A Semantic Taxonomy-Based Personalizable Meta-Search Agent

Larry Kerschberg[1], Wooju Kim[1], and Anthony Scime[2]

1) E-Center for E-Business, George Mason University
4400 University Drive, Fairfax, VA 22030, USA
kersch,wkim1@gmu.edu

2) Department of Computer Science, SUNY-Brockport
ascime@brockport.edu

Abstract

This paper addresses the problem of specifying, retrieving, filtering and rating Web searches so as to improve the relevance and quality of hits, based on the user's search intent and preferences. We present a methodology and architecture for an agent-based system, called WebSifter II, that captures the semantics of a user's decision-oriented search intent, transforms the semantic query into target queries for existing search engines, and then ranks the resulting page hits according to a user-specified weighted-rating scheme. Users create personalized search taxonomies via our Weighted Semantic-Taxonomy Tree. The terms in the tree can be refined by consulting a web taxonomy agent such as Wordnet. The concepts represented in the tree are then transformed into a collection of queries processed by existing search engines. Each returned page is rated according to user-specified preferences such as semantic relevance, syntactic relevance, categorical match, page popularity and authority/hub rating.

1. Introduction

With the advent of Internet and WWW, the amount of information available from the web grows exponentially every day. However, having too much information at one's fingertips does not always mean good quality information, and rather, it may often prevent a decision maker from making sound decisions, usually degrading the quality of decision. Helping decision makers to locate relevant information in an efficient manner is very important not only to a person but also to an organization in terms of time, cost, data quality and risk management.

Although search engines assist users in finding information, many of the results are irrelevant to the decision problem. This is due in part, to the keyword search approach, which does not capture the user's intent, what we call meta-knowledge. Search engines also have their own ranking system, which a user's criteria may change over time as more information about the problem is gathered. Thus, there is a "semantic gap" between the user's perception of the problem domain and the search results provided by search engines.

To overcome these two major problems, we proposed a semantic taxonomy-based personalizable meta-search agent approach. We build upon the ideas presented by Scime and Kerschberg [1]. We develop a tree-structured search intent representation scheme with which users describe their search intent. We call this representation scheme the "Weighted Semantic Taxonomy Tree (WSTT)", in which each node denotes a concept that pertains to the user's problem-domain. To address the second weakness, we present an elaborate user preference representation scheme based on various components, each of which represents a specific decision-criterion. Users can easily and precisely express their preference in a search using this representation scheme.

In order to rate the relevance of a page hit, we use a rating mechanism combining the WSTT and the component-based preference representation. Since web page rating can itself be viewed as a decision-making problem, where a decision maker (a user) must evaluate various alternatives (web pages) for his/her problem (user's web search intention), we use decision-analytic methods in the design of our rating mechanism.

Finally, we have designed and are presently implementing a meta-search agent called WebSifter II that cooperates with Wordnet for concept retrieval, and most well-known search engines. For the empirical validation of our approach, we also present some real world examples of our system.

The remainder of the paper is organized as the follows. Section 2 explains related previous issues and related research. Section 3 presents the major aspects of our semantic-based personalizable approach to the representation of user intention, and the multi-component rating of search hits. In Section 4, we discuss the system architecture of WebSifter II, the search agent that implements our methodology. We deal with some collaboration issues too in this section. The results of empirical studies are presented in Section 5.

2. Related Work

Most of current internet search engines such as Yahoo, Excite, Altavista, WebCrawler, Lycos, Google, etc. suffer from *Recall* and *Precision* problems [2]. The relatively low coverage of individual search engines leads to the use the concept of meta-search engines to improve the recall of a query. Examples are MetaCrawler [3], SavvySearch [4], NECI Metasearch Engine [5], and Copernic (http://www.copernic.com). This meta-search engine approach partly addresses the recall problem but still suffers from the precision problem.

We can categorize research regarding the precision problem into three major themes: content-based, collaborative, and domain-knowledge approaches.

The content-based approach first represents a user's explicit preferences and then evaluates web page relevance in terms of its content and user preferences. Syskill & Webert [6], WebWatcher [7], WAWA [8], and WebSail [9] fall into this category. Further, some research takes into account not only web page content but also its structure (e.g. hyperlinks) to evaluate relevance [10, 11].

The collaborative approach determines information relevancy based on similarity among users rather than similarity of information itself. Example systems are Firefly and Ringo [12], Phoaks [13], and Siteseer [14]. In addition, some hybrid approaches incorporate approaches for example Fab [15], Lifestyle Finder [16], WebCobra [17].

The third category is the domain knowledge approach that uses user and organizational domain knowledge to improve the relevancy of search results. One of the popular domain knowledge approaches provides a predefined taxonomy path, e.g., Yahoo!. So, classifying web pages automatically into a pre-defined, or a dynamically created taxonomy [18] is a related issue to this approach. NorthernLight (www.northernlight.com) is a search engine that supports this kind of dynamic taxonomy service. Using NorthernLight's *Custom Search Folder* service, users can refine their search query to a specific domain, when the search engine presents too much information.

Some research incorporates users domain knowledge in a more explicit way. For example, Aridor et al. [19] represent user domain knowledge as a small set of example web pages provided by users. Chakrabarti et al. adopted both a pre-defined (but modifiable) taxonomy and a set of example web pages provided by users as domain knowledge [20].

From this survey of related research, we have identified several aspects that merit further consideration. First, most approaches force users to use a search engine in a passive rather than active manner. Often, the user cannot understand why extraneous and irrelevant results are retrieved. There is a pressing need for users to control search by specifying their intent. Second, current approaches lack sufficient expressive power to capture a users' search intent and preferences, because most of the representation schemes are based on a vector space model [21] or its variants. Third, most approaches do not take full advantage of domain-specific knowledge with which to scope the search, interpret, and classify the query result.

Regarding the first limitation, there is another related research category, the ontology-based approach by which users can express their search intent in a more semantic fashion. Domain-specific ontologies are being developed for commercial and public purposes [22] and OntoSeek [23], On2Broker [24], GETESS [25], and WebKB [26] are example systems.

Although the ontology-based approach is a promising way to solve some aspects of the precision problem, it still requires two major pre-requisites. First, the entire collection of web pages must be transformed into ontological form. Second, there is as yet no common agreement on the representation of the ontology, nor the query or reasoning mechanisms. Even if these two prerequisites are satisfied, the precision problem in web search will remain due to the huge amount of the information on the web. That is, a user-centric information relevancy evaluation scheme will complement the above approaches.

3. Semantic Taxonomy-Tree-Based Approach for Personalized Information Retrieval

3.1 Weighted Semantic Taxonomy Tree

Usually a keyword-based search representation is insufficient to express a user's search intent. By postulating a user's decision-making process as depicted in Figure 1, we can support readily query formulation and search.

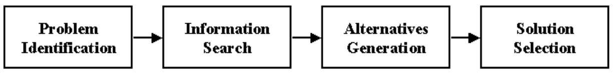


Figure 1 Four Phases of Decision Making Process

This process starts with a problem identification phase and then a user seeks relevant information to solve the identified problem. Based on the collected information, listing alternatives, evaluating them, and selecting a solution are the following steps. One implication of the decision-making process is that the more we understand a user's problems, the better we can support a user's information search. In our approach, we represent a user's search intent by a hierarchical concept tree with weights associated with each concept, thereby reflecting user-perceived relevance of concepts to the search.

Let's assume that a person has started a new business and is looking for office equipment. He wants to search for information about office equipment on the web. Suppose he wants information about chairs, so he might build a query using a single term, "chair". If he is a more skilled user of internet search engines, he might build a query using two terms, "office" and "chair" to obtain more precise results. He may also use 'AND' or 'OR' operator between them. In this case, the term "office" provides added context for the search. However, this formulation is still very implicit and passive. As we mentioned earlier, one way to express this kind of context information is by using a taxonomy tree as shown in Figure 2. Figure 2(a) shows a simple taxonomy tree that represents a search intention to find a chair in the context of office, while a search for finding an office in the context of chair is expressed by Figure 2(b). The taxonomy tree provides more expressive semantics than simple keyword-based representations used by most current search engines.

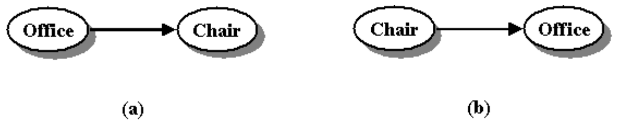


Figure 2 A Simple Example of Taxonomy Tree

The taxonomy tree approach is already used in many of search engines such as Yahoo! We have devised a tree-based search representation model that allows users to present their search intention by defining their own taxonomy topology. We call this the *Weighted Semantic Taxonomy Tree* (WSTT) model. Now, let us formally define this model. The WSTT consists of a set of nodes that is denoted as *N* in the sequel. Because it is a tree, all nodes, except the root node, must have one parent node. Every node should have one representative term and a weight that represents the importance of this node for a search. For a node $n \in N$, we denote a representative term, or label, and its weight as $rt(n)$ and $w(n)$, respectively. We restrict the feasible range of the value of $w(n)$ from 0 to 10. Figure 3 shows a realistic example of the businessman's search intention using our WSTT scheme. Users can build their own hierarchical taxonomy tree, and assign importance levels to each term within the context of their antecedent terms. For example, we can translate the upper sub-tree as that a businessman wants to find information about chairs, desks, and phones within the context of office furniture and office equipment where the numbers that appear to the left to each term, 10, 9, and 6 denote the respective importance levels of chairs, desks, and phones.

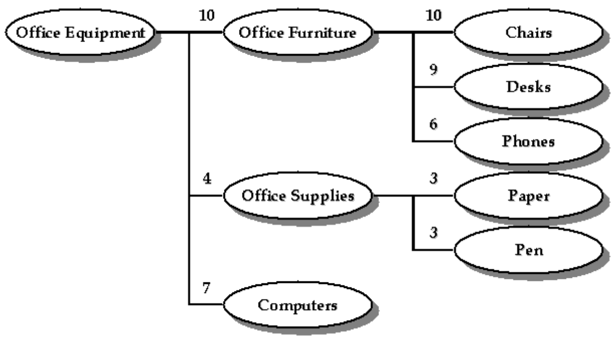


Figure 3 An Example of WSTT that represent a businessman's search intention

One drawback is that the terms may have multiple meanings, and this is one of the major reasons that search engines return irrelevant search results. To address this limitation in using just simple terms, we introduce the notion of "word senses" from Wordnet [27] into our WSTT scheme to allow users to refine their search intention. Wordnet is a linguistic database that uses sets of terms that have same semantics (*synsets*) to represent word senses. Each synset corresponds to a specific meaning in English and so each word may be associated with multiple synsets. In this paper, we rename this synset as *Concept* for our own use and the user can choose one of the concepts available from Wordnet for the term of a specific node in WSTT. We denote an available concept, that is, a set of terms for a node *n* as $c(n)$. For example, the "chair" term has the following four possible concepts from Wordnet.

(1) {chair} // a seat for one person, with a support for the back,
(2) {professorship, chair} // the position of professor, or a chaired professorship,
(3) {president, chairman, chairwoman, chair, chairperson} // the officer who presides at the meetings of an organization, and
(4) {electric chair, chair, death chair, hot seat} // an instrument of death by electrocution that resembles a chair.

If the user wants to search for a chair to sit on, he would choose the first concept. If the user selects the first concept, then without loss of generality, we can assume that the remaining concepts are not of interest, thereby obtaining both positive and negative indicators of his intent. Now, let's distinguish the set of terms of selected concept from the set of terms of the unselected concepts as *Positive Concept Terms* and *Negative Concept Terms*, and denote them as $pct(n)$ and $nct(n)$ for a node *n*, respectively. If we denote a term as *t* and assume that a user selects the *k*-th concept, then we can formalize the definitions of them for a given node *n* as follows:

$$pct(n) = \{t \mid t \in c_k(n)\} \qquad (1)$$

$$nct(n) = \left\{ t \,\middle|\, t \in \bigcup_{i \neq k} c_i(n) \right\} - \{rt(n)\} \qquad (2)$$

where $c_i(n)$ denotes the i-th concept available from Wordnet for a node n and
$rt(n)$ denotes the representative term of n.

If a user selects the second concept from our example, according to the definitions from (1) and (2), $pct(n)$ and $nct(n)$ are as follows: $pct(n)$ = {professorship, chair} and $nct(n)$ = {president, chairman, chairwoman, chairperson, electric chair, death chair, hot seat}.

Figure 4 shows an internal representation of the user's intention via the WSTT schema, after the concept selection process has finished; the user however sees the tree of Figure 3. Another advantage using the tree structure is that it is possible to represent many concepts at the same time. This allows the user to specify a broad range of interests simultaneously.

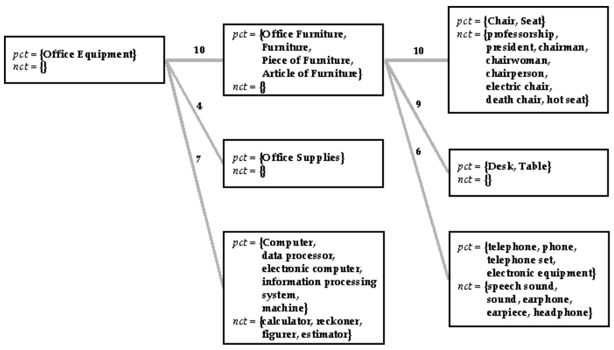

pct = {Office Equipment}
nct = {}
10
4
7
pct = {Office Furniture, Furniture, Piece of Furniture, Article of Furniture}
nct = {}
pct = {Office Supplies}
nct = {}
pct = {Computer, data processor, electronic computer, information processing system, machine}
nct = {calculator, reckoner, figurer, estimator}
10
9
6
pct = {Chair, Seat}
nct = {professorship, president, chairman, chairwoman, chairperson, electric chair, death chair, hot seat}
pct = {Desk, Table}
nct = {}
pct = {telephone, phone, telephone set, electronic equipment}
nct = {speech sound, sound, earphone, earpiece, headphone}

Figure 4 An Example of Internal Representation of User's Search Intention

3.2 Multi-Attribute-Based Search Preference Representation

The ranking of web search hits by users involves the evaluation of multiple attributes, which reflect user preferences and their conception of the decision problem. In our approach, we pose the ranking problem as a multi-attribute decision problem. Thus, we examine the search results provided by multiple search engines, and rank the pages, according to multiple decision criteria. Both Multi-Attribute Utility Technology (MAUT) [28] and Repertory Grid [29] are two major approaches that address our information evaluation problem. Our ranking approach combines MAUT and the Repertory Grid. We define six search evaluation components as follows:

(1) *Semantic* component: represents a web page's relevance with respect to its content.
(2) *Syntactic* component: represents the syntactic relevance with respect to its URL. This considers URL structure, the location of the document, the type of information provider, and the page type (e.g., home, directory, and content).
(3) *Categorical Match* component: represents the similarity measure between the structure of user-created taxonomy and the category information provided by search engines for the retrieved web pages.
(4) *Search Engine* component: represents the user's biases toward and confidence in search engine's results.
(5) *Authority/Hub* component: represents the level of user preference for *Authority* or *Hub* sites and pages [30].
(6) *Popularity* component: represents the user's preference for popular sites. Popularity can be measured by the number of visitors or the number of requests for the specific page or site.

Further, in this multi-component-based preference representation scheme, the user can assign a preference level to each of these components, and also to each available search engine within the search engine component. Then, these components and the assigned preference level are eventually synthesized into a single unified value resulting in the relevance measure for a specific web page. Figure 5 conceptually depicts our scheme. In this figure, each number assigned to an edge denotes user's preference level for that component. This multi-component preference scheme allows users more control over their searches and the determination of a page's relevance.

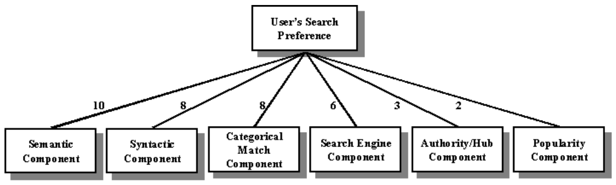


Figure 5 A Conceptual Model of User's Preference Representation Scheme

Thus far, we have discussed how to capture and represent semantically the user's search intention and search preferences. Now, we turn our attention to deriving a good estimate of the relevancy of a web page based on these semantics. In the following sections, we will first discuss how to obtain web information using existing search engines and then address the derivation of relevance estimates.

3.3 Gathering Web Information based on Search Intention

Since we adopt a meta-search approach to web information gathering to preserve the benefits of meta-search engines discussed in [3, 4, 19], we neither create nor maintain our own index database of web information. At present, there is no search engine that accepts a search request based on the WSTT. We have developed a translation mechanism from our WSTT-based query, to Boolean queries that most of current search engines can process.

As already mentioned, we represent a user's search intention as a tree, as shown in Figure 4. The leaf nodes denote the terms of interest to the user, and the antecedent nodes for each node form a search context. We transform the entire tree into a set of separate queries where each is acceptable to existing search engines. To do this, first we decompose the tree into a set of paths from the root to each leaf node. Then for each path, we generate all possible combinations of terms, when selecting one term from the positive concept terms of each node in the path from a root node to a leaf node. Finally, we pose each query to search engines to obtain query results.

We now provide definitions to formalize the above discussion. Let's first define a *Leaf Path* as an ordered set of nodes, $\{n_0, n_1, n_2, \ldots, n_{l-1}, n_l\}$, where n_0 is a root node, n_l is a leaf node, and $n_1, n_2, \ldots, n_{l-1}$ are consecutive intermediate nodes on the path from n_0 to n_l in the WSTT. We denote a leaf path as *lp*. We also define a set of all distinct leaf paths available from the WSTT as *lpset*. For example, we have six leaf paths from the example WSTT as in the Figure 3 and its *lpset* becomes {{Office Equipment, Office Furniture, Chairs}, {Office Equipment, Office Furniture, Desks}, {Office Equipment, Office Furniture, Phones}, {Office Equipment, Office Supplies, Paper}, {Office Equipment, Office Supplies, Pen}, {Office Equipment, Computers}}. Now, let's define a *Term Combination Set* for a *Leaf Path lp*, as a set of all possible combinations of terms by selecting one term from each *pct*(*n*), where $n \in lp$ and denote it as *tcslp*(*lp*). We also denote a set of all term combinations available from a given WSTT and each of its elements as *tcs* and *tc*, respectively. Then, using the above definitions, a *tcslp*(*lp*) and *tcs* can be formally represented respectively as follows:

$$tcslp(lp) = pct(n_0) \times pct(n_1) \times pct(n_2) \times \ldots \times pct(n_l) \qquad (3)$$

where symbol × denotes the Cartesian product of sets.

$$tcs = \bigcup_{lp \in lpset} tcslp(lp) \qquad (4)$$

If *lp* is the first element, that is, {Office Equipment, Office Furniture, Chairs} of the *lpset* in the case of Figure 3 and Figure 4, then according to equation (3), *tcslp*(*lp*) = {{Office Equipment, Office Furniture, Chair}, {Office Equipment, Office Furniture, Seat}, {Office Equipment, Furniture, Chair}, {Office Equipment, Furniture, Seat}, {Office Equipment, Piece of Furniture, Chair}, {Office Equipment, Piece of Furniture, Seat}, {Office Equipment, Article of Furniture, Chair}, {Office Equipment, Article of Furniture, Seat}}.

Once we get *tcs*, then we make each term combination, $tc \in tcs$ as a separate request and pose them to each search engine for web information gathering. Now, the problem is how to generate actual query statements to each query engine based on each *tc*. We have trade-offs between *Precision* and *Coverage* depending on which logical operators we impose between terms. Actually, each *tc* is a set of terms and so, it can be represented as $\{t_1, t_2, \ldots, t_n\}$ where $t_1, t_2, \ldots, t_n \in tc$. To generate an actual query statement from a *tc*, we can have two different alternative choices, "$t_1 \wedge t_2 \wedge \ldots \wedge t_n$" and "$t_1 \vee t_2 \vee \ldots \vee t_n$" where $\wedge$ denotes AND and $\vee$ denotes OR. The first one provides more precise search results, while the second allows greater coverage.

Based on the fact that a general user tends to use the AND operator between terms when considering additional terms for the context of a search, we adopt the AND operator in generating actual query statements. We leave the more general scheme for future research. For the illustration of our query generation method, let's use the case depicted in Figure 4 again. According to the procedures mentioned thus far, the upper-most leaf path of the WSTT in Figure 4 is translated into eight separate query statements as follow. (1) "Office Equipment" AND "Office Furniture" AND "Chair", (2) "Office Equipment" AND "Office Furniture" AND "Seat", (3) "Office Equipment" AND "Furniture" AND "Chair", (4) "Office Equipment" AND "Furniture" AND "Seat", (5) "Office Equipment" AND "Piece of Furniture" AND "Chair", (6) "Office Equipment" AND "Piece of Furniture" AND "Seat", (7) "Office Equipment" AND "Article of Furniture" AND "Chair", and (8) "Office Equipment" AND "Article of Furniture" AND "Seat".

These queries can now be submitted to each target search engine, and the query results are stored for further processing, as discussed in the next section.

3.4 Unified Web Information Rating Mechanism

In this section, we discuss a rating mechanism to evaluate each resulting page hit from the target search engines for the generated query statements. Through this mechanism, each web page will have its own value representing the relevance level from the user's viewpoint. To accomplish this goal, six relevance values of a web page are computed, corresponding to each of the six components. Then a composite value of these six relevance values is computed based on a function of the multi-attribute-based search preference representation scheme. In the following sub-sections, we will first discuss how this composite relevance value is computed, and then a set of methods to compute each of component's relevance values.

3.4.1 Composite Relevance Value Computation

Let's first assume we evaluated six components' relevance values for a web page retrieved from search engines. Then we need to synthesize these six values into one single composite relevance value to compare web pages to each other and to list them to the user in an order of relevance. This problem can be viewed as a multi-attribute decision-making problem.

One of the popularly accepted approaches in decision science community is AHP (Analytic Hierarchy Process) [31] and it converts user's subjective assessments of relative importance between preference components into a linear set of weights, which is further used to rank alternatives. Although we adopt AHP approach as a basis of our synthesizing mechanism, we have modified the original AHP to fit to our weight acquisition scheme, because it requires pair-wise comparisons between all components to obtain importance ratios between each pair of them. Actually in our approach, a user assigns an absolute importance weight on each component rather than relative ratios between components. However, since we still need those relative ratios, we first approximate them by dividing absolute importance weights of components by each other. Then, we follow the same remaining steps of AHP to compute the composite relevance value for each web page.

We now provide notations to formalize the above discussion as follows.

compset :	denotes a set of preference components to be considered in our scheme.
$cw^U(x)$:	denotes a weight provided by the user to represent the importance of a component x.
$rvc(x, pg)$:	denotes a relevance value of a web page pg with respect to a component x.
$lr(x, y)$:	denotes a relative importance ratio of component x compared to component y.
$ns(z)$:	denotes a function that returns the number of elements in a set z.

We first approximate $lr(x, y)$ by the (5) based on the user-provided importance weights for each pair of components:

$$lr(x, y) = cw^U(x) / cw^U(y) \qquad (5)$$

where $x \in compset$ and $y \in compset$.

Then, the AHP computes normalized importance weights for each component based on these relative ratios. We denote the normalized importance weight for a component *com* and the composite relevance value of a web page *pg* as $cw^N(com)$ and $rv(pg)$, respectively. According to AHP, these two values can be calculated respectively as follows:

$$cw^N(com) = \left[\sum_x \left(\frac{lr(com, x)}{\sum_y lr(x, y)} \right) \right] \Big/ ns(compset) \qquad (6)$$

where $x \in compset$ and $y \in compset$.

$$rv(pg) = \sum_x cw^N(x) \cdot rvc(x, pg) \qquad (7)$$

where $x \in compset$.

Finally, web pages are presented to users in descending order of *rv*. This, together with the page relevancy value indicates the relative importance of that page to the user.

Thus far, we have discussed how to synthesize the relevance values of a user's preference components into a single composite value, under the assumption that these relevance values of the components have already been computed. Now, we show how to compute relevance values of each of the six preference components based on the user's preference, as well as the user's search intent as represented by the WSTT.

3.4.2 Semantic Component Relevancy Computation

The semantic component represents relevancy of a web page to a user's search intent represented by the WSTT with respect to its content. To compute this relevance, we conceptually follow the reverse steps that we performed in the section 3.3 to generate separate queries from the WSTT.

First, we evaluate the semantic relevancies of a retrieved web page for each of term combinations; we then combine the semantic measures for each leaf path; and then we bind each of these semantic measures to the corresponding leaf node; and finally we compute a semantic component relevancy of the web page using an AHP-based WSTT relevance value composition mechanism that propagates the bound values on the leaf nodes toward the root node, thereby providing a single combined relevance value at the root node.

Now, let's explain the details of this procedure in a formal manner. We first define $rvtc^{SM}(tc, pg)$ as a semantic relevance value of a web page *pg* to a term combination *tc* and it is computed by a simple counting method as follows:

$$rvtc^{SM}(tc, pg) = \frac{\sum_{t \in tc} appear(t, pg)}{ns(tc)} \qquad (8)$$

where t is a term and the function $appear(t, pg)$ returns 1 if t appears in pg and 0, otherwise.

Based on these $rvtc^{SM}$ values, we define $rvlp^{SM}(lp, pg)$ as a semantic relevance value of a web page *pg* to a leaf path *lp*. When we compute this $rvlp^{SM}$, we have to consider two aspects. First, we have to synthesize multiple $rvtc^{SM}$ values obtained from equation (8) for a leaf path into a single measure and we adopt a max function for this. Second, we have to consider negative concepts related to a leaf path. To incorporate these negative concepts into computing $rvlp^{SM}$, we first develop a measure to evaluate irrelevancy of a web page *pg* in terms of negative concept terms related to a leaf path *lp* and we denote it as $irv(lp, pg)$. The following equation (9) shows its mathematical definition.

$$irv(lp, pg) = \sum_t appear(t, pg) \qquad (9)$$

where t is a term and also $t \in \bigcup_{n \in lp} nct(n)$.

Now, we can compute $rvlp^{SM}$ using the following equation (10).

$$rvlp^{SM}(lp, pg) = \left(\max_{tc} rvtc^{SM}(tc, pg)\right) \cdot (1 - \theta)^{irv(lp, pg)} \qquad (10)$$

where $tc \in tcslp(lp)$ and θ is a given [0, 1] scale degradation rate.

In equation (10), θ denotes the level of degradation with respect to the irrelevance caused by negative concepts. So if θ is close to 1, then a little irrelevancy results in a big impact on $rvlp^{SM}$. On the other hand, if it is close to 0, the irrelevancy does not have any impact on the *rvp* value. The user can control this rate and we set it to a default of 0.1.

Now, we synthesize a single semantic relevancy value of a web page according to the WSTT. Since AHP was originally developed to derive a unified measure to evaluate decision alternatives based on a tree like WSTT, we easily apply this approach to our WSTT scheme by combining our $rvlp^{SM}$ values for each leaf path into a single semantic relevance value of a web page. However, we need to normalize the user-provided weights for the nodes of WSTT, for reason similar to those discussed in the previous section. For this normalization, we apply equation (5) to each hierarchical branch of the WSTT, and we obtain a set of normalized weights for each node within the scope of the branch to which the nodes belong. We denote this normalized weight for a node *n*, $w^N(n)$. With the normalized weights, let's formalize the AHP-based WSTT relevance value composition mechanism. Equation (11) shows a relevance value determination rule on each node of WSTT for a web page *pg* and we denote a relevance value of a web page *pg* on a node *n* as *rvn*(*n*, *pg*).

$$rvn(n, pg) = \begin{cases} bndfn(lp, pg) & \text{if } n \text{ is a leaf node of a leaf path } lp. \\ \sum_{x \in children(n)} w^N(x) \cdot rvn(x, pg) & \text{otherwise.} \end{cases} \quad (11)$$

where *children*(*n*) is a set of nodes that is a child of *n* and *bndfn*(*lp*, *pg*) is an arbitrary value binding function to leaf nodes.

To perform this mechanism, we first bind relevance values from *bndfn*() to all corresponding leaf nodes and then these values are propagated from leaf nodes to the root node, finally obtaining a single composite relevance value of a web page for the WSTT. In this semantic component case, by setting *bndfn*(*lp*, *pg*) as $rvlp^{SM}(lp, pg)$ in the equation (10), we can obtain a single composite semantic relevance value of a web page *pg* as $rvn(n_0, pg)$, where n_0 is the root node of the WSTT. This obtained value is then assigned to *rvc*(Semantic Component, *pg*) for further computing of composite relevance value with other preference components, discussed in the previous section.

Figure 6 shows conceptually the entire flow of computations from relevancy computing for a term combination to relevancy computing across the WSTT, which is required to compute a semantic relevance value of a web page. In this figure, $PageSet(tc_i, s_j)$ denotes a set of resulting pages from a search engine s_j for a term combination tc_i. Actually, we will use a similar method when computing categorical match and search engine components' relevancies in the following sections.

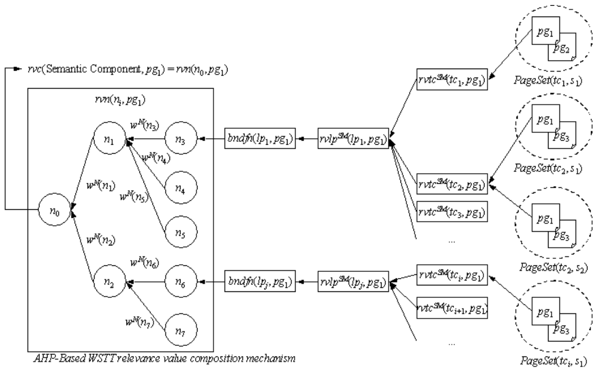


Figure 6 Conceptual Flow of Computation of Semantic Component Relevancy

3.4.3 Syntactic Component Relevancy Computation

The syntactic component of web document measures the structural aspects of the page as a function of the role of that page with the structure of a web site. Our approach takes into account the location of the document, its role (e.g., home, directory, and content), and the well formedness of its URL.

We define three types of web pages:

- *Direct-Hit* – the page may be a home page or a page with significant content relevant to the search.
- *Directory-Hit* – this page has links to other pages relevant to the search.
- *Page-Hit* – web pages that are subordinate to direct-hit and directory-hit pages fall into this category. These pages contain partial information related to the search.

Scime and Kerschberg [1] define a set of heuristics to classify a web page returned from a search engine as either a direct, directory, or page hit. Further, a page may have more than one classification. In order to manipulate syntactic relevancy, we assign a numeric value to each type as a real number in the interval [0, 1]. Default values for direct, directory, and page hits are 1.0, 0.6, and 0.4 respectively. The assumption is that users would prefer to view direct hits over the other two.

Since a web page might be classified into more than one class, we need to synthesize those multiple matches into one measure. To do this, we introduce an averaging mechanism and define some necessary notations and a formula to compute the syntactic relevance value of a web page *pg*, *rvc*(Syntatic Component, *pg*) as follows:

rset(*cl*)	:	denote a set of rules to classify a web page into the class *cl*.
rsc(*r*)	:	denotes a score of a rule *r* and it returns 1.0 if $r \in rset$(Direct Hit), 0.6 if $r \in rset$(Directory Hit), and 0.4 if $r \in rset$(Page Hit).
mat(*r*, *pg*)	:	denotes a function that returns 1 if a rule *r* is matched to a web page *pg* and 0, otherwise.

rvc(Syntatic Component, *pg*) =

$$\left(\sum_r rsc(r)\cdot mat(r,pg)\right)\Big/\sum_r mat(r,pg) \qquad (12)$$

3.4.4 Categorical Match Component Relevancy Computation

Categorical Match component represents the similarity measure between the structure of user-created taxonomy and the category information provided by search engines for the retrieved web pages. Nowadays, many popular search engines respond to the users query not only with a list of URLs for web pages but also with their own categorical information for each web page. For example, the followings are some portion of search results provided by Lycos for the query "chair".

```
(1) Donald B. Brown Research Chair on Obesity
    Health > Mental Health > Disorders
    > Eating Disorders > Obesity
(2) Steel Chair Wrestling
    Sports > Fantasy > Pro Wrestling
    …
(3) Chair Technologies
    Business > Industries > Manufacturing
    > Consumer Products > Furniture
    > Seating > Office Chairs
    …
```

In the above search results, the left hand side numbers are ranks of the corresponding web pages and the associated lines below each title show the related category information of web pages. Although different search engines associate different category information to the same web page, such categorical information helps users filter out some of the returned search results without actually visiting the URL. Actually, the categorical match component is designed to provide the benefits of manual filtering by automatic means; this is accomplished by comparing the WWST terms with the categorical information provided by search engines. This is one of the major contributions of this paper.

Now, let's discuss how to measure the relevancy between the WSTT and the categorical information in more detail. We first represent the category information for a web page *pg* from a search engine *s*, as an ordered set of category terms in a form like $\{cat_1, cat_2, \ldots, cat_m\}$, where cat_i is the *i*-th category term and *m* is total number of category terms in the set and we denote it *catinfo*(*pg*, *s*). For example, *catinfo*(Chair Technologies, Lycos) in the above case, can be represented as the ordered set of category terms, {Business, Industries, Manufacturing, Consumer Products, Furniture, Seating, Office Chairs}. However, since it is hard to directly compare such *catinfo* to the entire WSTT, here we adopt a similar approach applied to Semantic Component case, where we first measure the relevance of a *catinfo* to a single term combination, and then, combine them up to a single composite measure with respect to the entire WSTT.

So now, the relevance between a *catinfo* and a term combination *tc* can be measured from two different aspects, co-occurrence of terms and order consistency of terms. To measure the co-occurrence, we devised a following formula (13).

coccur(*tc*, *catinfo*) =

$$\left(\frac{\sum_t member(t, catinfo)}{ns(tc)}\right)\cdot\left(\frac{\sum_{cat} member(cat, tc)}{ns(catinfo)}\right) \qquad (13)$$

where $t \in tc$ is a term, $cat \in catinfo$ is a category term, and *member*(*x*, *y*) is a function that returns 1 if *x* is a member of *y* and 0, otherwise.

To consider the order consistency, let's first denote the precedence relationship of two arbitrary terms, t_l and t_r as (t_l, t_r), and that means t_l precedes t_r in an ordered terms set. We also define a set of all available precedence relationships from an ordered set of terms *x*, as *prelset*(*x*). Then we measure the consistency of *catinfo* with respect to a precedence relationship, (t_l, t_r) as follows:

$cons((t_l, t_r), catinfo)$ =

$$\begin{cases} 1 & \text{if } t_l, t_r \in catinfo \text{ and } t_l \text{ precedes } t_r \text{ in } catinfo. \\ 0 & \text{otherwise} \end{cases} \qquad (14)$$

Now, let's define a consistency of a category information *catinfo* to a term combination *tc* as *constc*(*tc*, *catinfo*) and the equation (15) shows how to compute it. Because we want to focus only on order consistency between *catinfo* and *tc* not depending on co-occurrence between them, we additionally define an ordered intersection set of *tc* and *catinfo*, where order of its element terms follows *tc*, as *isset*(*tc*, *catinfo*) and then we can remove co-occurrence effect by only considering the precedence relationships in that set.

constc(*tc*, *catinfo*) =

$$\sum_{pr} cons(pr, catinfo)\Big/\binom{ns(isset(tc, catinfo))}{2} \qquad (15)$$

where $pr \in prelset(isset(tc, catinfo))$, is a precedence relationship.

For example, let a term combination *tc* be {*a*, *b*, *c*, *d*, *e*} and a category information *catinfo* be {*a*, *e*, *c*, *f*}. Then *isset*(*tc*, *catinfo*) becomes {*a*, *c*, *e*} and also *prelset*(*isset*(*tc*, *catinfo*)) becomes {(*a*, *c*), (*a*, *e*), (*c*, *e*)}. According to the formula (14), *cons*((*a*, *c*), *catinfo*), *cons*((*a*, *e*), *catinfo*), and *cons*((*c*, *e*), *catinfo*) have their value as 1, 1, and 0, respectively. Since *ns*(*isset*(*tc*, *catinfo*)) is 3 in this case, the denominator of the equation (15) becomes 3, and finally *constc*(*tc*, *catinfo*) becomes (1+1+0)/3 = 2/3. Also in this case, *coccur*(*tc*, *catinfo*) becomes 3/5×3/4=9/20, because 3 of 5 terms of *tc* appear in *catinfo* and 3 of 4 terms of *catinfo* appear in *tc*.

To synthesize both the above aspects of categorical match between a term combination and a category information, we define the following measure, *rvtcc*(*tc*, *catinfo*).

rvtcc(*tc*, *catinfo*) =

$$\alpha \cdot coccur(tc, catinfo) + (1-\alpha) \cdot constc(tc, catinfo) \qquad (16)$$

where α is a [0, 1] scale factor to represent the relative importance of co-occurrence to order consistency.

Actually since a web page can have several category labels from different search engines for a given term combination, we need to further synthesize to obtain a single categorical match relevance value of a web page *pg* for a term combination *tc*, $rvtc^{CM}(tc, pg)$ and it is formalized in (17).

$$rvtc^{CM}(tc, pg) = \sum_s sw(s) \cdot rvtcc(tc, catinfo(pg, s)) \qquad (17)$$

where *s* is a search engine and *sw*(*s*) is a normalized preference weight for the search engine *s*.

As in the case of Semantic Component, we adopt the max function to synthesize $rvtc^{CM}$s to obtain a categorical match relevance value of a web page *pg* for a leaf path *lp*, $rvlp^{CM}(lp, pg)$ as follows:

$$rvlp^{CM}(lp, pg) = \max_{tc \in tcslp(lp)} rvtc^{CM}(tc, pg) \qquad (18)$$

We also can obtain a single composite categorical match relevance value of a web page *pg*, *rvc*(Categorical Match Component, *pg*) using the AHP-based WSTT relevance value composition mechanism that is formalized in (11). To do this, we first set *bndfn*(*lp*, *pg*) in the equation (11) as $rvlp^{CM}(lp, pg)$, then we propagate values from leaf nodes to the root node. At the root node n_0, we obtain a single composite categorical match relevance value of a web page *pg* as $rvn(n_0, pg)$ and we finally assign this value to *rvc*(Categorical Match Component, *pg*), which will be used to obtain a composite relevance value with other preference components.

3.4.5 Search Engine Component Relevancy Computation

The Search Engine component represents the user's biases toward and confidence in a search engine's results. To measure this search engine component, let's first define a basic unit information, that is, rank of a web page *pg* by search engine *s* for the request from term combination *tc* as *rank*(*tc*, *pg*, *s*) and also define the number of resulting web pages from search engine *s* for term combination *tc* as *npgs*(*tc*, *s*). In order to synthesize the search engine component with other components, we transform the rank information to a [0, 1] scale normalized rank, $rank^N(tc, pg, s)$ according to the following equation.

$$rank^N(tc, pg, s) = 1 - \frac{(rank(tc, pg, s) - 1)}{npg(tc, s)} \qquad (19)$$

The above normalization implies our intention to further discriminate the similarly-ranked pages depending on the size of populations of those pages. For example, it transforms the second rank of ten results to a larger value than the same second of five results. Now, to obtain a composite search engine relevance value of a web page *pg* for a term combination *tc*, $rvtc^{SE}(tc, pg)$, we adopt a weighted average method based on user's search engine preference as follows:

$$rvtc^{SE}(tc, pg) = \sum_s sw(s) \cdot rank^N(tc, pg, s) \qquad (20)$$

To synthesize this in terms of a leaf path, we also define a search engine relevance measure of a web page *pg* for a leaf path *lp* as $rvlp^{SE}(lp, pg)$ and formalize it as the equation (21).

$$rvlp^{SE}(lp, pg) = \frac{\sum_{tc \in tcslp(lp)} rvtc^{SE}(tc, pg)}{ns(lp)} \qquad (21)$$

Finally to obtain a search engine relevance value of a web page with respect to WSTT, we also adopt AHP-based WSTT relevance value composition mechanism and so, we set *bndfn*(*lp*, *pg*) in the equation (11) as $rvlp^{SE}(lp, pg)$. After value propagation process, we obtain a single synthesized search engine relevance value at the root node n_0 and assign its value, $rvn(n_0, pg)$ to *rvc*(Search Engine Component, *pg*).

3.4.6 Authority/Hub Component Relevancy Computation

Authority/Hub component: represents the level of user preference for *Authority* or *Hub* sites and pages [30]. At present, no such authority or hub ranking service exists on the Web. Therefore, we have not incorporated this component into our proof-of-concept prototype..

3.4.7 Popularity Component Relevancy Computation

Our final component to be considered is Popularity component and it represents the user's preference for popular sites. Popularity can be measured by the number of visitors or the number of requests for the specific page or site and there exist some publicly available services for this popularity information like www.yep.com. To compute the relevance value of a web page *pg* in terms of popularity component, let's introduce some definitions as follows.

pop(*pg*)	:	denotes the average number of daily visitors to the web page *pg*.
pgset	:	denotes the set of whole web pages retrieved.

Based on the definitions, we formalize the popularity relevance measure of a web page *pg* as follows:

$$rvc(\text{Popularity Component}, pg) = \frac{pop(pg)}{\max_{x \in pgset} pop(x)} \qquad (22)$$

So far, we have presented our approach for users to express their search intent, their search preference in terms of six preference components, have proposed a series of rating methods to compute each of relevance values for the components, and provide a mechanism to combine them into a single measure of relevance. Finally we use this single measure to provide the users more relevant information with a list of resulting web pages in a descending order of relevance value.

4. System Architecture of A Semantic Taxonomy-Based Personalizable Meta-Search Agent System: WebSifter II

In this section we present the architecture of WebSifter II, a semantic taxonomy-based personalizable meta-search agent system. Figure 7 shows the overall architecture of WebSifter II and its components. Major information flows are also depicted. WebSifter II consists of eight subsystems and four major information stores. Currently we have finished the detailed system design of each subsystem and the information stores; we are implementing them in the Java language.

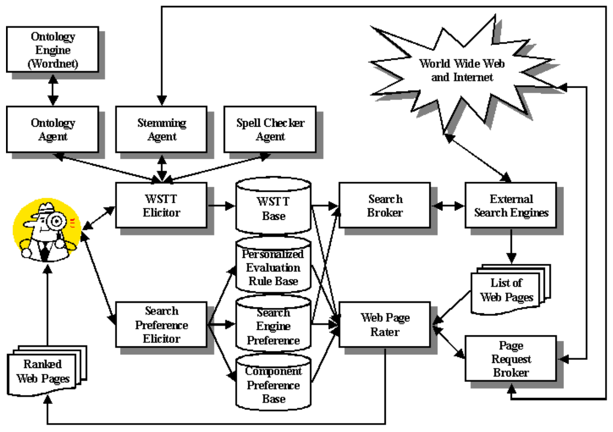


Figure 7 System Architecture of WebSifter II

Now let's briefly introduce each of the components, their roles, and related architectural issues.

1) WSTT Elicitor

The WSTT elicitor supports the entire processes required in the section 3.1 to build a WSTT in a GUI environment. A user can express his search intent as a WSTT through interactions with the WSTT elicitor. This includes building a taxonomy tree, assigning weights on each node, and choosing a concept from available list of Wordnet concepts. To achieve this goal, the WSTT elicitor also cooperates with an Ontology agent, a Stemming agent, and a Spell Check agent. Once a user finishes building a WSTT, then WSTT elicitor stores the WSTT information into the WSTT base in XML format.

2) Ontology Agent

The ontology agent is responsible for requesting available concepts of a given term via a web version of Wordnet (http://www.cogsci.princeton.edu/cgi-bin/webwn/) and also interpreting the corresponding HTTP based results. The agent receives requests for the concepts from WSTT elicitor and returns available concepts in an understandable form. Although WebSifter presently supports cooperation only with Wordnet, its design can be easily extended to cooperate with other ontology servers such as CYC [32] and EDR [33].

3) Stemming Agent

Our stemming agent is developed based on Porter's algorithm [34]. It has two major roles:1) to cooperate with WSTT elicitor in transforming the terms in a concept to the stemmed terms, and 2) to transform the content of web pages into the stemmed terms internally through cooperation with a page request broker. As a result, the terms in concepts and the terms in web pages can be compared to each other via their stemmed versions.

4) Spell Check Agent

Spell check agent monitors user's text input to the WSTT elicitor and checks and suggests correct words to the user in real time.

5) Search Preference Elicitor

Search preference elicitor, via a GUI, supports the process required in section 3.2 to capture the user's search preferences. A user can express his search preference through interaction with this search preference elicitor by assigning their preference weights to each of preference components and also to their favorite search engines. Moreover, it allows the user to modify the default values assigned to each syntactic URL class such as Direct Hit, Directory Hit and Page Hit. Whenever the user modifies them, it instantly updates the related information stored in the Personalized Evaluation Rule Base, the Search Engine Preference Base, and the Component Preference Base.

6) Search Broker

Search broker performs the processes required in section 3.3. It first interprets the XML-based WSTT and then generates all corresponding query statements. Using this set of queries, it requests information from a set of popular search engines simultaneously. Finally, it interprets the results returned from the search engines and then stores parsed information in a temporary data store. When it finishes its works, it activates web page rater so as to begin the rating process.

7) Page Request Broker

Page request broker is responsible for requesting the content of a specific URL and it cooperates with both the stemming agent and the web page rater.

8) Web Page Rater

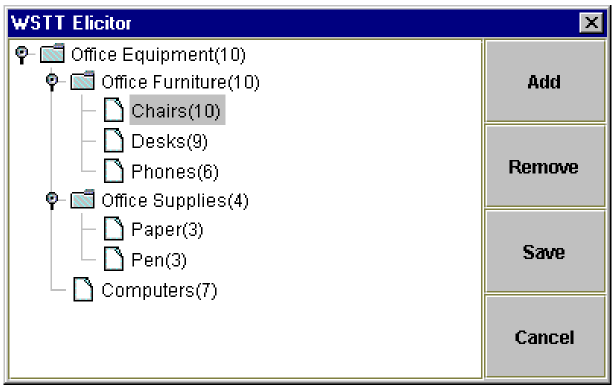
Web page rater supports the entire web page evaluation process required in section 3.4 and also is responsible for displaying the evaluation result to the users. This subsystem is the most complex and computationally intensive module of WebSifter II, and it uses all of four major information stores and also communicates with search broker and page request broker.

5. Empirical Results on Implementation

We are currently developing our meta-search agent system, WebSifter II. Some of sub-systems such as the ontology agent, stemming agent, search broker and page request broker are already developed and are operational. We have almost finished the development of the WSTT elicitor, while the search preference elicitor and the web page rater are still under development. We also plan to incorporate a commercial spell check agent into our system.

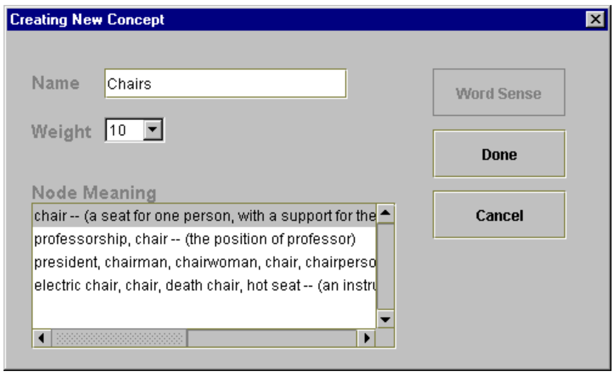
So, at this time, what we can show explicitly are some of our user interface screen to guide the user to express their search intent as WSTT. Figure 8 shows an illustrative screen where the user is building WSTT using WSTT elicitor. Figure 9 shows another screen of the WSTT elicitor supporting the selection of an intended concept from available concepts for a given term that have been obtained through cooperation with the ontology agent.

We are currently doing empirical experiments on our approach and we expect to include empirical results to support our approach when we submit the final Camera Ready Paper.



Office Equipment(10)
Office Furniture(10)
Chairs(10)
Desks(9)
Phones(6)
Office Supplies(4)
Paper(3)
Pen(3)
Computers(7)

Figure 8 An Illustrative Screen of WSTT Elicitor



Name Chairs

Weight 10

Node Meaning

chair -- (a seat for one person, with a support for the
professorship, chair -- (the position of professor)
president, chairman, chairwoman, chair, chairperso
electric chair, chair, death chair, hot seat -- (an instru

Figure 9 An Illustrative Screen for Concept Creation

6. Conclusions

We have proposed a semantic taxonomy-based personalizable meta-search agent approach to achieve two important and complementary goals: 1) allowing users more expressive power in formulating their web searches, and 2) improving the relevancy of search results based on the user's real intent. In contrast to the previous research, we have focused not only on the search problem itself, but also on the decision-making problem that motivates users to search on the web. .

Now, let's briefly summarize what we have done with three concluding remarks as follows.

First, to enhance user's search intent and preference expressional power, we propose a search-intention representation scheme, the Weighted Semantic-Taxonomy Tree, by which users express their real search intentions by specifying domain-specific concepts, assigning appropriate weights to each concept, and expressing their decision problem as a structured tree of concepts. We also allow users to express their search result evaluation preferences as a function of six preference components.

Second, to enhance the *precision* of the retrieved information, we present a hybrid rating mechanism which considers both the user's search intent represented by the WSTT and user's search preference represented by multi-preference components such as semantic relevance, syntactic relevance, categorical match, page popularity, and authority/hub rating.

Third, we have designed and are presently implementing a meta-search agent system called WebSifter II that cooperates with Wordnet for concept retrieval, and most well known search engines for web page retrieval. For the empirical validation of our approach, we are also doing some real world experiments of our system.

References

[1] Scime, A. and L. Kerschberg, "WebSifter: An Ontology-Based Personalizable Search Agent for the Web," *International Conference on Digital Libraries: Research and Practice*, Kyoto Japan, 2000, pp. 493-446.

[2] Lawrence, S. and C. L. Giles, "Accessibility of Information on the Web," *Nature*, vol. 400, 1999, pp. 107-109.

[3] Selberg, E. and O. Etzioni, "The MetaCrawler architecture for resource aggregation on the Web," *IEEE Expert*, vol. 12, no. 1, 1997, pp. 11-14.

[4] Howe, A. E. and D. Dreilinger, "Savvy Search: A Metasearch Engine That Learns Which Search Engines to Query," *AI Magazine*, vol. 18, no. 2, 1997, pp. 19-25.

[5] Lawrence, S. and C. L. Giles, "Context and page analysis for improved Web search," *IEEE Internet Computing*, vol. 2, no. 4, 1998, pp. 38-46.

[6] Ackerman, M., et al., "Learning Probabilistic User Profiles - Applications for Finding Interesting Web Sites, Notifying Users of Relevant Changes to Web Pages, and Locating Grant Opportunities," *AI Magazine*, vol. 18, no. 2, 1997, pp. 47-56.

[7] Armstrong, R., et al., "WebWatcher: A Learning Apprentice for the World Wide Web," *Proceedings of the 1995 AAAI Spring Symposium on Information Gathering from Heterogeneous, Distributed Environments*, 1995.

[8] Shavlik, J. and T. Eliassi-Rad, "Building intelligent agents for web-based tasks: A theory-Refinement approach," *Proceedings of the Conference on Automated Learning and Discovery: Workshop on Learning from Text and the Web*, Pittsburgh, PA, 1998.

[9] Chen, Z., et al., "WebSail: from on-line learning to Web search," *Proceedings of the First International Conference on Web Information Systems Engineering*, vol. 1, 2000, pp. 206-213.

[10] Chakrabarti, S., et al., "Enhanced hypertext categorization using hyperlinks," *Proceedings of ACM SIGMOD International Conference on Management of Data*, Seattle, Washington, 1998, pp. 307-318.

[11] Li, Y., "Toward a qualitative search engine," *IEEE Internet Computing*, vol. 2, no. 4, 1998, pp. 24-29.

[12] Maes, P., "Agents that reduce work and information overload," *Communications of the ACM*, vol. 37, no. 7, 1994, pp. 30-40.

[13] Terveen, L., et al., "PHOAKS: a system for sharing recommendations," *Communications of the ACM*, vol. 40, no. 3, 1997, pp. 59-62.

[14] Bollacker, K. D., et al., "Discovering Relevant Scientific Literature on the Web," *IEEE Intelligent Systems*, vol. 15, no. 2, 2000, pp. 42-47.

[15] Balabanovic, M. and Y. Shoham, "Content-Based, Collaborative Recommendation," *Communications of the ACM*, vol. 40, no. 3, 1997, pp. 66-72.

[16] Krulwich, B., "Lifestyle Finder," *AI Magazine*, vol. 18, no. 2, 1997, pp. 37-46.

[17] de Vel, O. and S. Nesbitt, "A Collaborative Filtering Agent System for Dynamic Virtual Communities on the Web," *Working notes of Learning from Text and the Web, Conference on Automated Learning and Discovery CONALD-98*, Carnegie Mellon University, Pittsburgh, 1998.

[18] Chen, H. and S. Dumais, "Bringing order to the Web: automatically categorizing search results," *Proceedings of the CHI 2000 conference on Human factors in computing systems*, The Hague Netherlands, 2000, pp. 145-152.

[19] Aridor, Y., et al., "Knowledge Agent on the Web," *Proceedings of the 4th International Workshop on Cooperative Information Agents IV*, 2000, pp. 15-26.

[20] Chakrabarti, S., et al., "Focused Crawling: A New Approach to Topic-Specific Web Resource Discovery," *Proceedings of the Eighth International WWW Conference*, 1999, pp. 545-562.

[21] Salton, G., et al., "A Vector Space Model for Automatic Indexing," *Communications of the ACM*, vol. 18, no. 11, 1975, pp. 613-620.

[22] Clark, D., "Mad cows, metathesauri, and meaning," *IEEE Intelligent Systems*, vol. 14, no. 1, 1999, pp. 75-77.

[23] Guarino, N., et al., "OntoSeek: content-based access to the Web," *IEEE Intelligent Systems*, vol. 14, no. 3, 1999, pp. 70-80.

[24] Fensel, D., et al., "On2broker: Semantic-Based Access to Information Sources at the WWW," *Proceedings of the World Conference on the WWW and Internet (WebNet 99)*, Honolulu, Hawaii, USA, 1999, pp. 25-30.

[25] Staab, S., et al., "A System for Facilitating and Enhancing Web Search," *Proceedings of IWANN '99 - International Working Conference on Artificial and Natural Neural Networks*, Berlin, Heidelberg, 1999.

[26] Martin, P. and P. W. Eklund, "Knowledge retrieval and the World Wide Web," *IEEE Intelligent Systems*, vol. 15, no. 3, 2000, pp. 18-25.

[27] Miller, G. A., "WordNet a lexical database for English," *Communications of the ACM*, vol. 38, no. 11, 1995, pp. 39-41.

[28] Klein, D. A., *Decision-Analytic Intelligent Systems: Automated Explanation and Knowledge Acquisition*, Lawrence Erlbaum Associates, 1994.

[29] Boose, J. H. and J. M. Bradshaw, "Expertise Transfer and Complex Problems: Using AQUINAS as a Knowledge-acquisition Workbench for Knowledge-Based Systems," *Int. J. Man-Machine Studies*, vol. 26, 1987, pp. 3-28.

[30] Kleinberg, J. M., "Authoritative sources in a hyperlinked environment," *Journal of the ACM*, vol. 46, no. 5, 1999, pp. 604-632.

[31] Saaty, T. L., *The Analytic Hierarchy Process*, New York, McGraw-Hill, 1980.

[32] Lenat, D. B., "Cyc: A Large-Scale Investment in Knowledge Infrastructure," *Communications of the ACM*, vol. 38, no. 11, 1995, pp. 33-38.

[33] Yokoi, T., "The EDR Electronic Dictionary," *Communications of the ACM*, vol. 38, no. 11, 1995, pp. 45-48.

[34] Porter, M., "An Algorithm for Suffix Stripping," available at ttp://www.muscat.co.uk/~martin/def.txt.