Effective Resource Discovery on the World Wide Web

Evangelos P. Markatos, Christina Tziviskou, and Athanasios Papathanasiou

Institute of Computer Science (ICS),

Foundation for Research & Technology -- Hellas (FORTH), Crete

P.O.Box 1385 Heraklio, Crete, GR-711-10 GREECE markatos@ics.forth.gr

Abstract: The world-wide web grows at alarming rates, making information retrieval an increasingly difficult process. Traditional search methods based on search-engines usually flood the users with an overwhelming number of URLs. If a user wants to stay up-to-date on some issue and (s)he repeatedly queries the above search engines, (s)he will be repeatedly flooded with (almost) the same set of URLs, out of which only a small percentage will point to new, previously unseen documents. In this paper, we present USEwebNET, a resource discovery tool built on top of traditional search engines. USEwebNET registers each user's interests and repeatedly queries several search engines for URLs matching a user's registered interests. USEwebNET keeps track of which URLs have been visited by each user. Thus, when a user invokes USEwebNET, (s)he is presented only with new or "unvisited" URLs. We view USEwebNET as a value-added service on top of existing search engines and information providers, which helps users effectively find "what's new" in the rapidly evolving web of our world.

Introduction

Information retrieval on the Web is like searching for a needle in a haystack: one needs the right tools (like a metal detector) to separate the needle from the hay. In this paper we present a tool (called USEwebNET), which facilitates information discovery on the web. According to the **traditional model of searching** for information on the world-wide web, when a user wants to find information about a specific topic (s)he sends a query to a search engine, which replies with several URLs. Every time the user wants to find new information about the same topic, the search engine returns (roughly) the same URLs, flooding the user with unnecessary information. Finding the new (and interesting) URLs within a (slow moving) flood of previously visited URLs is a boring and time-consuming task. Assume, for example, that a user is interested in learning new developments in the field of "web caching". Searching for this information in a popular search engine (like HOTBOT) will return more than 2,000 URLs. Of those URLs, only a small percentage was created/modified within the last month. In this search model, users who are interested in getting only new information should process 2,000 URLs just to find the few recent ones. One could argue that recent information can be effectively retrieved by requesting only documents that have been published/changed only after a specific date (several search engines allow users to search for documents modified after some specific date). Unfortunately, this approach may still result in a document flood. Since the world-wide web is growing at alarming rates, the robots associated with search engines visit (and index) various sites rather infrequently. Popular web robots (like Alta-Vista for example) visit their archived sites periodically every 2-3 months. As the information available through the web grows larger, this interval is bound to increase (say up to 5-6 months). Thus, if a user wants to find new (previously unseen) information on a specific topic (s)he would have to search for documents that are 6 months old, or younger. Returning to our example, searching HOTBOT for documents on "web caching" that are 6 months old or younger resulted in more than 1,000 documents. Thus, searching for recent documents only, may still result in a document flood, which is exactly the problem we try to avoid.

The core of the search problem is that all these engines represent single-shot search mechanisms. A user that repeatedly searches using the same keywords is repeatedly flooded with almost the same URLs, even if (s)he has visited the URLs as a result of some previous search. This **single-shot search** is in contrast with the modern methods of knowledge discovery which are based on **re-search**. Research is an **iterative process** that filters away useless or already acquired knowledge, focusing on new and unexplored territory. USEwebNET is a layer of software on top of existing search engines, filtering away previously seen information, providing a service that allows users to stay informed of new developments.

Design

USEwebNET consists of a front-end that interacts with users and a back-end that interacts with search engines. Users register their interests with USEwebNET as a set of keyword-based queries. For example, people interested in caching mechanisms for the world-wide web may register their interest as "web caching". Along with that, people indicate several search engines which they would like to query about their interests.

Periodically (usually every night) the back-end of USEwebNET submits each query to the indicated search engines, which reply with a (usually long) sequence of URLs and a short description for each URL. USEwebNET gathers all replies, merges them, deletes duplicates and constructs a list with URLs that satisfy the query. It then removes from the list all URLs that have been previously viewed by the user.

When the user is interested in learning the recent developments on a given query, (s)he invokes USEwebNET, and (s)he is presented only with those URLs that have not been previously viewed. The user may decide to view a URL, or mark it as uninteresting and delete it. In both cases, USEwebNET will consider the URL as "viewed" and will not present it to the user anymore. Viewed documents will only be shown again to the user, if USEwebNET detects that an update to them has been made. To facilitate acquisition of knowledge, USEwebNET allows users to "save" URLs in folders. Thus, users can store all interesting URLs and visit them at some later time, or use them as references. Over time, folders will eventually become indispensable reference tools for research.

We view USEwebNET as a value-added service on top of existing information providers and search engines. The advantages of USEwebNET are:

- It **filters information** so that users are able to focus on "what is new" and thus, they are able to follow developments on their field of interest.
- It helps users **discover new information.**
- It **runs offline** every night when communication costs are low.
- It **offloads busy web servers** and proxies on the Internet (by running at night).

Implementation

In a nutshell, USEwebNET works as follows: Using USEwebNET's Setup interface, the user specifies his/her profile. This profile includes preferences relevant to the user queries, the desirable maximum age of the returned documents and the search engines that should be requested. A user profile is saved in several files located under the personal directory of the user. These files are used by USEwebNET's search utility to form the desirable queries for each search engine and forward to them the relevant requests. When the search utility receives the results, it filters them, discarding all those that have already been accessed by the interested user. Links to the remaining documents are saved locally, so that they can be accessed later. USEwebNET's search utility is written in C. Architecturally, it may be divided into two major components. The first one is responsible for creating the requests and filtering the returned results, while the second sends the requests to the various search engines and receives the results. The interface of USEwebNET has been implemented using several Cgi-Bin scripts written in the C programming language. These scripts are used for manipulating the user's preferences, dynamically creating HTML pages, which contain information that is not statically available, and verifying the ID of each user.

Example of use

In this section we will show an example of using USEwebNET. Suppose that the user is interested in finding information on “Distributed Systems”. These user's preferences are entered as shown in Figure 1.

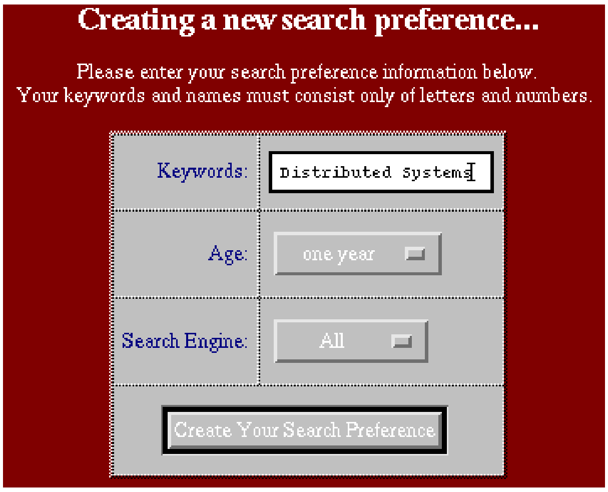
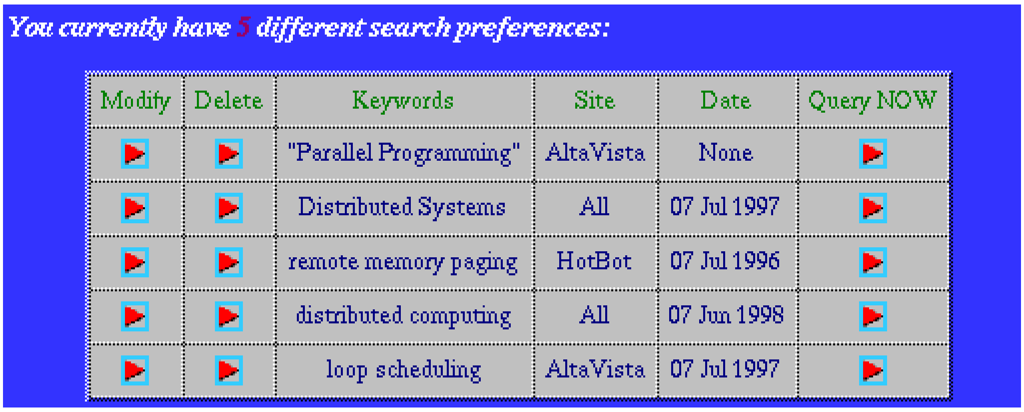


Figure 1: Entering a query on "Distributed Systems".

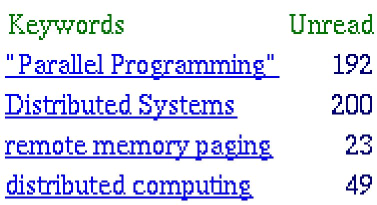


You currently have 5 different search preferences:

Modify	Delete	Keywords	Site	Date	Query NOW
▶	▶	"Parallel Programming"	AltaVista	None	▶
▶	▶	Distributed Systems	All	07 Jul 1997	▶
▶	▶	remote memory paging	HotBot	07 Jul 1996	▶
▶	▶	distributed computing	All	07 Jun 1998	▶
▶	▶	loop scheduling	AltaVista	07 Jul 1997	▶

Figure 2: USEwebNET displays the 6 queries that the user has submitted, the search engines that will be searched for each query as well as some other relevant information**.**

Once the user submits the preferences, the system will summarize them (as shown in Figure 2) and submit the queries to the selected search engines off-line (unless the user specifies that the search should start right away). After the search has been completed, the user may invoke USEwebNET which will display a screen like the one shown in Figure 3.



Keywords	Unread
"Parallel Programming"	192
Distributed Systems	200
remote memory paging	23
distributed computing	49

Figure 3: After all queries have been submitted to the appropriate engines, USEwebNET presents the results to the user in a summary form like the one shown. For each query the number of "still unread" URLs is shown.

The screen shot in Figure 3 informs the user that USEwebNET has found 192 URLs satisfying the query "Parallel Programming", 200 URLs satisfying the query "Distributed Systems", etc. The user may now "click" on the mentioned preference and a screen like the one shown in Figure 4 comes up. The user may now read/save these URLs. USEwebNET always shows the status of URLs as shown in Figure 5. The next time the user will invoke USEwebNET (s)he will not be shown the read/saved URLs again.

Title	Date	Read	Save
Parallel Programming with Fortran-90	17 Jan 1997	▶	▶
Parallel Programming Tools	31 Mar 1997	▶	▶
Introduction to Parallel Programming	09 Mar 1998	▶	▶
ETRI Parallel Programming Section	23 Feb 1998	▶	▶
Parallel Programming With Mpi	31 Mar 1998	▶	▶

Figure 4: Screenshot showing the results of a query in “Parallel Programming”.

Title	Date	Read	Save
Parallel Programming with Fortran-90	17 Jan 1997	✓	▶
Parallel Programming Tools	31 Mar 1997	✓	✓
Introduction to Parallel Programming	09 Mar 1998	✓	▶
ETRI Parallel Programming Section	23 Feb 1998	✓	✓
Parallel Programming With Mpi	31 Mar 1998	▶	▶

Figure 5: Screenshot of the results on “Parallel Programming”. The user has read 4 articles and has saved 2 of them.

Discussion

USEwebNET is a layer of software that runs on top of existing information providers and search engines, which helps the user discover and filter information on a specific subject. USEwebNET can be customized to work with specific databases in order to find information more effectively. In this section we look at two such examples: a research paper discovery tool on digital libraries, and a product discovery tool for electronic commerce.

Digital Libraries

Scientists always want to stay informed on their field. In order to do so they subscribe to scientific journals, go to conferences, collaborate with colleagues, etc. To narrow down the information they receive, scientists subscribe only to a small subset of journals and follow only a small number of conferences. Unfortunately, the number of scientific publications increases year after year making it increasingly harder for a single person to keep track of all (or even most) published papers on a field. Thus, a tool that could deliver to scientists only the interesting research papers that fall within their field would be very useful.

Fortunately, USEwebNET meets these requirements and comes very close to this ideal tool. Currently, most of the scientific publishers provide on-line databases with the titles, authors, abstracts, and sometimes even full text of their publications. USEwebNET can be used as a research paper discovery tool on top of several such databases. For example, a scientist may submit to USEwebNET that (s)he is interested in "web caching". USEwebNET will continually monitor the research paper databases to find papers that match the query. If such matches are found they are stored in a database. When the user invokes USEwebNET, (s)he will view the new papers found. After the user "reads" these papers they will not show up again, unless the user specifically saves them in a folder. Effectively, the user registers his(her) interests with USEwebNET, and the tool continually delivers new research papers found on this field without delivering the same paper twice.

Electronic Commerce

It is not uncommon for people to search for specific items over a long period of time, either because these items are rare, or because they are not affordable. For example, a person may want to purchase a particular model of a particular car at a particular price. USEwebNET may help in this search by searching databases of news, classified ads, etc. in order to find a car that matches a user's needs. Once such a car has been found, it is stored in a database. When the user invokes USEwebNET (s)he will be alerted to the existence of the new ad on the particular car.

Extensions

Currently most users enclose all the information about the field they would like to explore into a few keywords. For example, suppose that a person is interested in caching mechanisms in order to reduce the latency of the world-wide web. To find available on-line documentation, the person may submit a query like "web caching". Although several related documents will be returned, not all the relevant documents contain the "web caching" substring. They may contain the "www caching" or the "caching in the web" substrings or none at all. Choosing the right keywords is a tedious and difficult task, even for experts. We plan to extend USEwebNET's capabilities with approximate searching based on a set of "seed" documents. That is, the user presents some "seed" documents that are representative of a given field, and asks USEwebNET to find "similar" documents. We envision several diverse definitions of similarity. One could consider "similar" documents, those documents having common co-authors, or documents sharing a large number of keywords, or documents referencing a large number of common publications, or documents sharing a large number of common URLs, etc. The initial set of "seed documents" may even be formed after a search for "web caching". After the initial set is formed, it can be augmented/changed as more relevant documents are discovered. In this way, the user will be able to find "similar" documents that do not contain a given substring, but they are semantically close to a defining set of web caching documents.

Previous Work

Finding information in cyberspace is a difficult but exciting field of research and development. Even before the dawn of the world-wide web, several tools that helped users find information available somewhere in the network had already appeared. For example, ARCHIE[1] and Veronica[2] are tools that help users find files available via ftp, and via gopher respectively. Netfind [Scwartz 91] was a first approach to a «yellow pages» directory. Netfind help users locate phone numbers and e-mail addresses of people that have an account on a computer on the Internet.

With the advent of the world-wide web, and the significant amounts of information that became accessible, several search engines that index (a significant portion of) the web have appeared. Such engines include Alta-Vista[3], Excite[4], etc. Some search engines even provide a "meta-search" capability, that is the ability to submit a query to several search engines at once. For example, PROFUSION [Gauch 96] sends user queries to multiple underlying search engines in parallel, retrieves, and merges the resulting URLs. Although these engines are a significant step towards information discovery in the cyberspace, they provide a single-shot search mechanism that is in direct contrast with the re-search mechanism that scientists have been used for hundreds of years. Current searches are memory-less, in the sense that they do not separate between new URLs from previously presented ones. Thus, every time a user supplies a query, (s)he is flooded with (almost) the same URLs. To reduce the number of URLs returned, several of these search engines provide a time-limit parameter that allows the user to search for URLs that appeared (or have changed) in the last (say) 2 weeks. Unfortunately, this ability does not always stand to its name. Currently, search engines visit the indexed servers once every 2-3 months. This implies that from the time a URL is published on the network till the time it appears in the database of a search engine a period of 2-3 months may easily pass. Thus, in order to get new URLs, a user needs to search for URLs dated at least 2-3 months back. As the World Wide Web grows larger, search engines will probably visit their indexed servers even more infrequently, and the problem will get even worse. Thus, the time-limit parameter provided by search engines will practically increase to a granularity of several months.

SenseMaker [Baldonado 97] is a tool that helps users find the information they are looking for in the world wide web. SenseMaker organizes the results of queries into information contexts based on the domain that the URL is

---

[1] http://www.nexor.com/archie.html

[2] gopher://veronica.scs.unr.edu/111/veronica

[3] http://altavista.digital.com

[4] http://www.excite.com

located or on some other aspect. SenseMaker allows users to expand and refine their queries based on the information they acquire during this research process. We view SenseMaker as complimentary to USEwebNET. Although SenseMaker focuses on guiding users during a research session, USEwebNET focuses on keeping users up-to-date on a given topic over a long period of time.

The Informant[5] allows users to register a number of queries which are then sent to a popular search engine. The system keeps track of the top 10 URLs that are returned for each query. When the set of the top 10 URLs changes, the system notifies the user by sending an email. Although the Informant sounds similar to USEwebNET, there are three main differences:

- Several URLs that may be of interest may never make it to the top 10 URLs and thus will never be seen by the users. Although the threshold of 10 may easily change to become 100, or even 1000, such changes would severely challenge the scalability of Informant.
- Informant alerts the user about changes to URLs using email. We believe that most users either ignore or store-for-later such notification email messages. On the contrary, our approach does not use notifications. Instead, when the user would like to know if there are any news with respect to a topic, (s)he invokes USEwebNET. We have borrowed this interface and style of access from the USENET news, which have been used successfully by millions of users for more than a decade.
- USEwebNET can exploit database specific knowledge to improve the results of a registered query. For example, if a user is looking for papers on a particular subject from a digital library, USEwebNET may use additional information provided by the paper (like authors, or references), in order to make a more effective search.

Summarizing, USEwebNET: (1) capitalizes on the familiar and effective user-interface of USENET news, (2) exploits search engines to find out what is new in the World-Wide Web, and (3) reduces information pollution by not repeating previously explored URLs.

Summary

In this paper we tackle the problem of effective resource discovery on the world-wide web. We present the design and a prototype implementation of USEwebNET, a tool that helps users effectively find the information they need. We believe that as the Internet grows larger and as more people depend on finding critical information available on the network, tools that help users in resource discovering and reduce information overloading (like USEwebNET) will be increasingly important.

Acknowledgements

This work was supported in part by PENED project (2041 2270/1-2-95) funded by the General Secretariat of Research and Technology, and in part by the USENIX Association. We deeply acknowledge this financial support.

References

[Baldonado 97]: Baldonado, M.Q.W. & Winograd, T. (1997). SenseMaker: An Information-Exploration Interface Supporting the Contextual Evolution of a User's Interests. Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI '97), Atlanta, Georgia, April 1997, pp. 11-18.

[Gauch 96]: Gauch S., & Wang G.(1996). Information Fusion with ProFusion. Proceedings of WebNet 96.

[Schwartz 91]: Michael F. Schwartz, & Panagiotis G. Tsirigotis. Experience with a Semantically Cognizant Internet White Pages Directory Tool. Journal of Internetworking Research and Experience, pages 23-50, March 1991.

[5] http://informant.dartmouth.edu