semantic annotation research plan. I will discuss each in turn.

3.1 Creation of Automatic Web Semantic Annotation Tools

As mentioned in Section 1, there are two types of automatic semantic annotation tools: either accompanied with automatic ontology generation or accompanied with automatic information extraction. Because of the difficulty of the ontology generation, and also because of our previous experiences with data-extraction ontologies, I only plan to focus on creating automatic web semantic annotation using automatic IE techniques.

Currently there are four major approaches to semantic annotation using automatic IE tools [20, 42, 12, 26]. All of them share some common features. First, they all exclude the ontology generation problem; instead, they assume a given pre-created domain ontology. Second, they all adopt an existing automated IE engine. Thus, because none of the adopted IE engines initially include an ontology, they all need to create a mapping layer between the domain ontology and the adopted IE engine so that they can link the recognized data instances to their semantic categories.

As we have seen, the automated IE tools play an important role on automatic semantic annotation approaches. Researchers have studied automatic information extraction for over a decade. In the most recent survey of automatic IE tools [29], the researchers presents six different categories. I am going to summarize them and discuss the possibilities and suitableness of using them for automatic semantic annotation.

1. Wrapper languages provide formal presentations of extraction patterns [4, 9, 19, 24, 31]. But because the languages are hard to design and the extraction patterns require manual specification, this IE approach is not suitable for automatic semantic annotation. Indeed, no current semantic annotation researchers have tried to employ this type of IE tools into their system.

2. HTML-aware IE tools can provide data extraction results with very high precision and the generation of this type of IE tools can be very much automated [5, 10, 30, 37]. But the automatic created wrappers are trained to find the locations of data of interest instead of the semantic meanings of data of interest. People

may need to create additional semantic labellers to automatic label the recognized data instances. Because the original wrappers are not designed for semantic understanding, the creation of semantic labellers could be very difficult for these IE tools [2, 33].

3. NLP (Natural Language Processing)-based IE tools use NLP techniques, such as filtering, part-of-speech tagging, and lexical semantic tagging, to analysis regular texts [7, 16, 40]. They work very well for unstructured web documents, for instance, the news domain. Probably because many of current semantic annotation researchers are closely related with NLP studies before, three out of the four major current semantic annotation tools adapt this type of IE tools in their system [20, 42, 26]. The limitation of this type of IE tools is that the NLP rules often do not work well for fully structured documents, or even semi-structured documents.

4. ILP (Inductive Learning Processing)-based tools used to deal with data extraction from fully structured and semi-structured documents [23, 27, 34]. Instead of linguistic constraints, ILP-based tools learn the formatting features that implicitly delineate the structure of the pieces of data found. There is one semantic annotation tool that uses this type of IE tools [12].

5. Modeling-based tools design supervised learners for data extraction purposes [1, 28]. People can specify the extraction model by inputting selected training documents through an interactive user-interface either manually or semi-automatically. Because of this reason, it is hard to scale this type of IE tools for large-size applications. Until now, no researchers have tried to employ this type of IE tools for semantic annotation work.

6. Ontology-based IE tools apply pre-defined data-extraction ontologies to do data extraction [15]. Except for the representation of ontologies, we can adopt this type of IE tools for semantic annotation work directly. Ontology-based IE tools is resilient to different layouts of web pages and adaptive to the change of web pages. As a trade-off, it performs slower and may not always guarantee the same high accuracy for some web sites comparing with the other IE tools, especially the pure layout-based ones like HTML-aware tools.

There are no previous semantic annotation approach that uses ontology-based IE tools. This dissertation is the first time to test adopting an ontology-based IE tool, which is data-extraction ontology, to do semantic annotation. The advantages are attracting while there are still some problems we must try to solve. First, we do not need the mapping layer between domain ontologies and adopted IE tools any more. Although ontology representations may be different, both data-extraction ontologies and semantic web ontologies can specify domain knowledge in the same level. The transformation of ontology representations is much easier to be automated than the specification of the mapping layer between domain ontologies and adopted IE tools. Section 3.2 discusses more in detail about the ontology representation and transformation problem.

The resiliency and adaptiveness of ontology-based IE tools give the inherent semantic annotation tool the same properties. The resiliency property lets the corresponding semantic annotation tool be able to annotate information discarding the formatting of web pages. The adaptiveness property assures that we can apply the same tool without further training or adjustments to check, update, and validate annotations. In this sense, we can combine the annotation generation, the annotation validation, and the annotation updating services to be one uniform tool.

Because ontology-based IE tools use conceptual level information to help recognize data instances, they perform better on concept disambiguation cases. But at the same time, the complex conceptual level data recognition makes the performances slower. Section 3.3 discusses more details about the accuracy and speed performance issues.

Another special characteristic of ontology-based IE tools is the comparative independency among the concepts in ontologies. Many of the other IE tools generate wrappers that very much tighten the concepts together with respect to the document structures. This feature limits that the corresponding semantic annotation tool has to apply a full-size domain ontology to annotate the whole domain simultaneously. By adopting ontology-based IE tools, however, we are able to divide the whole domain into multiple small domains and annotate them separately. Through this approach, we may design a new way to scale large size semantic annotation tasks. Section 3.4 discusses more details about the scalability problem.

3.2 Ontology Representation and Transformation

Ontology plays important roles in semantic annotation research. People use ontologies specifying domain knowledge for the semantic annotation. People may also use ontologies to represent results of the semantic annotation. When using ontology-based IE tools, the ontology itself is also the bridge linking the domain knowledge representation and the data recognition procedure.

As we have mentioned earlier, the use of ontology-based IE engines eliminate the need of mappings between adopted IE engines and domain ontologies. Those mappings are tedious to be specified and very hard to be automated. The non-ontology-based IE engines use extraction rules, which do not necessarily reveal the complete semantic meanings of corresponding concepts, instead of individual concept recognizors, which are designed to catch individual concept instances by their semantic meanings, to identify data instances.

Because a data-extraction ontology can provide the same knowledge specification power as for a semantic domain ontology, we can use data-extraction ontologies for both information extraction process and semantic annotation representation. But there are still another problem remaining. The representation used for data-extraction ontologies is different from the representation used for semantic web ontologies. Besides, different ontology representations provide slightly different functions too. It is a challenge for us to find ways that can transform one representation to another and reserve their unique features as much as possible.

3.2.1 Data-Extraction Ontology

The data-extraction ontology is an unique development for ontology-based data extraction [15]. Each data-extraction ontology models a domain with its object sets and the relationships and participation constraints among these object sets. Because of the data extraction purpose, there are two types of object sets defined in data-extraction ontologies. The lexical object sets are the object sets that do contain concrete real-world data instantiations. They are the ones that do extract data from documents. For example, the domain of Car has a lexical object set named $CarMake$ that has an instance $Ford$ in the target document. The nonlexical object sets are the object sets that either are abstract concepts or only implicitly appear in the domain. For example, the domain of Car has a nonlexical object set named $CarId$ that has no real-world explicit instantiations in the target document. Instead, the extraction engine may automatically generate an unique ID, for example $CarNo.29$, for each extracted car instance so that people can have easy index of retrieved information in the storage repository.

A unique feature of data-extraction ontologies is the data frame [14]. A data frame encapsulates the essential properties of every data items such as currencies, dates, weights, and measures. A data frame extends an abstract data type to include not only an internal data representation and applicable operations but also highly sophisticated representational and contextual information that allows a string that appears in a text document to be classified as belonging to the data frame. For instance, a data frame for a phone number has regular expressions that recognize all forms of phone numbers and regular expression recognizers for keywords such as “cell phone” or “home phone” to distinguish phone numbers from one another. Data frames may also simply have lists of values such as a list of all country names; they may also simply have sample values such as some well-known city names.

Data-extraction ontologies use a FOL (First-Order Logic)-based language, which is named the OSML (Object-oriented Systems Model Language). People can specify the conceptual hierarchies and induce their relationships based on the specified hierarchies. People can also easily specify arbitrary n-ary relationships in data-extraction ontologies.

3.2.2 Semantic Web Ontology

Ontology is also an essential component for the semantic web. The semantic web community has proposed to use ontologies to represent web content so that we may build a machine-understandable web. In the mean time, OWL (Web Ontology Language) is the proposed standard to represent a semantic web ontology [43].

OWL can formally describe the semantics of classes and properties used in web documents. OWL ontologies divide the world up into classes of objects with common properties, identify some classes as specializations of others, inherit all of the properties of the more general class and (possibly) add new properties of their own. In addition to this methodology, OWL provides more features to be able to check whether descriptions are logically consistent, to compute class generalization/specialization relationships, and to infer from the description of an individual whether it is a member of a given class. With these abilities to exploit the semantics, OWL can fully provide machine readable forms.

To trade off the balance between the expressivity and the computational complexity, OWL provides three sublanguages with reduced expressivity and computational complexity: OWL-Full, OWL-DL, and OWL-Lite. OWL-Full provides users’ ultimate needs for the maximum expressiveness and the syntactic freedom of RDF but with no computational guarantees. It would be difficult for reasoning software to support complete reasoning for every feature of OWL Full. On the contrary, OWL-Lite supports users’ primary needs for a classification hierarchy and simple constraints. The supporting tool for it is much simpler than for its more expressive relatives, and OWL Lite provides a quick migration path for thesauri and other taxonomies. In the between, OWL-DL supports maximum expressiveness without losing computational completeness and decidability of reasoning systems. OWL-DL, which got the name because it supports the existing description logic business segment, includes almost all of the OWL language constructs but with some restrictions. For example, a class cannot also be an individual or property, and a property cannot also be an individual or class.

3.2.3 Ontology Transformation and Unification

Since the ontologies used for data extraction process and data annotation process are different in their representations, we need to figure out a way to transform them from one to another. As a further step, we want to uniform different ontology representations to make them compatible to both data extraction and data annotation fields.

It is promising that we can compromise the use of data-extraction ontologies and OWL ontologies. Both the ontology languages are a subset of FOL language. Though there are some unique features for each, both data-extraction ontologies and OWL ontologies share many common features, like concept and relationship definitions, participation constraints, concept generalization and specialization, etc. In theory, it is possible to prove that they should have the same capability of representing knowledge and the same reasoning power too. In practice, therefore, we are able to develop a mechanism to transform one representation to another.

It would not be hard to transform the common features between the two ontology representations. Many of these transformations may just be syntactic reformatting. But there are still some common features, the hardness of representing them is different inside the two types of ontologies. For example, both ontologies can specify primitive concepts and defined concepts. While OWL ontologies can directly specify the difference, data-extraction ontologies can only implicitly specify them either by their data frames or by the relationship sets.

More difficulties of transformation are to handle the differences between the two ontological representations. For example, with data-extraction ontologies, people can easily specify n-ary relationships, which is hard with current version of OWL. Inside data-extraction ontologies, users can specify data frames for lexical object set, which is also not possible by using current OWL specifications. From the other side, OWL supports relationship hierarchies, which is not supported by data-extraction ontologies. In addition, there are many more primitive datatypes defined in OWL than in the data-extraction ontology language.

As we mentioned at the beginning of this section, the final aim of ontology representation studies is not the transformation but the unification of the representations. We want to develop an uniformed ontological representation format that has the advantages of both of the current two ontologies, and it is compromise to the knowledge representation of current semantic web society. The latter request is very important because there are many researchers use semantic web ontologies to do their research and they have or will have developed many tools to manipulate the semantic web ontologies. To compromise this standard will let the semantic annotation results be directly applicable for the whole semantic web community.

Because of these reasons, I want to augment current web ontology language to be the web ontology language with semantic annotation augmentation, which may be named as OWL/A (OWL with Annotation augmentation). The purpose of this ontology language augmentation is to add the knowledge recognition power to the current web ontology language beyond the current knowledge specification and knowledge reasoning powers it already has. By this augmentation, the future web ontologies will not only represent knowledge to let web content machine readable, and not only provide reasoning mechanism that future semantic searching engines can rely on, but also they can recognize and annotate information, and even be able to validate the existing annotations.

3.3 Accuracy of Semantic Annotation

There are two accuracy issues in semantic annotation study: concept recognition and concept disambiguation. Concept recognition relates to how well the annotation system can find the data instances with respect to the concepts defined inside the domain ontology. Concept disambiguation relates to how well the annotated data instances are correctly assigned to the corresponding semantic categories. Comparing the two ones, concept recognition issue more relates to the data extraction/recognition process while concept disambiguation issue more concerns about the data annotation process. Again, the two issues are treated separately only because currently the adopted IE engines and the employed domain ontologies are separate. When we combine the IE engine and domain ontologies with the ontology-based IE engine, the two accuracy issues can also be combined into one, which is the accuracy of the whole semantic annotation process.

3.3.1 Concept Recognition Issues

Based on the previous survey [29], many of the existing IE tools may achieve very high concept recognition accuracy when the web pages fit their specific requirements. Their specific requirements, their speed of performance, and their resiliency to the formatting of web pages, however, are quite different from one to another. For the special research interest of this dissertation, I compare the IE tools with the category of layout-based wrappers vs. concept-based wrappers. The pure layout-based wrappers use only the formatting information of web pages to do data extraction. The pure concept-based wrappers use only the specified individual concept recognizers and their conceptual relationships to find data instances with totally discarding the formatting of the web pages. Based on the six categories discussed in Section 3.1, the classic example of pure layout-based wrappers is the HTML-aware tools and the classic example of pure concept-based wrappers is the ontology-based tools. For many others, they are sorts of mixing the two ways.

The greatest benefit of pure layout-based wrappers is their speed of performance. For fully structured documents, the wrappers need only one straight pass of the document with the defined layout information to extract all the possible data instances. With the assumption that the target document must have fit the defined layout, the accuracy of data recognition can be as high as nearly 100%. The trade-off of these wrappers is that they almost totally lose the flexibility of extracting information from web pages that have different formats from the pre-defined one. Another problem, which is not so important for data extraction purposes but is critical for semantic annotation, is that these wrappers do not distinguish the extraction results with their semantic categories but only with their locations inside the pre-defined format. As mentioned in Section 3.1, it is very hard to design the semantic labeller algorithm that can compliments these tools for semantic annotation purposes.

On the contrary, pure concept-based wrappers fulfill the mission that pure layout-based wrappers cannot accomplish. Because they use only concept recognizers, which are independent of web page layouts, they are very much resilient to different types of web pages. As long as the target web page contains a data instance for a concept that they have already defined, they can recognize and extract it correctly most of the time. The accuracy of data recognition only depends on how well these data recognizers are. Comparing with pure layout-based wrappers, the construction of the data recognizers and their relationships is a burden. For example, to build a data-extraction ontology is usually harder than to build a layout pattern for a special web site. Although this is true for pure data extraction purposes, things are different when we consider semantic annotation tasks. Instead of just an option, ontology is a demanding for semantic annotation. Since we have known that the extra semantic labeller is the requirement for layout-based wrappers to do semantic annotation, the creation of data-extraction ontologies is no longer a pitfall. Another problem is the performance of speed. Because concept-based wrappers must traverse all the possible matches of instances and data recognizers, the speed of data recognition is slower than pure layout-based wrappers. This is the problem we must try to solve when we plan to apply ontology-based IE tools for semantic annotation purposes.

The other wrappers mix concept recognizers and layout information together. They have tried to increase the resiliency of the tools while they still preserve the speed of performance as fast as possible. Although they have been proved successfully in their specified data extraction purposes, these kinds of combinations have difficulties for semantic annotation purposes. It is hard to add precise semantic meanings to their extraction categories because of their dependency of layout information. It is desired that we can develop a new combination of concept recognizers and layout information that can balance the resiliency property, the ontological specification, and the speed of performance.

3.3.2 Concept Disambiguation Issues

The problem of concept disambiguation becomes important because of the need of precisely assigning semantic categories to recognized data instances. Because many of current annotation tools do not use ontology-based IE engines, this problem becomes a critical one for semantic annotation studies [39]. The reality is that it is much easier to align the expected results to their corresponding semantic categories before the extraction process than to figure out the corresponding semantic categories out of the extracted results after the extraction process. This is the reason the use of data extraction ontologies could remove the problem of concept disambiguation.

3.3.3 My Solution to Accuracy Issues

In order to get the greatest benefits from both the concept-based and layout-based wrappers, we need to use the pure versions of both. Therefore, instead of mixing, I plan to develop a transformation between two types. The pure ontology-based IE engine will be the base of concept recognition so that the whole system is resilient to the layout of web site. On the top of the base, whenever it has been applied to a large web site, the system automatically create a pure layout-based wrapper that can perform quick annotation on the specified web site. The semantic meanings of extracting categories inside the pure layout-based wrapper is promising because of the underlayer data-extraction ontology. By using this two-layer data recognition prototype, we may achieve the highest degree of resiliency and speed of performance while preserve the high accuracy of data recognition and concept disambiguation.

3.4 Scalability of Semantic Annotation

The scalability problem of semantic annotation is pretty complicated. It contains at least four issues. There are huge numbers of concepts; there are numerous numbers of domains; there may be lots of concepts in one domain; and there are tons of web pages out there to be annotated. Among the four issues, it is very unlikely that we are able to find a good scalable solution for the first one. Concepts are the basic units and unless people have specified the semantic meaning, there are hardly any other ways that machines can automatically figure out a new concept. The last one, on the other hand, tends to be a traditional parallel programming issue, which is not the focus of this dissertation work. The second and third issues are the ones that are special to the semantic annotation studies. They are the focuses of the scalability problem of this dissertation work.

Discarding the slow speed of performance that may be caused by the huge numbers of candidate web pages, the most severer scalability problem for semantic annotation is how to manipulate a large domain. Previous ontology generation studies have taught us that it is very difficult to build a large size domain ontology [13]. Even we have developed large size domain ontologies, our previous data-extraction ontology studies showed that large ontologies causes very slow speed of performance. On the contrary, small size ontologies are easy to be created and fast to be performed. In addition, it is easier to reuse small ontologies than to reuse large ontologies. All these facts suggest that we should try to apply small ontologies for semantic annotation instead of the large ones.

Until now, there are two reported research that deals with the large-size annotation applications, while none of them have explicitly mentioned how to solve the scalability problem of large domain sizes. SemTag annotates a collection of approximately 264 million Web pages and generate approximately 434 million automatically disambiguated semantic tags [12]. Although they have annotated web pages of many domains, the total number of concepts is very limited. They are interested in only 24 internal nodes inside an ontology. The main purpose of their research is to test a new concept disambiguation algorithm that can precisely categorize data items into their semantic categories. They focused only on the fourth scalability issue I mentioned earlier. Another group of researchers reported that they have annotated over million numbers of instances with a total of 4 to 8 weeks of effort [39]. But they have not mentioned how many concepts involved and how many domains they dealt with. Instead, in the same paper, the authors said that both the concept disambiguation and scalability problems are the most challenging issues that they were still struggling with.

My observation is that we can split most of the large domain annotation problems into a combination of multiple small domain annotation problems through a divide-and-conquer style of algorithm. There are many small domains that are static to the participating larger domains. For example, no matter there is a automobile shopping event web page provided by a dealer or a travellings event in a school's web site that mentions the travelling van, an automobile is just an automobile. It has make, model, year, features, and so on. The difference is that a selling event may include the price of the car and a travelling event may have a date the van is driven, while both the information are independent to the small domain *Automobile*.

The idea of my scalability algorithm is as follows. First I will build a library of small domain ontologies. For any given annotation application, I will first apply an information retrieval technique to identify the applicable domains in the library that are contained in the target domain. Then I can employ the semantic annotation technique to annotate the related web pages using the selected small domain ontologies. Finally, I can merge the annotated results with respect to the user specified relationships among the individual domains and output the final annotations for the target domain.

Based on the paradigm I presented, people only need to create these small domain ontologies once and they are very much reusable for different applications. It also means that we may figure ways to optimize the annotation for each individual small domain ontologies without worrying about the others. The only required human effort in this paradigm is the specification of relations that link the small domains. But this work is much more easier comparing with the effort to create a whole ontology for the target domain. If the users can specify the relations among the potential small domains before hand, it is possible for people to fully automate the whole annotation process.

4 Research Plan

I plan to build a semantic annotation demo to illustrate my approach. This demo will be a prototype system that can automatically annotate Web pages so that machine agents can understand them. As a base platform and also a connection to other work in our research lab, this demo will be extendable to support future semantic searching and semantic reasoning. I will run experiments against real web sites to evaluate the performance of the system. I will divide my research plan into two parts: (1) semantic annotation prototype system creation and augmentation, and (2) semantic annotation representation and language augmentation. Essentially there are no sequential limits on the two parts. Although they are closely related with each other, each of them can be studied independently.

4.1 Prototype System Creation and Augmentation

The *Ontos* extraction engine that uses data-extraction ontologies is used as the base platform for the new semantic annotation prototype system. The data-extraction ontology approach is a unique ontology-based IE approach developed in BYU. In theory, because it uses pre-defined ontologies to extract data instances, we need only add an attachment to the existing engine to let it assign corresponding semantic tags to the annotated content. In practice, however, the work is far more complicated.

4.1.1 Basic Prototype System

One of the major challenges of semantic annotation study is the standard of annotation representation. There are no formal standard now. The annotation representation should not only be easy to represent semantic meanings, but also be practical for future semantic searching and other semantic web manipulations. I want to propose an ontology-based semantic disclosure file, which may be called an instance ontology, to represent annotations. The use of standard ontology format will make the annotations machine understandable and manipulatable. The details of the instance ontology are still under exploration.

To enable the ontology-based IE tool for semantic annotation tasks, I need to augment current *Ontos* engine with the position information of the data instances and let the system generate instance ontologies as the output. The position information is very important for semantic annotation because the same instance may appear multiple times in the same web page. In the different positions, the same instance may have different connections with different other instances. Sometimes, machines need to know that even the same instance may mean different when it is in different positions.

When the annotation is done, I need to design or adapt a storage repository system to store and share the annotated information. For a simpler prototype system, the repository may be a straight directory structure. But it can be very complicated when we need to deal with numerous domains and when there are many users who want to share the annotations. This research topic is, however, a little bit out of the focus of this dissertation. Because of this reason, I would like to put the further studies of the annotation storing, indexing, and sharing problem as a future work.

To test the performance of the basic semantic annotation prototype, I plan to run it over about 5∼10 domains with different sizes. During the experiments, I will calculate the precisions and recalls of the results. The precision is how well the annotated results are correct; and the recall is how complete the system has covered the domains. In the mean time, I may also want to compare the performances due to different domain sizes. The performance comparison may include precision, recall, and speed issues.

4.1.2 Prototype System Augmentations

Whenever the basic semantic annotation system is built up, I need to augment it to improve its performance on both its accuracy and its scalability. The augmentations, however, should not hurt the original nice features of data-extraction ontologies, for example, the highly resiliency to the layouts of web pages.

One augmentation is to compliment the ontology-based extractor with layout-based wrappers. Based on the same preassumption used by the RoadRunner [10], i.e., the web pages generated using automatic tools always follow a common layout pattern, we can automatically create a layout-based wrapper for each of these web sites. This process can be fully automated. For each of these large automatically created web site, I arbitrarily take two web pages and annotate them using the original ontology-based extractor. By comparing the annotated positions in the two web pages, the system can figure out their common features surrounding the annotated locations. Through the structural analysis, the system can automatically create a layout-based wrapper that fit for the specific web site. The difference of this approach from original RoadRunner extractor is that the latter one does not have the capability to know the semantic meanings of the extracting patterns while our new augmentation does have the power because of the use of ontologies as the base of extraction.

Our new proposed augmentation can actually do more beyond the limit of the RoadRunner, which only works for the large automatically created web sites. Even for the web pages that do not fall into the described category, this augmentation can be very helpful for validating and updating annotations. The system can always create a pure layout-based wrapper for each web page fully automatically. Later on, with taking the advantages of the very high speed of performance of the layout-based wrappers, machine agents can go to use these wrappers either to validate the correctness of the annotations or to check the necessariness of updating the annotations.

To test this part of augmentations, I plan to run it over about 5 large web sites and compare the precision, recall, and speed issues of the generated layout-based wrappers with these of the original pure ontology-based annotator. Further, I plan to run it over about 10 individual web pages to see how well the augmentation can be to improve the performance of annotation validation studies. This experiment also includes the comparison of precision, recall, and speed issues between the created layout-based wrappers and the original ontology-based annotator. Moreover, I plan to compare these performance issues with other published annotators on their domains. But the success of this part of work depends on the availability of the resources I can obtain.

Another augmentation is to scale the semantic annotation to handle large domains without building large scale ontologies. First, I need to build many small domain ontologies. This manual craft step is unavoidable because they are the knowledge base on which machines rely. For this dissertation work, I plan to create at least 20 well-tested small domain ontologies so that there is a big enough base to do large size annotation tests.

After I have created a large enough base of small ontologies, I will choose some web sites that contain large domains for my testing base for scalability experiments. I plan to test on about 5 large domains that contains from 2 to 10 small domains and test the precision, recall, and speed issues of the scalability algorithm.

When I have chosen a large domain application, I first adopt an existing text classification tool, e.g. the rainbow classifier [32], to select the potential small domains to be annotated. I then apply the selected small domains to annotate the web pages for the large domain. Since each of the small domain is only a subset of the large domain, every process may only annotate part of the web pages. After they have done, I can integrate the annotation results and merge them to be one with the help of user specified rules among the selected small domains.

4.2 Semantic Annotation Language Augmentation

Current web ontology languages only focus on knowledge specification and knowledge reasoning. As we have seen, in order to make semantic annotation compliment to semantic web studies, we'd better augment web ontology languages themselves the capability of knowledge recognition.

The first step of this part of the research is to develop a transformation mechanism between OWL and OSML. I plan to do both the theoretic proof and the practical implementation of the transformation study.

In theory, I want to prove that OWL and OSML have the same knowledge specification and knowledge reasoning powers except that they may have different favorites on displaying individual features. In order to prove the same knowledge specification power, they should be able to specify concepts, the relationships between the concepts, and the possible constraints within the relationships. Also they should be able to correctly illustrate the conceptual hierarchies and relationship hierarchies. In order to prove the same knowledge reasoning power, they should be able to answer the same set of queries with the same answers based on the same conceptual model discarding that their representations are different. Upon to my knowledge, the answers to all of these requirements are positive. Therefore, such a proof should be sound and complete.

The practical transformation implementation is harder than the theoretic proof. Although they are equivalent on the knowledge specification and knowledge reasoning powers, the representations are very much different with each other. In addition, one representation may favor on some features while the other one does not. This fact causes the very much different degrees of difficulties to represent those features in the two presentations. I will carefully enumerate all the features and transform them from one to another representations in implementation.

As a next step of the research, I plan to augment OWL with the annotation extension (OWL/A) so that the web ontology language itself can have the knowledge recognition power. In order to accomplish this plan, I need to enrich the set of OWL tags to be able to describe knowledge recognition pattern, if applicable, with the concepts. By this improvement, I need to modify the semantic annotation system to take OWL/A ontologies instead of data-extraction ontologies as the inputs to annotate web pages. The modification can base on the language transformation tool so that users can even adopt the existing OWL ontologies complimenting with concept recognizers for semantic annotation tasks. After the language augmentation, I will test the system using the same domains tested before to see how well it performs comparing with using the data-extraction ontologies.

4.3 Deliverable Artifacts

When the dissertation is finished, the following artifacts are expected to be available.

- prototype of a web semantic annotation toll based on data-extraction ontologies;
- annotation storage repository;
- a prototype of a tool that can transform data-extraction ontologies to OWL ontologies and vice versa; and
- a proposed OWL ontology language augmented with a knowledge recognition compliment;

4.4 Delimitations of the Dissertation

In this dissertation, I assume that the following are not part of this research study.

- Automatic ontology creation
- Annotation storing, indexing, and sharing infrastructure
- Scale the semantic annotation approach to millions of web pages

5 Research Papers

The following are proposed titles of papers.

- Semantic Web Annotation – Using Ontology-based Information Extraction
- Highly Accurate and Resilient Semantic Annotation – A Combination of Data-Extraction ontologies and Layout-based Wrappers
- Using Small Domain Annotators to do Large Web Semantic Annotation Tasks – A Divide-and-Conquer Method
- The Similarity Comparison between Data Extraction Ontologies and OWL Ontologies – A Theoretic Study
- OWL/A: A Knowledge Recognition Augmentation of OWL for Annotation

6 Contribution to Computer Science

My vision is to provide a generic web semantic annotation method to transform current web to be the semantic web. The new method will propose a new representation of semantic annotation, which wishes to contribute for the web semantic annotation standard. The new method will annotate web pages with high accuracy and high speed of performance, while it still inherits the original advantage of highly resilient to web page layouts. Because the new method directly uses ontologies to recognize data instances, it eliminates the extra mappings between IE engines and domain ontologies when using other types of IE engines. This research also tries to extend the capability of ontologies from their original knowledge specification and knowledge reasoning powers to obtain knowledge recognition power. This new method proposes a divide-and-conquer fashion method to solve the scalability problem. It avoids the very complicated problem of building large size ontologies and applying them for huge size annotation applications.

7 Dissertation Schedule

The following schedule shows the expected deadline of each step of my dissertation work.

- Basic ontology-based web semantic annotation tool (transformed from original Ontos tool) (January 2005)
- The data-extraction ontology to OWL ontology transformation tool (including the theoretic proof of the equivalence between the two) (April 2005)
- Layout-based IE extension for the web semantic annotation tool (October 2005)
- Instance ontology merging tool for the scalability problem (April 2006)

References

[1] B. Adelberg. Nodose – a tool for semi-automatically extracting structured and semistructured data from text documents. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 283–294, Seattle, Washington, 1998.

[2] H. Alani, S. Kim, D. Millard, M. Weal, W. Hall, P. Lewis, and N. Shadbolt. Web based knowledge extraction and consolidation for automatic ontology population.

In *Proceedings of Knowledge Markup and Semantic Annotation Workshop, 2nd International Conference on Knowledge Capture (K-CAP'03)*, Sanibel Island, Florida, October 2003.

[3] Aristotle. *Metaphysics* (about 350 BC). Oxford University Press, New York, 1993.

[4] G. O. Arocena and A. O. Mendelzon. Weboql: Restructuring documents, databases, and webs. In *Proceedings of the 14th International Conference on Data Engineering*, pages 24–33, Orlando, Florida, 1998.

[5] R. Baumgartner, S. Flesca, and G. Gottlob. Visual web information extraction with lixto. In *Proceedings of the 26th International Conference on Very Large Data Bases (VLDB)*, pages 119–128, Rome, Italy, 2001.

[6] T. Berners-Lee. *Weaving the Web*. Harper SanFrancisco, San Francisco, California, 1999.

[7] M. E. Califf and R. J. Mooney. Relational learning of pattern-match rules for information extraction. In *Proceedings of the 16th National Conference on Artificial Intelligence and 11th Conference on Innovative Applications of Artificial Intelligence*, pages 328–334, Orlando, Florida, 1999.

[8] Citeseer paper search. http://citeseer.ist.psu.edu.

[9] V. Crescenzi and G. Mecca. Grammars have exceptions. *Informaion Systems*, 23(8):539–565, 1998.

[10] V. Crescenzi, G. Mecca, and P. Merialdo. Roadrunner: Towards automatic data extraction from large web sites. In *Proceedings of the 26th International Conference on Very Large Data Bases (VLDB)*, pages 109–118, Rome, Italy, 2001.

[11] The critlink mediator. http://www.doc.ic.ac.uk/~ids/dotdot/misc/research_related/AnnotatingTheWeb/CritSuite/crit-0.9.2-ids-0.0.1-as-cgi-script/web/.

[12] S. Dill, N. Eiron, D. Gibson, D. Gruhl, R. Guha, A. Jhingran, T. Kanungo, K. S. McCurley, S. Rajagopalan, A. Tomkins, J. A. Tomlin, , and J. Y. Zien. A case for automated large scale semantic annotations. *Journal of Web Semantics*, 1(1):115–132, December 2003.

[13] Y. Ding and S. Foo. Ontology research and development: Part 1 a review of ontology generation. *Journal of Information Science*, 28(2):123–136, February 2002.

[14] D. W. Embley. Programming with data frames for everyday data items. In *Proceedings of AFIPS National Computer Conference (NCC80)*, pages 301–305, Anaheim, California, May 1980.

[15] D. W. Embley, D. M. Campbell, Y. S. Jiang, S. W. Liddle, Y.-K. Ng, D. Quass, and R. D. Smith. Conceptual-model-based data extraction from multiple-record web pages. *Data Knowledge Engineering*, 31(3):227–251, 1999.

[16] D. Freitag. Machine learning for information extraction in information domains. *Machine Learning*, 39(2/3):169–202, 2000.

[17] G. Gay, A. Sturgill, and W. Martin. Document-centered peer collaborations: An exploration of the educational uses of networked communication technologies. *Journal of Computer-Mediated Communication*, 4(3), March 1999.

[18] Google Search Engine. http://www.google.com.

[19] J. Hammer, J. McHugh, and H. Garcia-Molina. Semistructured data: The tsimmis experience. In *Proceedings of the First East-European Symposium on Advances in Databases and Information Systems*, pages 1–8, St. Petersburg, Russia, 1997.

[20] S. Handschuh, S. Staab, and F. Ciravegna. S-CREAM — Semi-automatic CREAtion of Metadata. In *Proceedings of the European Conference on Knowledge Acquisition and Management (EKAW-2002)*, Madrid, Spain, October 1-4, 2002.

[21] R. M. Heck, S. M. Luebke, and C. H. Obermark. A survey of web annotation systems. Technical report, Grinnell College, Grinnell, Iowa, 1999.

[22] K. Holtman. The futplex system. In *Proceedings of the ERCIM workshop on CSCW and the Web*, Sankt Augustin, Germany, February 7-9 1996.

[23] C.-N. Hsu and M.-T. Dung. Generating finite-state transducers for semi-structured data extraction from the web. *Information Systems*, 23(8):521–538, 1998.

[24] G. Huck, P. Fankhauser, K. Aberer, and E. J. Neuhold. Jedi: Extracting and synthesizing information from the web. In *Proceedings of the 3rd International Conference on Cooperative Information Systems (IFCIS)*, pages 32–43, New York City, New York, 1998.

[25] J. Kahan, M.-R. Koivunen, E. Prud'Hommeaux, and R. R. Swickd. Annotea: An open rdf infrastructure for shared web annotations. In *Proceedings of The Tenth*

International World Wide Web Conference, pages 623–632, Hong Kong, China, May 1-5 2001.

[26] A. Kiryakov, B. Popov, I. Terziev, D. Manov, and D. Ognyanoff. Semantic annotation, indexing, and retrieval. *Journal of Web Sematics*, 2(1):49–79, December 2004.

[27] N. Kushmerick. Wrapper induction: Efficiency and expressiveness. *Artificial Intelligence Journal*, 118(1-2):15–68, 2000.

[28] A. H. F. Laender, B. Ribeiro-Neto, and A. S. DaSilva. Debye – data extraction by example. *Data and Knowledge Engineering*, 40(2):121–154, 2002.

[29] A. H. F. Laender, B. A. Ribeiro-Neto, A. S. da Silva, and J. S. Teixeira. A brief survey of web data extraction tools. *SIGMOD Record*, 31(2):84–93, June 2002.

[30] L. Liu, C. Pu, and W. Han. Xwrap: An xml-enabled wrapper construction system for web information sources. In *Proceedings of the 16th International Conference on Data Engineering (ICDE)*, pages 611–621, San Diego, California, 2000.

[31] B. Ludascher, R. Himmeroder, G. Lausen, W. May, and C. Schlepphorst. Managing semistructured data with florid: A deductive object-oriented perspective. *Informaion Systems*, 23(8):589–613, 1998.

[32] A. K. McCallum. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering, 1996. http://www.cs.cmu.edu/∼mccallum/bow.

[33] S. Mukherjee, G. Yang, and I. V. Ramakrishnan. Automatic annotation of content-rich html documents: Structural and semantic analysis. In *Proceedings of Second International Semantic Web Conference (ISWC 2003)*, Sanibel Island, Florida, October 2003.

[34] I. Muslea, S. Minton, and C. Knoblock. Hierarchical wrapper induction for semistructured information sources. *Autonomous Agents and Multi-Agent Systems*, 4(1/2):93–114, 2001.

[35] I. A. Ovsiannikov, M. A. Arabib, and T. H. McNeill. Annotation technology. *International Journal of Human-Computer Studies*, 50:329–362, 1999.

[36] M. Roscheisen, C. Mogensen, and T. Winograd. Shared web annotations as a platform for third-party value-added information providers: Architecture, protocols,

and usage examples. Technical report, Stanford University, Stanford, California, 1997.

[37] A. Sahuguet and F. Azavant. Building intelligent web applications using lightweight wrappers. *Data and Knowledge Engineering*, 36(3):283–516, 2001.

[38] A. Sheth, C. Bertram, D. Avant, B. Hammond, K. Kochut, and Y. Warke. Managing semantic content for the web. *IEEE Internet Computing*, 6(4):80–87, July/August 2002.

[39] A. Sheth and C. Ramakrishnan. Semantic (Web) technology in action: Ontology driven information systems for search, integration and analysis. *IEEE Data Engineering Bulletin*, 26(4):40–48, December 2003.

[40] S. Soderland. Learning information extraction rules for semi-structured and free text. *Machine Learning*, 34(1-3):233–272, 1999.

[41] Thirdvoice web annotator. http://c2.com/cgi/wiki?ThirdVoice.

[42] M. Vargas-Vera, E. Motta, J. Domingue, M. Lanzoni, A. Stutt, and F. Ciravegna. Mnm: Ontology driven tool for semantic markup. In *Proceedings of the Workshop Semantic Authoring, Annotation & Knowledge Markup (SAAKM 2002) with ECAI 2002*, Lyon, France, July 2002.

[43] W3C (World Wide Web Consortium) *OWL Web Ontology Language Reference.* http://www.w3.org/TR/owl-ref/.

[44] Resource Description Framework. http://www.w3.org/RDF/.

[45] W3C (World Wide Web Consortium) *Semantic Web Activity Page.* http://www.w3.org/2001/sw/.

[46] XML Pointer Language (XPointer). http://www.w3.org/TR/WD-xptr.

This dissertation proposal by Yihong Ding is accepted in its present form by the Department of Computer Science of Brigham Young University as satisfying the dissertation proposal requirement in the Department of Computer Science for the degree of Doctor of Philosophy.

David W. Embley, Committee Chair

Deryle W. Lonsdale, Committee Member

Stephen W. Liddle, Committee Member

William A. Barrett, Committee Member

Eric G. Mercer, Committee Member

David W. Embley, Graduate Coordinator