Recommending Web Documents Based on User Preferences

Eric J. Glover[1,2] Steve Lawrence[1] Michael D. Gordon[3] William P. Birmingham[2] C. Lee Giles[1]

{compuman,lawrence,giles}@research.nj.nec.com[1]

{compuman,wpb}@eecs.umich.edu[2]

{mdgordon}@umich.edu[3]

NEC Research Institute[1]
4 Independence Way
Princeton, NJ 08540

Artificial Intelligence Laboratory[2]
University of Michigan
1101 Beal Avenue
Ann Arbor, MI 48109-2110

Business Administration[3]
University of Michigan
701 Tappan St
Ann Arbor, MI 48109-1234

Abstract

Making recommendations requires treating users as individuals. In this paper, we describe a metasearch engine available at NEC Research Institute that allows individual *search strategies* to be used. Each search strategy consists of a different set of sources, different query modification rules and a personalized ordering policy. We combine these three features with a dynamic interface that allows users to see the "current best" recommendations displayed at all times, and allows results to be displayed immediately upon retrieval. We present several examples where a single query produces different results, ordered based on different factors, accomplished without the use of training, or a local database.

1 Introduction

In this paper we describe Inquirus 2, a metasearch engine with a dynamic interface that makes individual recommendations based on user preferences. Users of Inquirus 2 specify both a keyword query and an information need category. The combination of the query and information need category is used to produce a personalized ordering of results found via a search strategy specific to the user's need. Since the search process does not require state about the user[1], no training is necessary. The dynamic interface allows the display and ordering of results, as they are found, reducing the wait time for users.

At NEC Research Institute, researchers have many different search needs, ranging from searching for organizations related to some research area, to research papers on some topic. The goal of our project was to create a single search system, by extending Inquirus [12, 13], capable of producing meaningful results tailored for each specific need. Unlike typical recommender systems, we do not have a local database. Rather, we use the web as our virtual database. Inquirus, a metasearch engine, sends user queries to over a dozen different Internet search engines and combines the results based on the predicted relevance to the given query. Inquirus 2 extends the notion of relevance to include user preferences. As a result, different researchers with the same query will receive personalized recommendations from a search strategy consistent with their need.

User preferences affect three parts of the search process, described in detail in Section 2: the sources used, modifications to the query, and the ordering policy for the results. A search strategy refers to the collection of these three decisions, and should be consistent with the stated user need. For example, a user looking for current events might prefer more recent documents to older ones, whereas a user looking for organizational homepages might prefer web pages with a shorter pathlength (i.e., top of a site's path) to those farther down the tree. Likewise a user searching for current events would likely search a news specific site, whereas a user looking for company homepages might not. To capture this, users specify an information need category in addition to a keyword query. The selection of the information need category determines the search strategy, which includes an associated utility function that determines how to score the results. Every user can have their own personal set of information need categories (and associated search strategies), or can use the "expert" defined categories available to everyone.

Figure 1 shows the Inquirus 2 interface and some of the information need categories available. The search strategies are stored in a text file, and can be easily added or edited. Our future work includes using learning to define better search strategies (learning the optimal list of sources and query modifications as well as the best utility functions for

[1] The architecture could use state if it is determined to improve results, although it does not require state to make meaningful recommendations given an information need category.

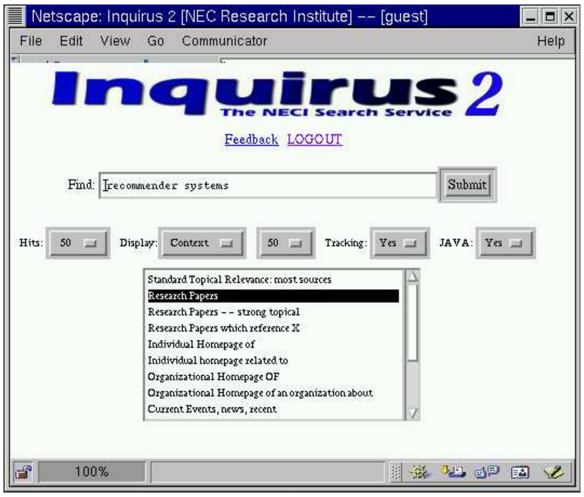

Figure 1: Screen shot of the Inquirus 2 interface

each need category).

1.1 Metasearch Engines

The World Wide Web, estimated to be 800 million pages [15], lacks organization with respect to content, due to the diversity of web pages and web-page authors; anyone can be a publisher. To enable users to locate "relevant" information, Internet search engines were created. Tools such as Compaq's AltaVista (`www.altavista.com`), or Infoseek (`www.infoseek.com`) allow users to search indexed web pages by entering a keyword query. The basic architecture of a regular Internet search engine is shown in Figure 2. Research has shown that searching more than one search engine can significantly enhance coverage, locating significantly more results than one engine alone [3, 9, 14]. In addition, there are several specialized content web tools, such as ABCNews (`www.abcnews.com`), for example, which specializes in News, or Yahoo (`www.yahoo.com`), which is a manually built hierarchy by subject.

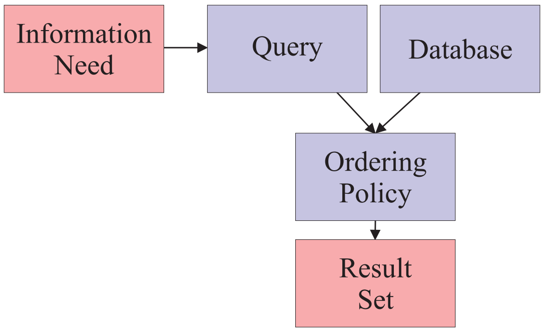

Figure 2: The architecture of a standard search engine with no feedback

A user might not know of every search engine or might not have the patience to submit their query to more than one. As a result metasearch engines were created. Some metasearch engines include: Metacrawler (`www.metacrawler.com`), SavvySearch [9] (`savvysearch.com`), MetaSEEK [3] (`http://www.ctr.columbia.edu/MetaSEEk/`) and ProFusion [6] (`profusion.com`). Figure 3 shows the architecture of a typical metasearch engine. Compared with a regular search engine, a metasearch engine does not have a local database and relies on other sources (other search engines) for its data. The results returned from various other search engines are combined through some *combination policy* also, called a *fusion policy*. The fusion policy is analogous to the ordering policy for a regular search engine.

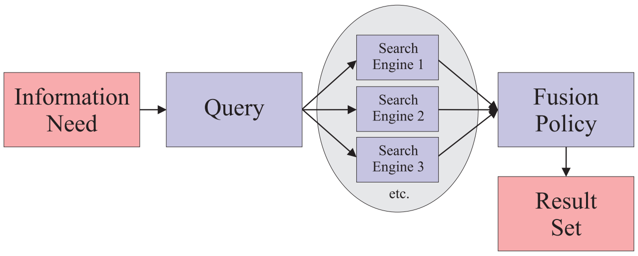

Figure 3: The architecture of a standard metasearch engine

A typical metasearch engine, such as DogPile (`dogpile.com`) uses a fixed list of sources, and combines results by simply listing the results in the original order given by the search engines. A user with a relatively broad query will be presented with numerous results. Some metasearch engines allow for limited personalization. SavvySearch [9] allows users to choose a category, and only sources which specialize in that area are considered. For example, a user searching for "news" will only search News specific sources. This approach helps to improve the precision of the results, but does not guarantee any meaningful ordering of results. In addition, many potentially good results, found through general-purpose search engines, but not in the specialized ones, are excluded.

Another limitation of typical metasearch engines is their inability to score results based on the complete content. A standard Internet search engine has a local database and considers the entire contents of a web page when making scoring decisions. Most metasearch engines consider only the title, URL, and other information, such as a short summary, returned from the search engines. As a result, such a metasearch engine cannot guarantee consistent scoring.

1.2 Inquirus

To improve upon regular metasearch engines, Inquirus [12, 13] was created. Inquirus, shown in Figure 4, adds several new features not previously available. First, Inquirus uses a page retriever and page analyzer to guarantee consistent result scoring. Unlike regular Internet search engines, Inquirus always has the most recent version of a page, eliminating

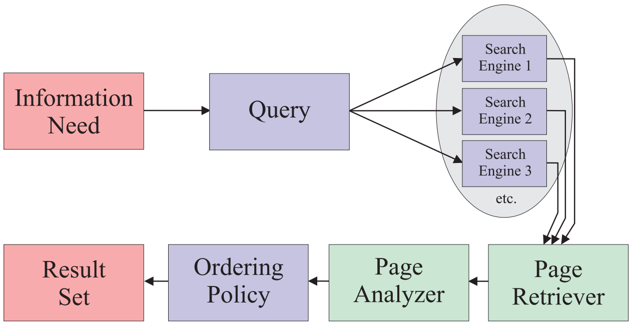

Figure 4: The architecture of the Inquirus search engine

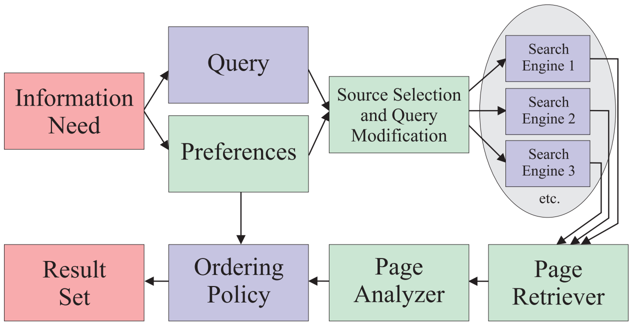

Figure 5: The architecture of the Inquirus 2 search engine

dead links and pages that are no-longer relevant. Second, Inquirus utilizes the full HTML to improve upon the search by providing keyword context information for every result. Inquirus provides many additional interface functions, including alternate query recommendations if too many results are found.

Like a standard metasearch engine, Inquirus sends the user query to a set of search engines[2]. However, unlike a standard metasearch engine, Inquirus downloads the entire contents of a web page and uses an ordering policy to rank the results. This approach can significantly improve precision because dead links and no-longer relevant pages are filtered out.

2 Architecture of Inquirus 2

The goal of Inquirus 2 was to allow for individual information needs and personalized search strategies, while retaining a simple search interface. The basic architecture of Inquirus 2 is shown in Figure 5. There are several changes made to Inquirus: the addition of a source selection and query modification module, replacing the fixed ordering policy with a preference-based ordering policy, addition of more attributes to the page analyzer, and an explicit specification of preferences, as distinct from the query. Separate from the architecture, we added a dynamic interface to allow scored results to be available immediately.

The new architecture allows specification of "information need categories" that each have an individual search strategy. Table 2 lists several currently available information need categories available to users. The search strategy can vary with respect to the sources searched, the query modifications performed, and the scoring function used to recommend results. Section 2.1 describes in detail the source selection process of Inquirus 2, and its effects on the results. Section 2.2 describes in detail the types of query modifications, and how the modifications increase the precision of the results with respect to the given information need category. Section 2.3 describes the individual scoring functions used and their form.

[2] Inquirus does support Image or News queries that query different search engines, but when searching for web content the same set is always used.

2.1 Source Selection

Intelligent source selection can help to improve the precision of results [3, 6, 9] as opposed to simply finding more. Several metasearch engines, SavvySearch, MetaSEEK, and ProFusion already perform some intelligent source selection. A simple example is a user looking for current events, versus a user searching for a company. The user searching for news might search a news site, while the other user might search Yahoo.

In the current version of Inquirus 2, each information need category specifies a list of appropriate sources. Choosing too many sources, especially those with many results of low value, can slow down the search process by requiring the downloading of low-valued content. Choosing many sources can also increase coverage, allowing users to find a greater number of relevant pages.

One method of choosing sources is by category. SavvySearch allows users to choose a category, such as news or auctions, and only search "relevant" sources for that category. This approach runs the risk of missing results from the more general, but not searched, search engines. ProFusion attempts to predict which search engines are most likely to have results for a given query by predicting the subject [6]. Inquirus 2 currently uses a fixed list of sources for each category, but can include the general search engines for specific needs, because of the ability to automatically modify the query, and score results based on the predicted value (for the given need). For example: A user searching for "news" might not normally search Northern Light, a general-purpose search engine. However, the user can specify that Northern Light return results in order by date, or only return results from a given date range. Likewise, searching AltaVista for research papers might normally return very few results that are research papers, but modifying the query to add the keyword "abstract" can increase the precision (with respect to the need for research papers), resulting in fewer erroneous results.

With Inquirus 2, it is acceptable to search a source that contains many good and many low-valued results, since it uses a scoring function that captures the specific information need, allowing it to filter out "bad" results. The combination

Name	Description
agrade	Average of three grade level algorithms, FOG, SMOG, and FK
GFOG	A reading level algorithm optimized for less advanced documents
daysOld	The predicted number of days old as computed by analyzing the full text and HTML not only considering the header
wordcount	The number of words per page
homepage	A measure of the number of homepage like features present
genscore	A measure of features indicative of a "general" page, such as the keywords "links" or "resources"
researchpaper	A measure of features indicative of a "research paper" page, such as having an abstract or references
anchorcount	The number of unique links present on a page
imagecount	The number of unique images present on a page
numkeywords	The number of keywords in the query matched on a page
sectioncount	The number of sections on a page
pathlength	The depth of a page from the top of a domain in levels
summary	An automatically generated summarization of the document
topicalrelevance	A query dependent attribute predicting how much a particular page is "about" the given query. Attribute is based on word distances, from each other and the top of the document, as well as number of occurrences of each term
latex	Binary attribute: true if page was generated by LaTeX2HTML, false otherwise

Table 1: List of some of the page specific attributes and their description

Name	Description	Search engines used
Research papers (or references to)	Detailed pages, preferably an actual article	Google* AltaVista Snap Yahoo* HotBot NorthernLight
Individual homepages	The homepage(s) of the individual listed in the query	Snap Google HotBot Yahoo
Organizational homepage of	The homepage(s) of the organization listed in the query	Snap Google HotBot Yahoo
Current events, news recent	Recent articles, or content about the given query, with significant content	ABCNews News.com Snap AltaVista Yahoo HotBot*
General introductory about	Getting started, references, "What is", etc...	Google* AltaVista* Snap Yahoo

Table 2: Information need categories, and the search engines used. * means the query sent to the search engine is modified to enhance precision, in some cases more than one modified query may be sent to the same engine

of query modification and information need based scoring functions extend the possible sources allowable for a given query and information need.

2.2 Query Modification

To enhance the precision of the results returned from the search engines and to allow use of general web search engines for specific needs, automatic query modification is performed. The simplest modifications include adding non-topical constraints, such as specifying "in the last two weeks" for the need "current events" when searching HotBot. More complex modifications include adding extra keywords to the query, such as adding "abstract keywords introduction" for the need "research papers" when searching Google. For a preference of "general introductory," a modified query of "what is X," where X is the original query entered by the user, is sent to AltaVista to help find general web pages about X. In some cases, more than one query is sent to a particular search engine to ensure good results are not removed by a particular query modification.

The effects of dynamic query modification can be seen in Tables 5 and 6, where all of the top ranked results were found through the use of modified queries. In theory, an unmodified query should eventually find the results of the modified one. In practice, however, search engines limit the total number of results retrievable, so an overly general query has the effect of removing valuable results. Adding extra terms, such as "abstract" may not affect the "aboutness" of a particular result, but will affect the chances of that page being a research paper. Since pages are scored based on the user entered query, adding extra terms to the queries submitted

Rank	Score	Page Title	Rank	Score	Page Title
1	717	Links for Will michael jordan Really Retire?	1	953	michael jordan at OnOnline.com, Find Pictures, Movies, Sounds, and Links
2	593	Chicago Bulls Internet Treasure Hunt	2	873	jordan, michael resources from Nerd World Media
3	581	TO BE DEAD BEFORE BORN by Mert Dogan	3	871	jordan, michael resources from Nerd World Media
4	579	Bryce	4	801	NBA links...
5	542	Capitalism and Consumption	5	792	michael jordan Links
6	417	http://www.capeathletic.com/lvl2/ lvl3/profile.html	6	717	Links for Will michael jordan Really Retire?

Table 3: Left: Top 6 results for the query `Michael Jordan` and the information need category "general introductory", after a few seconds, Right: Top 6 results, same query, after 50 total pages downloaded

to the search engines will not affect individual result scores. Increasing the precision (number of "good" results) can have the effect of speeding up the search, since fewer results need to be retrieved in order to find valuable ones. Section 4.2 describes the effect query modification had for the specific query `Pattie Maes` on the number of research papers found from Google and Yahoo.

Query modification can also alters how search engines are used, to further enhance result precision. For example, Northern Light, like ABCNews, allows results to be returned in Date or relevance order. Depending on the individual need, it might make sense to use different search options.

2.3 Personalized Result Scoring

A typical search engine, such as Lycos, scores results based on the keywords in the query and the terms in the document [16]. Typical metasearch engines score documents based on the original scores returned from the search engines queried, running the risk that the actual pages are no longer relevant, or the page scored high as a result of keyword spamming[3] or paid advertising. Inquirus deals with this by downloading every web page and applying its own scoring function. To further improve upon this, Inquirus 2 uses an ordering policy, Figure 5, defined by the user's preferences.

We treat the document-ordering task as a decision problem, and use utility theory [10] as the model for evaluating the results. The ordering policy is "sort by value," where utility theory provides the mechanism for assigning value. Each user-selected information need category has an associated additive value function of the form shown in,

$$\mathcal{U}(d_j) = \sum_k w_k v_k(x_{jk}) \tag{1}$$

where w_k is the weight of the k_{th} attribute, and by convention totals one, v_k is the value function for the k_{th} attribute, x_{jk} is the level of the k_{th} attribute for the j_{th} document, and by convention: $\forall k, d : v_k(d) \in [0,1]$.

Every need has several different significant attributes. Table 1 lists several of the page-specific attributes, which are used to form the individual utility functions. The additive linear form allows attributes to be differently mapped to a value. For example, for a research paper, the longer the better (to some maximum) for a general resource, a shorter page may be better than a very long one. Second, each attribute has a weight or relative importance. When looking for the homepage of an organization or company, the pathlength (how far from the top of the tree the page is) is slightly less important than the fact that the keywords occur in the title. Inquirus 2 allows every user to have their own personal utility functions, in addition to the provided information need categories' functions. Table 2 lists several of the categories available to every user. When a user logs in, their personal list is appended to the default choices.

Utility functions allow every user to have valuable results presented first. Combining utility functions with intelligent source selection and query modification provides higher precision results, reducing the number of "bad" results processed.

3 Dynamic Interface

When searching, users want results immediately. If the initial results are acceptable, users should be able to begin examining them as soon as they are processed by the system. Inquirus 2 allows users to see results as soon as they are scored. The (optional) dynamic Java-based interface ensures that the current best results are immediately available by sorting results in the applet as they are retrieved. As more documents are processed, the value of the best documents increases. Without a dynamic interface, the system can not show any sorted results until all results have been scored.

Table 3 lists the top six results as seen after only a few seconds, and at the completion of the search (when asking for 50 total results). Of the initial six, only the first two seem

[3] Keyword spamming is an attempt by content providers to cause their page to be ranked highly by actively altering the HTML to take advantage of the scoring functions used by the search engines.

#	Title	Comment
1	Pattie Maes' Home Page	Ranked 8 by Yahoo, first by Google, and 13 by Snap.
2	Pattie Maes Slides, VUB Presentations	This page lists several presentations, and had a link to her homepage, ranked 6 by Yahoo
3,4	politik-digital : Kopf der Woche : Pattie Maes	Two parts of an interview (in German) with Pattie Maes, the first contains a link to her homepage
5	EDGE 3rd Culture: INTELLIGENCE AUGMENTATION - A Talk With pattie maes	A brief chat with Pattie Maes.
6	Who is who @ Mediamatic - Patty Meas.	Misspelled name in title, a description of her, her work, and a link to her homepage

Table 4: Results for the query `pattie maes` with a preference of 'individual homepage of', a total of 50 documents were downloaded.

at all valuable as general resources about "Michael Jordan". As the search progressed, the top ten results continually improved. If the user had asked for 200 total pages, and waited until the search completed, we would expect the top ten would be even better than those shown in the table. The extra 150 documents may contain some which score better than the top ten found from only the first 50 documents.

The dynamic interface uses a Java applet that connects to a port opened by the CGI script. As a result is scored, it is sent to the applet, where it is inserted into the correct position. If the new result is worse than the current 20th best, it is not displayed. All the results shown by the applet are clickable, causing a different browser window to display the result so that the search is not interrupted.

4 Results

To demonstrate Inquirus 2, we ran the same query `Pattie Maes` with three different search strategies: Research papers, Research Papers that reference X, and individual homepages. The first two are very similar, the third demonstrates how "good" pages, which are not "exactly what you want," may still be valuable. Tables 4, 5 and 6 show several of these results, and the corresponding sections describe in more detail the search strategies used.

4.1 Individual Homepages

One of the first information need categories we made available to all users of Inquirus 2 was "individual homepage of." A user searching with this category is likely looking for the homepage of the query. The current version of this search strategy submits the query to: SNAP, Google, Yahoo-Inktomi, and HotBot. The query is not modified.

The scoring of a page using this strategy is a function of the keywords in the title, the keywords occurring in the automatically generated summary, the topical relevance and the homepage score. A page that is not the exact homepage might still score highly, as can be seen in the results shown in Table 4.

Pattie Maes' homepage was ranked first using this search strategy. It was also ranged first by Google. The pages ranked second through sixth may not be her homepage, but are reasonable, and valuable for a person with this information need. Of the pages ranked second through sixth, all but one had a link to Pattie Maes' homepage, and all were directly related to her personal information.

If none of the search engines returned her actual homepage, a page that directly linked to her homepage, or a page describing her might be the next best choice. If we could have instructed the search engines to "find only homepages," and her homepage was not listed, none of the high ranked pages would have been retrieved.

It is important to distinguish a person's homepage a company's, or a homepage which is "about" a query as opposed to "of" the query. We have implemented a different search strategy for searching for homepages of organizations that considers *pathlength* as one of the most important attributes, while an individual homepages does not.

If a user were interested in finding homepages of students of Pattie Maes, the search strategy would be different, and the relative importance of "Pattie Maes" in the title would be reduced.

4.2 Research Papers

At a research laboratory a common information need is to find research papers about some topic or by some author. On the web, however, there are many different types of pages that may be valuable. Simply tagging a page as a research paper is insufficient. A user might consider a bibliography page or a very detailed web page (that was not a research paper) acceptable, but not as good as other types of pages, even though neither are strictly research papers.

The form of utility function shown in Equation 1 assumes strict independence and considers only one "type" of web page. For example, one type of research paper might be a long, detailed, web page made to look like a conference pa-

#	Title	Comment
1	Communication and Learning on the Internet	Ranked 31 by a modified query to Google, this page is a research paper, which cites Pattie Maes
2	Zypher MOP	A paper which cites Pattie Maes, ranked 34 by a modified query to Yahoo
3	Footprints, Version 1	A research paper written by Pattie Maes, found by a modified query to Google
4	Wexelblat/maes: Issues for Software Agent UI	A research paper by Pattie Maes, found from a modified query to Google
5	Extended abstract	Shortened research paper titled: "Approaches to Integrated Malware Detection and Avoidance," found by a modified query to Yahoo
6	HOW TO DO THE RIGHT THING	A paper with Pattie Maes as an author, found from a modified query to Yahoo
7	Cooperating Mobile Agents for Mapping Networks	A paper with Pattie Maes as an author, found from a modified query to Yahoo

Table 5: Results for the query `pattie maes`, with a preference of 'Research Papers', 100 total documents downloaded, the first 12 documents were research papers, four with Pattie Maes as an author, and the remaing eight referenced her

per, another might be an abstract and a link to the .ps file. Both may be considered valuable, but there exists no function of the desired additive linear form that captures both simultaneous preferences. One solution is to have two different preferences, and have the user run two searches. Instead, we are experimenting with using a MAX function over two different additive linear functions. Currently, the actual functions used are hand coded, so we can "fine-tune" the function to capture both needs. When we begin using learning to discover user's utility functions, it will be necessary to derive a mathematical model for combining two utility functions.

Table 5 shows several results for a query for `Pattie Maes` with a search strategy of "Research Papers." Compare the results found in Table 6 which are for the need "Research Papers which Reference X." Both needs are similar, but the differences are represented in the different query modifications used, as well as different associated utility functions.

The primary attributes used in the utility function for "Research Papers" include the average grade level, *researchpaper*, *topicalrelevance*, *wordspersection* and others depending on the "type" of web page we are looking for. For long web pages that are full research papers, we consider *wordcount*. For web pages generated from LaTeX2HTML[4], we consider the *latex* attribute. The third type of web page that we consider good is one that has a short abstract and has a postscript (or pdf) file. For this type of web page, we consider the number of postscript or pdf files.

The primary attribute *researchpaper* is a function of how many "research paper" type features are found on the page. Such features include a heading called "Abstract" or "References;" features common to standard writing style of research papers. Other types of pages, such as an abstract (but not a whole research paper) or a reference list will score highly, but not as highly as a full research paper. We feel this is consistent with how a searcher would assign value when looking for research papers. If the full paper is not available, an abstract or bibliography might be the next best choice.

Of the top 12 documents found for this query and search strategy, all of them were full research papers, and four of them had Pattie Maes as an author. The same query sent without modification finds very few research papers. None of the first 10 pages returned from Google were research papers, of any kind for the unmodified query. Whereas eight of the top ten for the modified query to Google were full research papers, and two were lists of research papers. The modified query for Yahoo had all ten of the first ten pages as full research papers, while the unmodified query had one page as an abstract and a link to a .pdf, plus two publication lists, but no full research papers.

This need is not specific to searching for authors, i.e., one could search for research papers on "information filtering" and achieve similar results. This is possible by considering the *topicalrelevance* attribute, which places some importance on the terms being near the beginning of the document. Unfortunately, when searching for an author, a paper written by him and one that references him near the beginning of the document will score similarly. This is why so many papers that referenced Pattie Maes scored as high as those *by* her. We plan to add an attribute, similar to *wordsinrefspercent* which only considers text prior to the abstract (i.e., title and authors of a research paper).

4.3 Research Papers which Reference X

A similar need to "Research Papers" is papers that reference someone (or some topic). Both needs require a strategy (sources and query modifications) that pull out research papers and consider similar attributes. There are two key differences: first, when looking for papers that reference something, we want to make the "relevance" judgment on whether

[4] LaTeX2HTML is a tool for converting a LaTeX document into smaller web pages. We examine the web page comments to detect this, and use it as a strong indicator that the web page is a part of a larger research paper.

#	Title	Comment
1	Minimal Multi-Agent Systems	Originally ranked 11 by a modified query to Yahoo, and 7 by a modified query to Google, this research paper references Pattie Maes
2	An Agent System For Media on Demand Services	Found from a modified query to Yahoo, references Pattie Maes
3	Coordination without Communication	Found from a modified query to Google
4	Interactive adaptation of Intranet newsletters	Found both from a modified query to Google and Yahoo
5	ROCOCO - Home Page	Actual page was a paper titled "Trends in Distance Education: Interactive Hypermedia Educational Modules", originally ranked 12 by a modified query to Yahoo

Table 6: Results for the query `pattie maes` with a preference of 'research papers which reference X', the top 14 ranked results were research papers, none authored by Pattie Maes, but all of which referenced Pattie Maes. 100 total documents were downloaded.

the keywords occur as part of the "references" section, not anywhere in the document. Second, we wish to find papers with reference lists. Unfortunately, a web page with an abstract and a link to a .ps or .pdf file will score low, since we do not download the file to determine if the reference occurs in the reference section.

Table 6 shows the first five results for the query for `Pattie Maes` for the information need category of "Research Papers that Reference X." Of the top 14 results, all of them were research papers that referenced Pattie Maes. None of the top 14 results were authored by her. This contrasts to four in the top 14 written *by* her for the need of "Research Papers." We distinguish between the two by using the attribute *wordsinrefspercent* that considers only keywords occurring in the reference section. In addition, we added a modified query to Google that looks for common terms found in references such as "pp" or "no." or "vol."

A user can also use this search strategy to find papers that reference certain areas, such as "information retrieval" or "social filtering." Such a query returns reference lists on the topic, or papers that have references on the topic. Since we also consider the original *topicalrelevance* attribute, a paper which is both "about" the query and also references it will score higher than one which simply contains the keywords in the references.

We plan on adding attributes to identify specific references as opposed to simply considering percentages of keywords found in the reference section of a web page.

5 Related Work

The notion of considering multiple features for information retrieval systems is not new, work by Barry [2] describes experiments that showed that users considered several features when making relevance judgments, many of which were non-topical, i.e.,, recency. Mizzaro [17] presents an excellent survey paper describing the concept of *relevance* and the related works from 1977 through 1997.

The concept of using utility functions to personalize search is also not new. Cooper [4] suggested *utility* as the measure for information retrieval systems, but did not describe how to build a system using it. Kochen [11] suggested applying utility theory specifically to documents and described four axioms, which if met, imply the existence of a utility function that could be used to order documents. Those four axioms were reduced to the three described by Fishburn [5] to describe the existence of a utility function (in general).

More recently, work by Glover and Birmingham [8, 7] describes an agent, as part of the University of Michigan Digital Library project [1], that used utility functions to dynamically re-order web pages. Also the Decision-Theoretic Video Advisor (DIVA) project [18] uses individual utility functions to make recommendations. Both of these projects demonstrate the importance of considering multiple attributes (including non-topical) or features when making recommendations.

Several projects, including SavvySearch [9], MetaSEEK [3] and ProFusion [6] demonstrate the use of intelligent source selection as a means of improving precision and increasing coverage for a web search in a metasearch environment.

6 Future work

Inquirus 2 is a powerful tool for personalizing web searching. We are currently exploring using various forms of learning and collaboration to create improved information need categories. We will experiment with a collaborative notion of information need categories by allowing the feedback from many users to further improve utility functions.

Users of the system will be asked to comment on the *value* of various results. These data points will be used to refine the utility functions and possibly choose alternate sources or query modifications. Many users can provide feedback for a single category, thus creating a more group oriented, or collaborative utility function. The revised utility functions will be available, without requiring state, for the future users.

The primary problems are the ability to collect sufficient data to improve (or learn from scratch) the utility functions. Our utility functions are composed of multiple attributes, thus increasing the number of data points required for training or learning over other less data intensive methods.

7 Summary and Conclusions

In this paper, we described Inquirus 2, a personalized metasearch engine in use at NEC Research Institute. Inquirus 2 employs several types of personalization and a dynamic interface to make searching both easy and personal. Users of Inquirus 2 can use several of the predefined information need categories, such as "research papers" or "individual homepages" to help determine where to search, how to search (query modification), and how results should be scored (recommended). The modular architecture reads the search strategies at runtime, and makes different decisions for different users, even if their keyword queries happen to be the same.

Inquirus 2 personalizes web searching, so that a local database of user recommendations is not needed, allowing it to operate immediately on previously unseen content. Since Inquirus 2 is a general-purpose metasearch engine, new search engines can be easily added as they are discovered. The simple architecture allows a large variety of information need categories, each fine tuned to the individual searcher, increasing usefulness over a standard search engine or metasearch engine.

The techniques employed, need-based source selection, dynamic query modification, and a utility-based ordering policy, produce an architecture that provides valuable results for many different needs, while only downloading a relatively small number of total documents. The combination of these three techniques also extends the type of sources that can be used, without fear of "bad" results scoring highly.

References

[1] Daniel E. Atkins, William P. Birmingham, Edmund H. Durfee, Eric J. Glover, Tracy Mullen, Elke A. Rundensteiner, Elliot Soloway, Jose M. Vidal, Raven Wallace, and Michael P. Wellman. Toward inquiry-based education through interacting software agents. *IEEE Computer*, 29(5):69–76, 1996.

[2] Carol L. Barry. *The Identification of User Criteria of Relevance and Document Characteristics: Beyond the Topical Approach to Information Retrieval*. PhD thesis, Syracuse, 1993.

[3] Ana B. Benitez, Mandis Beigi, and Shih-Fu Chang. Using relevance feedback in content-based image metasearch. *IEEE Internet Computing*, 2(4):58–69, 1998.

[4] W. S. Cooper. A definition of relevance for information retrieval. *Information Storage and Retrieval*, 7:19–37, 1971.

[5] Peter C. Fishburn. *Nonlinear Preference and Utility Theory*. The Johns Hopkins University Press, 1988.

[6] Susan Gauch, Guihun Wang, and Mario Gomez. ProFusion: Intelligent fusion from multiple, distributed search engines. *Journal of Universal Computer Science*, 2(9), 1996.

[7] Eric J. Glover and William P. Birmingham. Using decision theory to order documents. In *Digital Libraries 98*, Pittsburgh, PA, 1998. ACM.

[8] Eric J. Glover, William P. Birmingham, and Michael D. Gordon. Improving web search using utility theory. In *Web Information and Data Management (WIDM'98)*, pages 5–8, Bethesda, MD, 1998. ACM.

[9] Adele E. Howe and Daniel Dreilinger. SavvySearch: A meta-search engine that learns which search engines to query. *AI Magazine*, 18(2), 1997.

[10] Ralph L. Keeney and Howard Raiffa. *Decisions with Multiple Objectives*. John Wiley and Sons, New York, 1976.

[11] Manfred Kochen. *Principles of Information Retrieval*. Melville Publishing Company, Los Angeles, California, 1974.

[12] Steve Lawrence and C. Lee Giles. Context and page analysis for improved web search. *IEEE Internet Computing, July-August*, pages 38–46, 1998.

[13] Steve Lawrence and C. Lee Giles. Inquirus, The NECI Meta Search Engine. In *WWW7*, Brisbane, Australia, 1998.

[14] Steve Lawrence and C. Lee Giles. Searching the World Wide Web. *Science*, 280(5360):98, 1998.

[15] Steve Lawrence and C. Lee Giles. Accessibility of information on the web. *Nature*, 400(July 8):107–109, 1999.

[16] Michael L. Mauldin. Lycos: Design choices in an Internet search service. *IEEE Expert*, (January–February):8–11, 1997.

[17] Stefano Mizzaro. Relevance: The whole history. *Journal of the American Society for Information Science*, 48(9):810–832, 1997.

[18] Hien Nguyen and Peter Haddawy. The Decision-Theoretic Video Advisor. In *AAAI Workshop on Recommender Systems*, 1998.