PARTICIPATORY SEARCH

G. Attardi, A. Esuli, L. Hancu, M. Simi
Dipartimento di Informatica, Universita di Pisa
{attardi,esuli,hancu,simi}@di.unipi.it

ABSTRACT

Participatory search can be defined as a search service framework where each participant takes care of gathering information and contributing indexes to a central location where the indexes are integrated and deployed for running a search service. Participatory search has distinguishing characteristics with respect to other search service architectures, among these the ability to share with others material that is not available to traditional search engine crawlers. We present the architecture for participatory search and a protocol for index exchange that we are developing. A deployment niche for participatory search is the implementation of search services for community networks.

KEYWORDS

Search services, crawler, distributed systems, search indexes, participatory search.

1. INTRODUCTION

A search service consists in three distinct components:

- The crawler, which collects Web contents
- The indexer, which builds the optimized data structures to perform search efficiently.
- The search engine, which computes answers to users' queries.

Search services can be built using different architectures, ranging from completely centralized to completely distributed, as in the peer-to-peer approach.

The different approaches to the implementation of a search service can be compared along several dimensions: the amount of network traffic generated by crawlers; the ability to keep search indexes up to date; search efficiency; the coverage of the target content they are able to achieve. In fact a large portion of the Web, the so called *Hidden Web*, is stored in databases; it is accessed through search forms and rendered throughout Web pages which are dynamically generated. The Hidden Web is estimated to contain 500 times more documents than the Publicly Indexable Web (PIW), i.e. 7500 TB of data, grows much faster and consists of high quality, topic-specific information [Bergman01]. The ability to acquire and search material from the Hidden Web is deemed to become an important issue in the evaluation of search services.

Participatory search can be defined as a search service framework where each participant takes care of gathering information and contributing indexes to a central location where the indexes are integrated and deployed for running a search service.

In this paper we outline the characteristics of a participatory approach to the implementation of a search service with respect to other approaches, argue for several advantages, present the architecture of a participatory search service and a protocol for index exchange that we have implemented, discuss the issue of the hidden Web and possible scenarios for the deployment of participatory search services.

Large scale indexes and web repositories are not available to research institutions willing to investigate on enhancements of web search technologies. In the context of the ECD (Enhanced Contents Delivery) [ECD] project, we are involved in building a large collection of Web material as an essential step for further Web content analysis. Other similar efforts were done by the University of Stanford in building their WebBase [Hirai99] and the Polytechnic University of New York in building their Polybot crawler [Shkapenyuk02].

Commercial web services are built on proprietary technologies and ranking is often not as transparent as one would like. In fact, there is little literature on the subject. An open-source crawler was developed for

Nutch, a Web search engine written in Java as an effort to provide a transparent alternative to commercial Web search engines [Nutch].

The technology we are developing for participatory search has the potential of creating a new way to produce, share, exchange and collect indexes according to an open format, thus contributing to a larger availability of a valuable collective resource for experimentation, research and implementation of low cost community services with transparent ranking criteria.

2. SEARCH SERVICES APPROACHES

2.1 Centralized Model

Traditional Web search engines collect data from publicly accessible Web pages by performing large scale crawling from a central location. In a few cases they also retrieve private documents (e.g. newspaper articles accessible only to subscribers), through specific arrangements with information providers.

In the centralized model all the components (the crawler, the indexer and the search engine) are implemented in a central location and are managed by the provider of the service.

Major commercial search engines (Yahoo, Google, Lycos …) are capable of answering over 5000 queries in less than one second, searching on an index of over 5 billion Web documents [Google]. In order to reach such performance and coverage of the Web they need to deploy a huge quantity of computational power and resources.

It is currently impossible to match the performance of a large scale centralized search system with a distributed or peer-to-peer approach [NutchFAQ]. Maintaining search indexes up-to-date as Web contents change however, is a challenging task for a large scale centralized search service. The crawler has to make a guess about which Web pages are frequently updated and regularly revisit them, in order to maintain the freshness of the local copies. Re-crawling strategies are usually based on heuristics, taking into account the modification frequency of documents and the site popularity [Chakrabarti03].

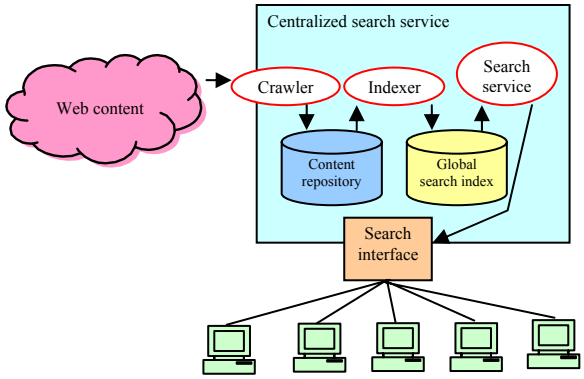

Figure 1. Centralized search approach

Despite the large coverage of the PIW, and the data acquired through specific agreements with content providers, commercial search systems are not able to reach the largest portion of the Web - the dynamically generated Web pages whose content is stored in large databases. These pages are publicly available to humans, but unreachable to programs, as gathering them requires filling out forms with data difficult to generate with automatic tools.

2.2 Collaborative Models

A number of services (like Google News [GoogleNews] or Froogle [Froogle]) are able to extract private information that is hidden behind registration forms, by establishing specific agreements with content providers. We call this model collaborative search, as partnership between the two parts is required. The collaboration works as follows: the content provider assigns login information to the information gatherer, which collects the Web pages after being authorized by the Web server. The gatherer indexes the information and makes it available to user queries, but disallows the user's access to the indexed data.

Grub is a distributed crawler which is run on local machines as a result of joining a collaborative network [Grub]. Grub's clients are instances of a distributed crawler managed by a central system, which maintains

control over the crawling process. This again can be regarded as a collaborative approach, where the collaboration consists in offering computing resources for the crawling process.

The control of the crawling process and the decision of which material to index and when is strictly in the hands of a central coordinator rather than the participants. Indexing and search is still performed centrally.

2.3 Peer-to-Peer Models

The Web is participatory by nature. As shown by the success of peer-to-peer applications and, more recently, RSS syndication and RSS news aggregators, contents providers are willing, and even eager, to share their contents and to make them available to other Web users.

Most of P2P search applications are dedicated to files sharing services (e.g.: KaZaA [KaZaA], Gnutella [Gnutella]). In these applications the contents usually consist of multimedia data (i.e.: MP3 audio, MPEG video) and search is done using meta-information about them (e.g.: song title, author, movie title).

In the P2P model all the three components are replicated on each node of the network; each P2P participant, willing to make its contents searchable, publishes a search interface. Typically there is no need for a central coordination, thus the global P2P search service does not require additional resources for its deployment, but uses little resources from each client.

In this model the crawling component has local access to documents and therefore the ability to manage access rights and access speed and has knowledge about content variation. The crawler can receive a real-time notification of creation, modification and deletion of a document and can maintain the index synchronized with these updates.

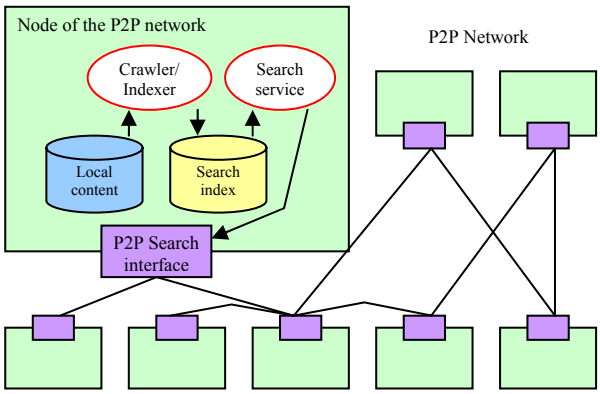

Figure 2. Peer-to-Peer approach

The search component is the most complex of the three in this model, because each node has to coordinate with the others to build a search network that answers to users' queries. A search query from a querying node is propagated from node to node on the entire network and then the answer of each node is sent back to the querying node to build the search result. The time to answer a query is obviously much higher than in the centralized search and often results are incomplete due to the unavailability of some nodes in the network.

In this model the crawling phase is efficient in the use of network resources while the search functionality can generate a conspicuous quantity of network traffic.

2.4 Participatory Model

The Participatory Search model combines the benefits of a centralized search index, and the advantages of a distributed approach in the collection of the material [Attardi03b].

The content providers themselves participate in creating the centralized index and are responsible of keeping the central index up-to-date according to the policy they decide. As participants in a Participatory Search System, they contribute with their own computing resources to collect documents from their local repositories, index them and transmit the index for integration in the overall collection index. They are required to install and configure a version of the Participatory Search System crawler/indexer to connect to the databases or repositories and to extract and index the relevant material.

A high performance search service is still possible, since the index is maintained in one location. At the same time, network bandwidth is saved in the crawling stage and there are better opportunities for solving the freshness problem and reaching the hidden web.

The advantages for the participants are:

- Complete control on the material they contribute to the central index and its freshness.
- The locally generated index can also be used to implement a local search service.

- A participatory architecture is best suited for extracting contents from the Hidden Web.

Instead of waiting for crawlers to come and visit and hoping that they index the right things with only the limited control that robot files provide, the members of a participatory search network have the possibility to decide which documents and data to make available to the general public, thus hiding reserved or private material or documents (and data) which are relevant only to a local audience. They may also decide to index and make searchable data which is hidden in databases.

Another advantage is the control over how often the indexes are to be refreshed depending on the dynamicity of the collections. Index updating can be done, as we will explain, with a differential approach, saving still more bandwidth.

A potential disadvantage of the participatory approach is the expected limit in the coverage of the Web; in fact, the degree of coverage strictly depends on the willingness of the participants to contribute with target material. On the other hand, a participatory approach for extracting information from the Hidden Web shows potential advantages, as we will argue next.

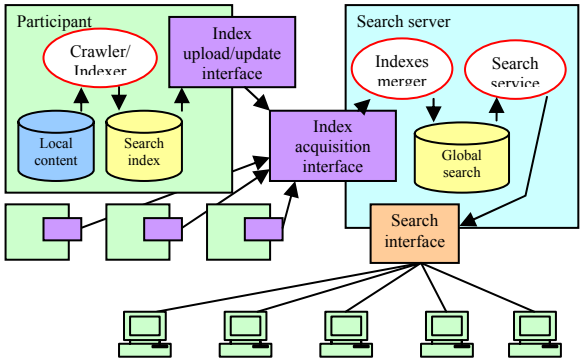

Figure 3. Participatory search approach

A participatory search system differs from a collaborative one, in the fact that more control is left to the content provider. This one, joining a participatory network, must only agree on using a common protocol for index exchange, while retaining decisions about the frequency and times of the updates and which material to contribute.

The following table summarizes the main differences among the four search models we have discussed above.

Table 1. Summary of search models features

Model	Crawler	Indexer	Search engine
Centralized	*centralized*	*centralized*	*centralized*
Collaborative	*centralized | distributed*	*centralized*	*centralized*
Participatory	***distributed***	***distributed***	***centralized***
Peer-to-peer	*distributed*	*distributed*	*distributed*

3 THE PARTICIPATORY SEARCH ARCHITECTURE

A participatory search system consists of a number of clients and a central coordinator. Each client periodically produces an index of the content collected from local resources (e.g.: local Web server, databases, FTP server, shared file system, news server, etc.) and sends it to the coordinator. The index produced by a client is a data structure designed to optimize full text search; its format is part of the ANTIX protocol defined in detail in section 3.3. The coordinator is responsible for gathering indexes from all clients and uses them to provide a global search service for the participatory network. Clients and coordinator are designed to minimize the amount of data they have to exchange, to maintain the search service up-to-date as the content on clients changes.

3.1 Client

A client in a participatory search system is responsible for gathering the content that has to be published on the search service, for indexing it and sending the index to the coordinator.

Gathering of content to be indexed is performed by a *collector* module. The collector can be configured to run a periodic crawling of all the local resources, to gather and index all the content that is changed (added, modified or deleted) from the previous crawling session. The resulting index is differential index that can be

merged with the previous index to obtain a new, updated, index of the gathered content (more on this below, in the section about the ANTIX protocol). A smart collector will use local resource facilities (e.g.: file system watches, database triggers) to receive immediate notice of content modification, activating indexing of modified content after a certain number of modifications have been registered. The client uses different specialized collectors to be able to gather content from various sources. For example, a collector specialized for Hidden Web resources [Hancu02] is configurable to automatically generate instances of parametric URLs (with parameter values extracted from a database) for the generation of the dynamic Web pages to be indexed.

The indexing module of the client produces an index in the format required by the ANTIX protocol. The resulting index is sent to the coordinator using the ANTIX protocol and can also be used by the client itself to provide a search service on local resources.

If the participant is willing to contribute to several participatory search services, the client can be configured to produce more than one index (e.g.: one from a RSS feed, one from an online catalog) and to publish them on different coordinator servers.

3.2 Coordinator

The coordinator's main task is to manage the collection of indexes from all clients and to produce a global search index to be used in the participatory search services.

The coordinator is always waiting for clients to connect and transfer an index update. To avoid a connection overload it associates to each client's index a minimum time interval between updates: the coordinator has the ability to deny an update of an index in between, but exceptions are possible (e.g.: if no one else is updating, an incoming index can be accepted disregarding its policy about the minimum time interval). Minimum time intervals can vary with content type (e.g.: a content provider will likely update its news indexes more frequently than an index of its library catalog).

The coordinator guarantees the consistency of the global search index with the indexes sent by clients and also with the content the indexes point to. Each index has associated a password that the client must provide to be authorized by the coordinator to upload it. This security measure prevents a malicious client from uploading a fake index and providing some damage to another client. The coordinator also performs a check on a random small part of the uploaded index to verify that its data correspond to the content it points to. This helps avoiding that a client, for example, produces an index with additional keywords, unrelated to content, to increase the probability of being returned in the search results. A third check is performed on the index structure to avoid that a bad formed index will damage the global index, when merged into it. If an index passes all controls it can be integrated into the global index.

The search service can be published using various interfaces like HTML forms, for human access, and Web services, for software access.

The indexing and search technology used for building our prototypes is IXE (IndeXing Engine), a C++ library for implementing high performance full text search services [IXE]. IXE provides full support for persistent C++ objects. Implementing object persistence requires knowledge about the object structure in order to serialize and de-serialize objects on a storage support. The library exploits C++ template meta-programming techniques to provide reflection capabilities, allowing the programmer to specify meta-information about the objects and their implementation, in particular supplying annotation about persistence [Attardi01].

3.3 The ANTIX Protocol

Client and coordinator communicate by using ANTIX, a communication protocol that defines the index format and the way client and coordinator interact during the index upload (or update) process. The ANTIX protocol was designed in the context of the ECD project; a preliminary draft of its specification is available as internal report [Attardi03c]. The ANTIX protocol has been designed to be implemented using Web services interfaces.

A client that wants to update its index on the server initiates an ANTIX session. The first step is the authentication of the client with the coordinator, which returns a session token, to be used in the session for any successive communication. If the server is too busy, or the client is updating its index too frequently, the

coordinator can send a "delay" message to the client, requesting it to wait a minimum amount of time before reconnecting to the server.

When the client has been authenticated, client and coordinator negotiate the type of transfer to be performed: upload or update. Upload consists in transferring a brand new index of the whole collection maintained by the client; update consists in transferring an index of only the modified part of the content, since previous indexing. Performing update instead of upload enables saving a lot of data transfer and is possible when both client and coordinator maintain a synchronized copy of the previous indexing.

The client sends data to the coordinator, respecting the index data format specified by the protocol: first a list of description of indexed documents (e.g.: URL, title, summary, etc.), then the list of URLs of deleted documents, if any, and finally the core index structure consisting of lexicon and postings lists.

The session ends when the index transfer completes. If some error occurs during transfer the coordinator asks the client to resend a part or, if the error is unrecoverable, aborts the session and discards all data transmitted during the session.

4 HIDDEN WEB

4.1.1 Surface and Hidden Web

Traditional search engines create their indices by crawling surface Web pages. The only way to collect URLs is to scan collected pages for hyperlinks to other pages that have not been collected yet [Bergman01]. To be discovered, the page must be static and linked to other pages.

Web pages currently classified as Hidden Web [Hancu02] (also known as the *deep* Web [Bergman01] or the *invisible* Web [Lin02]) exhibit two fundamental properties: are dynamically generated and require user input. A typical example of hidden Web pages are the ones obtained as response to a query that a user performs, throughout a form, on an online searchable database,

Black-box model approaches consider the gatherer as an external tool which does not have direct access to data repositories and to the scripting code which generates Hidden Web pages. By means of form analysis tools, by discovering common filling patterns and using heuristics, they are able, to some degree, to gather pages hidden behind search forms ([Raghavan01], [Lage02]).

4.1.2 Participatory Approach to the Hidden Web

In contrast with the black-box model approach, a participatory approach to the Hidden Web has the advantage of unrestricted access to the databases from which the Web pages are generated and to the source code of the scripts which generate them.

It is then possible to provide the administrator with semi-automatic tools to specify input parameters for the URLs of the dynamic pages that have to be indexed; the system relies on the information from the database to instantiate those input parameters and materialize Web pages ([Hancu02], [Attardi03a]).

As an alternative, we are exploring program analysis techniques on the source code of the scripts which generate the Web pages in order to derive dependencies between Web page's input parameters and columns from the data repositories. Once these dependences are extracted, one could simply collect all the possible values for each input parameter (a finite set of values) and generate all the possible Web pages by instantiating the parameters with those values.

5 SCENARIOS FOR PARTICIPATORY SEARCH

There are applications where the participatory approach to the implementation of search services is especially suited. As examples, we will discuss an application scenario for the deployment of participatory search.

5.1 Search Service for a University Network

A general search service has some limits for the use by a university or research network. This is an instance of a large community with special interests: researchers need to use a search service for documentation and would like to retrieve documents which are related to their job without having to dive in the large number of results which are typical of large scale Web search engines.

A slightly different issue are thematic communities, who might desire a search engine specializing on a specific subject area. This goal can also be achieved to some extent with focused crawling, exploiting classification techniques. In the case of a university network the main issue is not one of focusing on the appropriate material, but rather one of quality of information and authoritativeness of the source. Moreover, one would like to make available to a restricted community documents that would not be normally indexed by a search engine or are not targeted to the general public, such as interim project reports or course material.

In similar scenarios, participatory search is a viable and convenient alternative to implement a full-text search service. The cost of implementing and running a service implemented according to the centralized model would be too high and the service not completely satisfactory.

There are several levels in which participatory services could be deployed:

- *Department level*: researchers could make available their papers and the project reports they want to make searchable by the community, by storing them in a shared disc space and configuring the client.
- *University level*: indexes could be collected by a coordinator run at the level of the university and used to deploy a search service local to the university.
- *Network of universities* or research institutions: each university coordinator, acting as a client of a larger network, can contribute its indexes to a network coordinator thus allowing the deployment of a wider scale search service.

Given that a search client in a participatory network can index different collections and register with the coordinator for different search services, it could be possible with the same technology and low additional cost to contribute to a search service over administrative documents or course descriptions, if one wanted to build a search service over the course offerings of the university or network of universities.

6 CONCLUSION

We have described a participatory approach to the development of search services which differs from the way major search services are implemented: indexes are built by participants and sent to a central coordinator according to a suitably designed communication protocol. Advantages of this approach are a lower network overload (crawling is local and index upgrading is incremental); the decision of which contents are to be contributed and how often is in the hands of the participants and freshness can be achieved with the right upload strategy; contents which are part of the hidden Web have a chance to be indexed as well.

The technology we are developing for participatory search is also defining an open format for indexes, which, within a participatory framework, are created, exchanged, communicated and integrated according to a common protocol. We hope that this work will contribute to the creation of collective resources for experimentation and research, as well as implementation of low cost search services for user communities.

ACKNOWLEDGEMENT

This work has been performed in the framework of the ECD (Enhanced Content Delivery) project [ECD] and partially supported by MIUR. We thank all the participants in the Action 2 of the project for fruitful discussions and in particular Antonio Cisternino for technical advice and contribution of ideas.

REFERENCES

[Attardi01] G. Attardi, A. Cisternino, 2001, *Template Metaprogramming an Object Interface to Relational Tables*, Reflection 2001, LNCS 2192, 266-267, Springer-Verlag, Berlin.

[Attardi03a] G. Attardi, A. Esuli, M. Simi, 2003, *Sistema di Best Bets: descrizione del prototipo e sperimentazione*, Deliverable ClickWorld BB-3.

[Attardi03b] G. Attardi, A. Esuli, L. Hancu, M. Simi, 2003, *A Participatory Search System's Architecture*, *Enhanced Content Delivery Project – Technical Report*.

[Attardi03c] G. Attardi, A. Esuli, L. Hancu, 2003, *ANTIX: A Protocol for Gathering Full-Text Indices*, ECD Technical Report.

[Bergman01] M.K. Bergman, 2001, *The Deep Web: Surfacing Hidden Value*, White Paper, http://www.press.umich.edu/jep/07-01/bergman.html

[Chakrabarti03] S. Chakrabarti, 2003, *Mining The Web: Analysis of Hypertext and Semi Structured Data*, Morgan Kaufmann Publishers, Elsevier Science, http://www.cs.berkeley.edu/~soumen/mining-the-web/

[ECD] Technologies and Services for Enhanced Content Delivery, http://www-ecd.isti.cnr.it

[Froogle] *Froogle*, http://froogle.google.com/

[Gnutella] *Gnutella*, http://gnutella.wego.com

[Google] *Google*, http://google.com

[GoogleNews] *Google News*, http://news.google.com/

[Grub] *Grub*, http://www.grub.org/

[Hancu02] L. Hancu, 2002, *Discovering Hidden Web Content*, Graduation Paper, "Babeş-Bolyai" University, Cluj Napoca, http://www.di.unipi.it/~hancu/HiddenWeb/

[Heydon99] A. Heydon, M. Najork, 1999, *Mercator: A scalable, extensible Web crawler*, *World Wide Web Conference*, 2(4), pages 219-229.

[Hirai99] J. Hirai, S. Raghavan, H. Garcia-Molina, A. Paepcke, 1999, *WebBase: A repository of web pages*, Computer Networks, Amsterdam, Netherlands.

[IXE] IXE Library Architecture, 2001, *Documentazione tecnica Ideare SpA*, http://www.ideare.it

[KaZaA] *KaZaA*, http://www.kazaa.com/

[Kalogeraki02] V. Kalogeraki, D. Gunopulos, D. Zeinalipour-Yazti, 2002, *A local search mechanism for Peer-to-Peer Networks*, http://www.cs.ucr.edu/~csyiazti/downloads/papers/cikm02/cikm02.pdf

[Lage02] J.P. Lage, P.B. Golgher, A.S. da Silva, A.H.F. Laender, 2002, *Collecting Hidden Web Pages for Data Extraction*, The 4th International Workshop on Web Information and Data Management, McLean, USA.

[Lin02] K.I. Lin, H. Chen, 2002, *Automatic Information Discovery From The "Invisible Web"*, International Conference on Information Technology: Coding and Computing, Nevada, USA.

[Loo03] J. Li, B.T.Loo, J.M. Hellerstein, M.F.Kaashoek, D.R.Karger, and R. Morris, 2003, *On the Feasibility of Peer-to-Peer Web Indexing and Search*, 2nd International Workshop on Peer-to-Peer systems, http://iptps03.cs.berkeley.edu/final-papers/search_feasibility.ps

[Nutch] *Nutch*, http://www.nutch.org/

[NutchFAQ] *Nutch Frequently Asked Questions*, http://www.nutch.org/docs/en/faq.html

[Raghavan01] S. Raghavan, H. Garcia-Molina, 2001, *Crawling the Hidden Web*, Proceedings of the 27th Conference on Very Large Databases, Rome, Italy.

[Shkapenyuk02] V Shkapenyuk, T. Suel, 2002, *Design and Implementation of a High-Performance Distributed Web Crawler*, ICDE.