OntoCrawler: A focused crawler with ontology-supported website models for information agents

Sheng-Yuan Yang

Department of Computer and Communication Engineering, St. John's University, Taipei, 499, Sec. 4, TamKing Rd., Tamsui, Taipei County 25135, Taiwan, ROC

Abstract

This paper proposed the use of ontology-supported website models to provide a semantic level solution for an information agent so that it can provide fast, precise, and stable query results. We have based on the technique to develop a focused crawler, namely, OntoCrawler which can benefit both user requests and domain semantics. The technique in this research has practically applied on Google and Yahoo searching engines to actively search for webpages of related information, and the experiment outcomes indicated that this technique could definitely up-rise precision rate and recall rate of webpage query. Equipped with this technique, we have developed an ontology-supported information agent shell in Scholar domain which manifests the following interesting features: ontology-supported construction of website models, website models-supported website model expansion, website models-supported webpage retrieval, high-level outcomes of information recommendation, and accordingly proved the feasibility of the related techniques proposed in this paper.

1. Introduction

In this most quickly developed and shifting era of Internet, huge amount of information makes users ambiguous to adapt what information is necessary. How to search advantage information has become essential part in users. Current domain-specific search engines do help users to narrow down the search scope by the techniques of Query Expansion, Automatic Classification and Focused Crawling; their weakness, however, is almost completely ignoring the user interests (Wang, 2003). New standards for representing website documents, including XML (Henry, David, Murray, & Noah, 2001), RDF (Brickley & Guha, 2004), DOM (Arnaud et al., 2004), Dublin metatag (Weibel, 1999), and WOM (Manola, 1998), can help cross-reference of Web documents; they alone, however, cannot help the user in any semantic level during the searching of website information. OIL (2000), DAML (2003), and DAML+OIL (2001) and the concept of ontology stand for a possible rescue to the attribution of information semantics. In this paper, we advocated the use of ontology-supported website models to provide a semantic level solution for an information agent so that it can provide fast, precise, and stable query results.

We noticed that the concept of crawler is mostly used in the Web systems that work on information gathering or integration to improve their gathering processes or the search results from disparate resources. For instance, Dominos (Hafri & Djeraba, 2004) can crawl several thousands of pages every second, include a high-performance fault manager, be platform independent, and adapt transparently to a wide range of configurations without incurring additional hardware expenditure. Ganesh, Jayaraj, Kalyan, and Aghila (2004) proposed the association-metric to estimate the semantic content of the URL based on the domain-dependent ontology, which in turn strengthens the metric that is used for prioritizing the URL queue. UbiCrawler (Boldi, Codenotti, Samtini, & Vigna, 2004), a scalable distributed Web Crawler, was platform independent, linear scalability, graceful degradation in the presence of faults, a very effective assignment function for partitioning the domain to crawl, and more in general, the complete decentralization of every task. In this paper, we developed an OntoCrawler using ontology and website models as the core techniques, which can help information agents to successfully tackle the problems of search scope and user interests.

The Scholar domain was chosen as the target application of the system and will be used for explanation in the remaining sections. The rest of the paper is organized as follows. Section 2 develops the domain ontology. Section 3 describes Website models and how they are constructed. Section 4 illustrates how Website models can be used to do better Web search. Section 5 describes the design of our information agent shell and reports how it performs. Section 6 discusses related works, while Section 7 concludes the work.
2. Domain ontology as fundamental semantics

Ontology provides complete semantic models, which means in specified domain all related entities, attributes, and base knowledge among entities owning sharing and reusing characteristics which could be used for solving the problems of common sharing and communication (Yang & Ho, 1999). Protégé 3.3.1 (Grosso et al., 1999) was adapted in this paper, which supports two main ways of modeling ontologies via the Protégé-Frames and Protégé-OWL (adapted by us, shown in Fig. 1) editors. Protégé ontologies can be exported into a variety of formats including RDF(s), OWL, and XML Schema and lead knowledge workers to constructing knowledge management system based on ontology; furthermore, users could transfer to different formats of ontology such as RDF(S), OWL, XML or directly inherit into database just like MySQL and MS SQL Server (adapted by us and described later), which have better supported function than other tools (Chien, 2006).

Nowadays, the research on ontology can be branched into two fields: one is to configure huge ontology in a specified field and through it to assistant the knowledge analysis in this field; the other is to study how to construct and express precisely with ontology (Chien, 2006). In this paper, we adapted the former in which took advantage of built ontology database of some scholars to support OntoCrawler for querying webpage of related scholars, detailed in Yang and Hsu (2008). Briefly, we conducted statistics and survey of homepage of related scholars to fetch out the related concepts and their synonym appearing in the homepage. The second stage of ontology constructing of scholars is to transfer the ontology built with Protégé into MS SQL database. The procedures are as following:

1. With Protégé to define the scholars’ ontology analyzed from the first stage so as to share the ontology with other interesting researchers.
2. Exporting an XML file constructed with Protégé knowledge base and then importing into MS Excel for correcting.
3. Finally, importing MS Excel into MS SQL Server to finish the ontology construction of this system.

Fig. 1 indicated the structure of domain ontology of scholars in Protégé, taking the middle frame of the screen for instance, the related concept “Education” was linking behind with “M.S.”, “PH.D.”, and “B.S.”. In application, we define those as related concepts and that means “education” is nothing but a combination of these related concepts that would be conveniently interpreted by OntoCrawler to compare with content of the queried webpage, and if the compared outcomes corresponding to any item among the four, we would infer the related concept “Education” as matched condition for web page querying. Finally, we used Protégé’s APIs (http://protege.stanford.edu/doc/pdk/api/) to develop a set of ontology services, which provide primitive functions to support inference of the ontologies. The ontology services currently available include transforming query terms into canonical ontology terms, finding definitions of specific terms in ontology, finding relationships among terms, and finding compatible or conflicting terms against a specific term, etc.

3. Website model and construction

Fig. 2 illustrates the structure and example of a website model (Yang, 2008). The webpage profile contains three sections, namely, basic information, statistics information, and ontology information. The first two sections profile a webpage, and the last annotates domain semantics to the webpage. DocNo is automatically generated by the system for identifying a webpage in the structure index. Location remembers the path of the stored version of the Web page in the website model; we can use it to answer user queries. URL is the path of the webpage on the Internet, same as the returned URL index in the user query result; it helps hyperlinks analysis. WebType identifies one of the following six Web types: com (1), net (2), edu (3), gov (4), org (5), and other (0), each encoded as an integer in the parentheses. WebNo identifies the website that contains this webpage. Update_Time/Date remembers when the webpage was modified last time. The statistics information section stores statistics about HTML tag properties. Specifically, we remember the texts associated with Titles, Anchors, and Headings for webpage analysis; we also record Outbound_URLs for user-oriented webpage expansion. Finally, the ontology information section remembers how the webpage is interpreted by the domain ontology. Domain_Mark is used to remember whether the webpage belongs to a specific domain. This section annotates how a webpage is related with the domain and can serve as its semantics, which help a lot in correct retrieval of webpages.

Let us turn to the website profile. WebNo identifies a website. Through this number, we can access those webpage profiles describing the webpages that belong to this website. Website_Title remembers the text between tags <TITLE>of the homepage of the website. Start_URL stores the starting address of the website. WebType identifies one of the six Web types as used in the webpage profile. Tree_Level_Limit keeps the search agent from exploring too deeply. Update_Time/Date remembers when the website was modified last time. This model structure helps interpret the semantics of a website through the gathered information; it also helps fast retrieval of webpage information and autonomous search of Web resources.

During the construction and expansion process of a website model, we need to extract primitive webpage information as well as to perform statistics. Website modeling involves three modules, detailed in Yang (2006c). Briefly, we use OntoExtractor to extract basic webpage information and perform statistics. We then use OntoAnnotator to annotate ontology information. Since the ontology information contains webpage classes, OntoAnnotator needs to call OntoClassifier (described later) to perform webpage classification. In order to facilitate these activities, we have re-organized the
ontology structure into a two-layer structure (super and reference classes) (Yang, 2006a), which stresses on how concept attributes are related to class identification. Each super class contains a set of representative ontology features for a specific concept, while each reference class contains related ontology features between two concepts. This design clearly structures semantics between ontology classes and their relationships and can serve as a fast semantics decision mechanism for website expansion. We have proposed an ontology-directed classification mechanism, namely, OntoClassifier (Yang, 2006a) to make a decision of the class for a webpage or a website. OntoClassifier is a two-step classifier based on the deliberately organized ontology structure and can do very accurate and stable classification on web pages to support Web search. Briefly, the first stage uses a set of representative ontology features for measuring how strong a webpage/website is related to a specific class by calculating the number of ontological features of a class that appears in a webpage/website. If for any reason the first stage cannot return a class for a webpage/website, we move to the second stage of classification. It employs another set of related ontology features with a level-related weighting mechanism for webpage/website classification.

4. Website models application

4.1. Focused web crawling supported by website models

We proposed a focused crawler namely OntoCrawler as shown in Fig. 3, which featured a progressive crawling strategy in obtaining domain-relevant Web information. Inside the architecture, Web Crawler gathers data from the Web, detailed in Section 4.2. DocPool stores all returned Web pages from Web Crawler for OntoExtractor (Yang, 2006c) during the construction of webpage profiles. It also stores query results from search engines, which usually contains a list of URLs. URLExtractor is responsible for extracting URLs from the query results and dispatching those URLs that are domain dependent but not yet in the website models to Distiller. User-Oriented Webpage Expander pinpoints interesting URLs in the website models for further webpage expansion. User Priority Queue stores the user search strings and the website model URLS from User-Oriented Webpage Expander. Website Priority Queue stores the website model URLs from Autonomous Website Evolver and the URLs extracted by URLExtractor.

Distiller controls the Web search by associating a priority score with each URL (or search string) and placing it in a proper Priority Queue. We defined the ULScore as the priority score for each URL (or search string), detailed in Yang (2006b). Briefly, all search strings are treated as the top-priority requests, website model URLs are second-priority, and URLs extracted by URLExtractor are last-priority. This design prefers user-oriented Web resource crawling to website maintenance, since user-oriented query or webpage expansion takes into account both user interest and domain constraint, which can better meet our design goal than website maintenance.

4.2. Web Crawler operation and techniques

Fig. 4 showed the operation system structure of Web Crawler (developed with Java), and related techniques and functions of every part were described as below.

1. Action: transfer internal query into URI code, and then embed into Google's query URL: an example as follow http://www.google.com.tw/search?hl=zh-TW&q=%E5%AD%B8%E8%80%85&meta=.

2. Google/Yahoo Search Machine: declare an URL object, and add Google query URL on well-transferred URI code, and then used an iterative loop to read its content line by line. Finally, output the content as text file as final analysis reference, which was the html source file of the webpage.

3. RetrieveLinks: use regular expression, detailed in Yang and Hsu (2008), to search for whether there are matched URL. But it could not retrieve all the linkages in one time out because of the Google/Yahoo webpage editing with indenting. So we used an iterative loop and ran for twice. The semantic of the two in regular expression were slightly different so as to completely fetch out related hyperlinks corresponding to the conditions. Finally, returned all hyperlinks and output them into a text file to provide the system for further processing.

4. RetrieveContent: use the hyperlink file to read its content with an iterative loop line by line that meant we checked one URL link once a time and really linked the URL. After

Fig. 2. Website model structure and example.
judging what kind coding of the webpage was, we read in
the html source file of webpage with correct coding and out-
put it as text file so as to let system conduct further process-
ing. After completing all procedures mentioned above, we
could used SearchMatches method (described later) to judge
whether the webpage was located in the range we hoped to
query; supposed the answer was “yes”, we would execute
RemoveHTMLTags (described later) to delete the html label
from source file and remained only the text content so as
to let system conduct further processing and analyzing.
Finally, we collected the number of queried webpage, and
divided with total of the webpage, and the mean we got
was the percentage of query processing.

(5) SearchMatches: support RetrieveContent internal calling
service to judge whether the webpage was the range we
queried. It linked the ontology database and fetched out
the content to compare content of the webpage. If there
were any return value corresponding to the value we set,
and then system would return one “true” Boolean variable
“matches.” That meant the webpage matched our query con-
dition, on the other hand, if returned “false” meant the web-
page did not match our query condition.

(6) RemoveHTMLTags: just like SearchMatches, it supported
RetrieveContent internal calling service and deleted html
tags in the html source file.

(7) DelDURLs: proceed complete cross-comparison with URLs
returned by Google and Yahoo for deleting duplication web-
pages to avoid the duplicated operation of system backend
and accordingly enhance its performance.

4.3. User- and domain-oriented web search supported by website
models

The basic goal of the website models is to help Web search in
both a user-oriented and a domain-directed manner. We proposed
a direct query expansion mechanism, which adds the synonyms of
terms contained in the user query into the same query. More com-
licated expansion adds ontology concepts according to their rela-
tionships with the query terms. The most used relationships follow
the inheritance structure. We also proposed an implicit webpage
expansion mechanism oriented to the user interest to better cap-
ture the user intention. Here we exploited the outbound hyperlinks
of the stored webpages in the website models, detailed in Yang
(2006b). Briefly, we proposed a strategy to select those hyperlinks,
or URLs, that the users are strongly interested in according to the
degree of domain correlation of a website with respect to the do-
main, which needs the parameter Domain_Mark in the webpage
profile to determine it.

We then employed an implicit webpage expansion mechanism
which consulted the user models (Yang, 2009) for user interests
and used that information to add more webpages into the website
models by, for example, checking on how the anchor texts of the
outbound hyperlinks of the webpages in the website models were
strongly related with the user interests. We also employed a four-
phase progressive strategy to do website expansion, i.e., to add
more domain-dependant webpages into the website models, de-
tailed in Yang (2006b). Briefly, the expansion strategy starts with
the first phase, which expands the websites that are well profiled
in the website models but have less coverage of domain concepts;
the second phase then searches for those webpages that can help bring in more information to complete the specification of indefinite website profiles; the third phase collects every webpage that is referred to by the webpages in the website models; and finally, the last phase resorts to general website information to refresh and expand website profiles.

4.4. Webpage retrieval from website models

Webpage retrieval concerns the way of providing most-needed documents for users. Traditional ranking methods employ an inverted full-text index database along with a ranking algorithm to calculate the ranking sequence of relevant documents. The problems with this method are clear: too many entries in returned results and too slow response time. A simplified approach emerged, which employs various ad hoc mechanisms to reduce query space; however, they need a specific, labor-intensive and time-consuming pre-process, and they cannot respond to the changes of the real environment in time due to the off-line pre-process. Another method called PageRank (Page, Brin, Motwani, & Winograd, 1999) was employed in Google to rank webpages by their link information. Google’s high speed of response stems from a huge local webpage database along with a time-consuming, offline detailed link structure analysis.

Our solution ranking method takes advantage of the semantics in the website models, as shown in Fig. 5. The major index structure uses ontology features to index webpages in the website models. The second index structure is a partial full-text inverted index since it contains no ontology features. Since we require each query contain at least one ontology feature, we can always use the ontology index to locate a set of webpages. The partial full-text index is then used to further reduce them into a subset of webpages for users. This design of separating ontology indices from a traditional full-text is interesting. Since we then know what ontology features are contained in a user query. Based on this information, we can apply OntoClassifier to analyze what domain concepts the user are really interested in and use the information to fast locate user interested webpages. Finally, we can employ the identified ontology features in a user query to properly rank the webpages for the user using the ranking method (Yang, Chuang, & Ho, 2007).

5. System evaluation

5.1. System architecture

We have developed an Ontology-supported Information Agent Shell (OntoAS) (Yang, Hsu, Chu, & Wu, 2008) based on the technologies described before, shown in Fig. 6. It contains the four main modules of information agents, including information crawling, information extracting, information classifying, and information presenting/ranking, and orderly corresponding to OntoCrawler, OntoExtractor, OntoClassifier, and OntoRecommender, respectively. The reason of the beginning word “Onto-” is all of module functions supported by ontology, which means the information agent shell is the core part of information agent separated from the domain ontology. The advantages of the approach are practically the shell application areas based on the domain ontology and extensible its application areas according to assimilation and fusion processing from other related domain ontologies. Ontology Database is the key component, which stores both domain ontology and query ontology. Ontological Database (OD) is a stored structure designed according to the ontology structure, serving as an ontology-directed canonical format for storing webpage information processed by OntoAS, which is separated from the shell. The approach not only can provide the basic operation semantic to the shell, but also can make the shell fast and precisely access information based on those semantic understanding. User Interface is responsible for singling out significant keywords from the user queries. Specifically, we use ontology, which defines the physical meanings of the important concepts involved in a given task, to successfully help the user interface develop a user profile to provide the personality functionality and store in User Profile Database through OntoRecommender.

5.2. System experiments

In this experiment, we compared our query technique with the most popular query engines Google and Yahoo. Here, we used Eqs. (1) and (2) to define Precision Rate, $R_p$, and Recall Rate, $R_e$, in which $NW_T$ meant the number of total returned webpages; $NW_C$ meant number of correct returned webpages; $NW_E$ meant number of related returned webpages but they were not necessarily the correct webpage. The results in Table 1 were after comparing returned webpage one after another through domain experts, we can get the $R_p$ and $R_e$ of Google were 1% and 50.0% while Yahoo were 1% and 33.3%, respectively.

\[
R_p = \frac{NW_C}{NW_T}
\]

\[
R_e = \frac{NW_C}{NW_C + NW_E}
\]
Fig. 7. Returned screen of OntoIAS.

Table 1
Comparison of the front 100 queries on Google and Yahoo.

<table>
<thead>
<tr>
<th></th>
<th>NWc</th>
<th>NWo</th>
<th>NWt</th>
<th>Rc (%)</th>
<th>Re (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Google</td>
<td>1</td>
<td>1</td>
<td>100</td>
<td>1</td>
<td>50.0</td>
</tr>
<tr>
<td>Yahoo</td>
<td>1</td>
<td>2</td>
<td>100</td>
<td>1</td>
<td>33.3</td>
</tr>
</tbody>
</table>

Table 2
Query result of the system.

<table>
<thead>
<tr>
<th>Threshold = 5</th>
<th>NWc</th>
<th>NWo</th>
<th>NWt</th>
<th>Rc (%)</th>
<th>Re (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>OntoIAS</td>
<td>22</td>
<td>0</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
</tbody>
</table>

When we used Google and Yahoo as base of webpage query, but on OntoIAS, we keyed in the same set of keywords. The returned screen of the system was shown in Fig. 7, and comparison results were shown in Table 2. After comparing Tables 1 and 2, the query precision and recall rate for Google researching engine through assistance of OntoIAS has up-rise around 99% and 50% (100% - 100% - 100%), while conditions of Yahoo were about 99% and 66.7% (100% - 100% - 33.3%). From the above comparison, it indicated that OntoIAS offered more precision and recall rate than Google and Yahoo on webpage searching; in addition, the technique we proposed and verified has its availability and system performance.

The information query meant the best recommendations have chosen from a group of related information sets. That wonderfully possessed different approaches to the same purpose as whether sampling specimens can be on behaving of degree of sampling body in huge amount of datum. In the sampling survey domain, the reliability was usually employed to measure the degree of precision of sampling system itself, while the validity was emphasized whether it can be correct to reflect the properties of the appearance of things (Wu, 1985). In other words, the former evaluates the stability of the measurement tool, while the latter focuses on the correctness of the tool itself. In 1979, Peter (1979) had the aid of mathematic model to represent the definitions of the reliability and validity, detailed as below.

5.2.1. Reliability

To assume a measurement tool measured the value Xo (generally is assumed to be the mean value), which can be divided into:

\[ X_o = X_i + X_e \]  \hspace{1cm} (3)

where \( X_o \) means observed \( X \), \( X_i \) means true \( X \), and \( X_e \) means error \( X \). The variance of the measured/observed value also is assumed to be the \( V_o \), which can be also divided into:

\[ V_o = V_i + V_e \]  \hspace{1cm} (4)

where \( V_o \) means observed \( V \), \( V_i \) means true \( V \), \( V_e \) means error \( V \). A reliability coefficient \( r_{II} \) therefore, is nothing more than to the ratio of true variance to observed variance:

\[ r_{II} = \frac{V_i}{V_o} \]  \hspace{1cm} (5)

Because \( V_i \) cannot be estimated directly from statistic view, therefore, Eq. (5) can be rewritten into a computational formula as:

\[ r_{II} = \left( \frac{V_i - V_e}{V_o} \right) = 1 - \left( \frac{V_e}{V_o} \right) \]  \hspace{1cm} (6)

5.2.2. Validity

If \( V_i \) can be divided into \( V_{io} \) plus \( V_{ip} \) again, then

\[ V_o = V_{io} + V_{ip} + V_e \]  \hspace{1cm} (7)

where \( V_{io} \) means correlated \( V \) which is the common variance related to measurement properties, \( V_{ip} \) means specific \( V \) which is the individual variance unrelated to measurement properties. The definition of validity \( V_{id} \) is:

\[ V_{id} = \frac{V_{io}}{V_o} \]  \hspace{1cm} (8)

The recommending significant information of this experiment was asserted by the domain experts, including observed values, true values, error values, and related variances. Table 3 were used Eq. (6) to calculate the reliabilities of some scholars’ recommending information on “Courses” and “Academic Activities” which are 0.856 and 0.756 in average, respectively. Table 4 were used Eq. (8) to calculate the validities of some scholars’ recommending information on “Courses” and “Academic Activities” which are 0.856 and 0.756 in average, respectively. In the literatures, the regular-level values of reliability and validity are 0.7 and 0.5, respectively, which verify and validate our results are high-level outcomes of information recommendation. Finally, what merits attention is: the Professional Classification of each scholar can be accurately shown that prove the system structure we proposed in this paper has its accuracy and availability.

Table 5 shows the comparison of user satisfaction of OntoIAS against other search engines. In the table, \( S_e \), for Satisfaction of testers, represents the average of satisfaction responses from 10 ordinary users, while \( S_s \), for Satisfaction of experts, represents that of satisfaction responses from 10 experts. Basically, each search engine receives 100 queries and returns the first 100 webpages for evaluation of satisfaction by both experts and non-experts. The table shows that the system prototype with these technologies described before, the last row, enjoys the highest satisfaction in almost classes. Due to the various webpage construction problems, our system prototype still cannot completely handle all type of webpages, for example the webpages with the frame technology, which explain why the lower user satisfaction is on the column K3.

6. Related works and comparison

Topical crawling was first introduced by Menczer (1997), which can automatically traverse Internet and retrieve webpages by hyperlinks. A focused crawler or topical crawler is a Web Crawler that attempts to download only webpages that are relevant to a pre-defined topic or set of topics, which was first introduced by Chakrabarti, van den Berg, and Dom (1999). In the face of the innumerable spam websites, traditional Web Crawler cannot function well to solve this problem (Dong, Hussain, & Chang, 2008a).
Nowadays, the research of crawlers moves closer to the semantic Web, which utilizes different semantic technologies to analyze the semantics of hyperlinks and web documents and aims to abstract metadata and significant information from them (Dong, Hussain, & Chang, 2008b). Examples are: Jung (2009) proposed a novel framework of open decision support system that is capable of gathering relevant knowledge from an open-network environment, which exploited the context-based focused crawler architecture to discover local knowledge from interlinked systems and the knowledge alignment process to integrate the discovered local knowledge; Huang, Lin, and Shi (2008) proposed a focused crawling framework supported by a statistical semantic association model with four kinds of semantic models and semantic information: thesauruses, categories, ontologies, and folksonomies to boost the crawling performance for relevant prediction and ranking.

Ontology is a technology for conceptualizing specific domain knowledge, which can provide machine-readable definitions to the domain. Therefore, ontology should be utilized to enhance the performance of focused crawlers by precisely defining the crawling boundary. Here are many examples of such crawler systems. Dong, Hussain, and Chang (2008c) exhibited a conceptual framework of an ontology-based focused crawler serving in the domain of transport services. Xing (2008) proposed a framework and algorithm of the ontology-based adaptive topical crawling which used the ontology technology to reduce the crawler to get the unrelated information for improving the correlativity of the topical crawler. Pahal, Chauhan, and Sharma (2007) presented an approach for document discovery building on a comprehensive framework for context-ontology driven focused crawling of web documents. Su, Gao, Yang, and Luo (2005) proposed an ontology-based intelligent crawling technique which uses a self-evolving mechanism that can dynamically adapt ontology to the particular structure of the relevant predicate. Juffinger, Neidhart, Granitzer, and Weichselbraun (2007) described a Web2.0 crawling system, which can deliver high quality, noise reduced accurate data found on Internet for ontology evolution.

Semantic Web technologies in general and ontology-based approaches in particular are considered the foundation for the next generation of information services. In this paper, we proposed the use of ontology-supported website models to provide a semantic level solution for a focused crawler named OntoCrawler so that it can provide fast, precise, and stable query results. The technique in this research has practically applied on Google and Yahoo searching engines to actively search for webpages of related information, and the experiment outcomes indicated that this technique could definitely up-rise precision rate and recall rate of webpage query. In addition, we reckon that the indicators for evaluating the crawler are more and more growing and mature. Their analysis and evaluation features, supported by the underlying statistic capability, however, provides yet another level of criticism in crawlers and deserves more attention, such as harvest rate, precision, recall, mean average precision, and fallout rate mentioned in Dong et al. (2008c). Finally, the web documents are well characterized by the hypertext, and the hypertext can be used to determine the relevance of the document to the search domain, such as

### Table 3
Results of the reliability of classification information.

<table>
<thead>
<tr>
<th>Domain professor</th>
<th>Courses</th>
<th>Academic activities</th>
<th>Professional classification</th>
</tr>
</thead>
<tbody>
<tr>
<td>C.S. Ho (何正儒)</td>
<td>V₁ 1 0.92 2</td>
<td>V₀ 7 0.71 71</td>
<td>Al</td>
</tr>
<tr>
<td>T.W. Kuo (郭大維)</td>
<td>V₁ 0 2 1</td>
<td>V₀ 0 1 1</td>
<td>Al</td>
</tr>
<tr>
<td>S.Y. Yang (楊騰遠)</td>
<td>V₁ 1 5 1</td>
<td>V₀ 0 1 1</td>
<td>Al</td>
</tr>
<tr>
<td>S.M. Chen (陳麗明)</td>
<td>V₁ 7 11 0.36 3</td>
<td>V₀ 3 7 0.57 57</td>
<td>Fuzzy</td>
</tr>
<tr>
<td>W.L. Hsu (許聞慶)</td>
<td>V₁ 0 1 1</td>
<td>V₀ 2 4 0.5 5</td>
<td>Al</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td><strong>0.856</strong></td>
<td><strong>0.756</strong></td>
<td></td>
</tr>
</tbody>
</table>

### Table 4
Results of the validity of classification information.

<table>
<thead>
<tr>
<th>Domain professor</th>
<th>Courses</th>
<th>Academic activities</th>
<th>Professional classification</th>
</tr>
</thead>
<tbody>
<tr>
<td>C.S. Ho (何正儒)</td>
<td>V₁ 11 12 0.92 5</td>
<td>V₀ 5 7 0.71 71</td>
<td>Al</td>
</tr>
<tr>
<td>T.W. Kuo (郭大維)</td>
<td>V₁ 2 2 1</td>
<td>V₀ 1 1 1</td>
<td>Al</td>
</tr>
<tr>
<td>S.Y. Yang (楊騰遠)</td>
<td>V₁ 5 5 1</td>
<td>V₀ 1 1 1</td>
<td>Al</td>
</tr>
<tr>
<td>S.M. Chen (陳麗明)</td>
<td>V₁ 4 11 0.36 4</td>
<td>V₀ 4 7 0.57 57</td>
<td>Fuzzy</td>
</tr>
<tr>
<td>W.L. Hsu (許聞慶)</td>
<td>V₁ 1 1 1</td>
<td>V₀ 2 4 0.5 5</td>
<td>Al</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td><strong>0.856</strong></td>
<td><strong>0.756</strong></td>
<td></td>
</tr>
</tbody>
</table>

### Table 5
User satisfaction evaluation.

<table>
<thead>
<tr>
<th>Method</th>
<th>K_WORD</th>
<th>K₁ (S₁/S₁)</th>
<th>K₂ (S₂/S₁)</th>
<th>K₃ (S₃/S₁)</th>
<th>Average (S₁/S₁)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alta Vista</td>
<td>60%/59%</td>
<td>74%/76%</td>
<td>28%/21%</td>
<td>54%/52%</td>
<td></td>
</tr>
<tr>
<td>Excite</td>
<td>63%/60%</td>
<td>78%/79%</td>
<td>48%/24%</td>
<td>63%/54%</td>
<td></td>
</tr>
<tr>
<td>Google</td>
<td>63%/62%</td>
<td>78%/78%</td>
<td>36%/21%</td>
<td>63%/54%</td>
<td></td>
</tr>
<tr>
<td>HotBot</td>
<td>66%/61%</td>
<td>75%/74%</td>
<td>60%/31%</td>
<td>67%/55%</td>
<td></td>
</tr>
<tr>
<td>InfoSeek</td>
<td>65%/68%</td>
<td>68%/68%</td>
<td>47%/28%</td>
<td>60%/55%</td>
<td></td>
</tr>
<tr>
<td>Lycos</td>
<td>61%/65%</td>
<td>74%/74%</td>
<td>34%/20%</td>
<td>56%/53%</td>
<td></td>
</tr>
<tr>
<td>Yahoo</td>
<td>64%/59%</td>
<td>74%/73%</td>
<td>36%/17%</td>
<td>58%/51%</td>
<td></td>
</tr>
<tr>
<td>OntoIAS</td>
<td>75%/67%</td>
<td>81%/76%</td>
<td>43%/32%</td>
<td>66%/58%</td>
<td></td>
</tr>
</tbody>
</table>

Nowadays, the research of crawlers moves closer to the semantic Web, which utilize different semantic technologies to analyze the semantics of hyperlinks and web documents and aim to abstract metadata and significant information from them (Dong, Hussain, & Chang, 2008b). Examples are: Jung (2009) proposed a novel framework of open decision support system that is capable of gathering relevant knowledge from an open-network environment, which exploited the context-based focused crawler architecture to discover local knowledge from interlinked systems and the knowledge alignment process to integrate the discovered local knowledge; Huang, Lin, and Shi (2008) proposed a focused crawling framework supported by a statistical semantic association model with four kinds of semantic models and semantic information: thesauruses, categories, ontologies, and folksonomies to boost the crawling performance for relevant prediction and ranking.

Ontology is a technology for conceptualizing specific domain knowledge, which can provide machine-readable definitions to the domain. Therefore, ontology should be utilized to enhance the performance of focused crawlers by precisely defining the crawling boundary. Here are many examples of such crawler systems. Dong, Hussain, and Chang (2008c) exhibited a conceptual framework of an ontology-based focused crawler serving in the domain of transport services. Xing (2008) proposed a framework and algorithm of the ontology-based adaptive topical crawling which used the ontology technology to reduce the crawler to get the unrelated information for improving the correlativity of the topical crawler. Pahal, Chauhan, and Sharma (2007) presented an approach for document discovery building on a comprehensive framework for context-ontology driven focused crawling of web documents. Su, Gao, Yang, and Luo (2005) proposed an ontology-based intelligent crawling technique which uses a self-evolving mechanism that can dynamically adapt ontology to the particular structure of the relevant predicate. Juffinger, Neidhart, Granitzer, and Weichselbraun (2007) described a Web2.0 crawling system, which can deliver high quality, noise reduced accurate data found on Internet for ontology evolution.
7. Conclusions

We have described how ontology-supported website models can effectively support Web search, which is different from website model content, construction, and application over our previous works (Yang & Ho, 2003). The technique in this research has practically applied on Google and Yahoo searching engines to actively search for webpages of related information, and the experiment outcomes indicated that this technique could definitely up-rise precision and recall rate of webpage query. A website model contains webpage profiles, each recording basic information, statistics information, and ontology information of a webpage. The ontology information is an annotation of how the webpage is interpreted by the domain ontology. The website model also contains a webpage profile that remembers how a website is related to the webpages and how it is interpreted by the domain ontology. We have developed OntoCrawler, which employs domain ontology-supported website models as the core technology to search for Web resources that are both user interested and domain oriented. Equipped with this technique, we also have developed an ontology-supported information agent shell in Scholar domain which manifests the following interesting features: ontology-supported construction of website models, website models-supported website model expansion, website models-supported webpage retrieval, high-level outcomes of information recommendation, and accordingly proved the feasibility of the related techniques proposed in this paper. In addition, our ontology construction is based on a set of pre-collected webpages on a specific domain; it is hard to evaluate how critical this collection process is to the nature of different domains. We are planning to employ the technique of automatic ontology evolution to help studying the robustness of our ontology. Finally, how to improve the webpage handling capability of our system is another course in the future, just like the techniques for processing free text information mentioned in Meng, Tseng, and Yang (2004), for handling all type of webpages and accordingly enjoys the highest user satisfaction.

Acknowledgement

The authors would like to thank Chung-Min Wang, Yai-Hui Chang, Ssu-Hsien Lu, Ting-An Chen, Chi-Feng Wu, Hsieh-I Lo, and You-Jen Chang for their assistance in system implementation and experiments. The partial work was supported by the National Science Council, Taiwan, ROC, under Grant NSC-98-2221-E-129-012.

References


Yang, S. Y., & Hsu, C. L. (2008). Ontology-supported focused-crawler for specified scholar's webpages. In Proceedings of IEEE the eighth international conference...

