ADVANCED DISTRIBUTED SEARCH FOR THE WEB

Rinat Khoussainov, Tadhg O'Meara, and Ahmed Patel
Computer Networks and Distributed Systems Research Group,
Department of Computer Science, University College Dublin,
Belfield, Dublin 4, Ireland

ABSTRACT

In this paper, we present the concept of, and discuss problems related to, distributed search architectures for the World Wide Web. We structure the problem area and analyse what aspects have already been covered by previous research and what needs to be done. We outline possible approaches to some of the important research issues in distributed search architectures and discuss them in the context of the ADSA (Adaptive Distributed Search and Advertising) project.
Keywords: Distributed Search Architecture, Decentralised Service Management, Topic-specific Search Engine, Topic-specific Web Robot.

1. INTRODUCTION

At present, the Internet provides access to an enormous amount of information resources of various types including text, software, video and audio information. The rate at which new information resources appear on the network has far exceeded the ability of any single organisation to track and review new resources as they appear. Therefore, effective means for discovery and retrieval of information resources of interest have become vital to Internet users.

Information resource discovery is used to describe a complex collection of activities [1]. At the highest level, it involves searching various directories, catalogues or other databases that contain descriptions of information resources and the selection of the resources most relevant to the user's query. At a lower level, the duties of Resource Discovery Systems (RDS) include the initial identification of resources, the collection of information about the resources, the storage of these resource descriptions in a form suitable for efficient searching,

and the actual processing of user search queries. As the availability and characteristics of a resource on the Internet can change over time, an RDS must also regularly update resource descriptions stored in the database. One of the grand challenges for the design of information RDSs is scalability. By scalability we mean the ability of the system to increase its processing capacity by making quantitative rather than qualitative changes to the system (that is by adding new computing and storage resources rather than changing the system architecture).

RDSs for the information available on the World Wide Web (WWW) are traditionally called Web search engines [2]. The design of effective RDSs for the Web has become a challenging task and a focus of much research effort. The huge size and continuing growth of the Web, combined with its high popularity among Internet users, has made it difficult for existing Web search engines to provide complete, relevant, and up-to-date information in response to user search queries. A frequently cited survey estimated that in 1999 the 11 largest search engines combined only indexed 42% of public Web pages [3]. It also found that the larger databases tend to have the most out-of-date entries.

The source of much of these problems can be attributed to the use of centralised search architectures. Centralised search architectures, as employed by AltaVista, FAST, Google, and Inktomi, use many computers under centralised management to act as a single search engine [4]. While these approaches scale to large numbers of computers, and can process large numbers of Web pages and user queries, the scaling techniques they use require centralised access to, and control of, the underlying database containing descriptions of Web pages. This results in the following problems:

- *Cost*: The network and hardware resources required to download, index, and search all publicly accessible documents on the Web by a centralised search system is prohibitively expensive for any single organisation.
- *Minority users*: Given the incomplete coverage of the Web by the centralised search architectures, the search service providers typically focus on the most popular Web sites [3]. Because the control over what documents are indexed is concentrated within a small number of search service providers, there is no guarantee that pages relevant to "minority" user interests will be indexed.
- "*Invisible Web*": The centralised databases are built using automated techniques that follow hyper-links in Web documents. There are a large number of document collections on the Web,

however, that are not accessible using this method, such as non-Web document collections having a Web front-end [5]. The same problem exists when Web publishers are wish to protect their intellectual property by making their content accessible only to paid subscribers. These information resources are referred to collectively as the "invisible" (or "hidden", "deep") Web, and it is estimated that the amount of information in the deep Web is many times larger than in the visible Web [5].

The solution to these problems we believe lies in the use of *Distributed Search Architectures* (DSA) [6]. In distributed search architectures multiple search engines owned by different organisations or individuals act as a single search system. The major goals of this paper are:

- to provide an introduction to DSA for the Web;
- to describe the research problems associated with DSA;
- to analyse existing research in the area and identify outstanding issues;
- to propose possible approaches to some of these problems.

2. BACKGROUND

All types of search architectures share a number of common activities and components. In this section, we briefly review these common elements and the related terminology. The activities performed by a search engine can be divided into the following basic groups:

- *Storage of Web resource descriptions*. As with any RDS, a Web search engine needs to collect information about Web pages and store it in a form suitable for efficient processing. The element of a search engine responsible for storage is usually called the *document index* or *document database*. The process of adding/removing database entries is called *indexing*.
- *Discovery of Web resources*. Discovery of Web pages can be performed manually or automatically. Automatic discovery is performed by first downloading a specified set of Web pages and then recursively downloading pages linked from the originals. The process of automatic discovery is called *crawling*, and the component responsible for it is called a *Web crawler* or *Web robot*.

- *Processing of user search queries*. This is the main activity in a search engine. It includes receiving a query describing the Web resources that a user is interested in, searching the database for relevant entries, and presenting the results to the user. Results may be ranked by their relevance to the query or other properties.

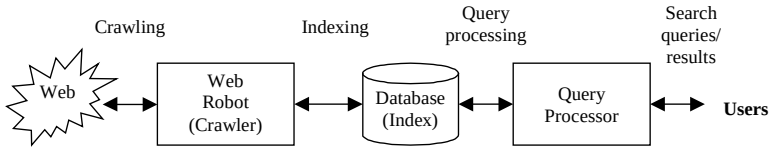


Figure 1: Elements of a search engine

Figure 1 illustrates these activities and the corresponding elements of a search engine.

3. A DISTRIBUTED SEARCH MODEL

All modern large search systems employ some sort of distribution to increase their capacity. For example, the AltaVista search engine uses separate networked machines to perform different functions such as crawling, indexing, and query processing [7]. Google, Inktomi, and Fast are using large computer clusters where given functions, such as crawling or indexing, can be performed by many computers in parallel [4].

Our classification of search architectures, however, distinguishes between centralised and distributed architectures by the distribution of control and management. *Centralised search architectures* are those using one or more computers to act as a single search engine under centralised management. The organisation owning the search engine has entire control over all processes inside the search engine (including query processing, indexing, and population) and access to the entire search engine index. Each centralised search engine aims to index all Web pages and to process all user queries.

In *distributed search architectures*, multiple search engines owned and controlled by different organisations or individuals act as a single search system. Each search engine in the system chooses to index only a subset of all Web pages and processes only a subset of all user queries. To allow multiple search engines to function as a single search system, two additional activities are required:

1) Sending user search queries to the appropriate search engines. This process is called *query routing* or *database selection.*
2) Aggregating search results returned from multiple engines and presenting them to the user. This process is called *result aggregation* or *merging*.

Distributed search architectures will work effectively only if most user queries have to be sent to a small subset of the available search engines. When this is not the case, engines quickly become overloaded, as each engine must process all queries. Avoiding this requires an appropriate distribution of Web pages between search engines such that, for most user queries, the relevant resources are indexed by a small number of the available engines.

In human terms, we call the property that unites relevant documents their *topic*. Given the topic-specific nature of user search requests, a topic-oriented distribution of Web pages between search engines will mean that for most queries, the relevant documents should be indexed by a small number of the available engines, thus allowing for efficient processing of the queries.

In an RDS consisting of multiple topic-specific search engines, each engine collects information and processes user queries only on a specific topic. User requests are only routed to those search engines that have (or are likely to have) information relevant to the query's topic. In such a system, independent search engines can both co-operate and compete with other engines. Separate engines can co-operate by relying on other engines in the system to process queries that they cannot satisfy. Engines can compete for user queries by choosing different topic selection, and other service parameters, such as service price.

This type of topic-specific distributed search architecture can address all of the problems of the centralised architectures identified in Section 1 of this paper:

- *Cost*: The total cost of the system is divided between multiple independent owners of the search engines (service providers).
- *Minority users*: Any organisation or individual can set up a search engine and include it into the distributed system, provided that the engine complies with the system inter-communication protocols used. Since the cost of each search engine can be arbitrarily small (it does not have to index the entire Web), even small user groups can afford a search engine to satisfy their needs.

- *"Invisible" Web*: Owners of "invisible" document collections can establish search engines for their collections and plug those search engines into the global distributed system. By analogy, providers of subscribed information can establish a search engine for their documents without actually giving access to the document content.

These advantages come at the price of new research problems that need to be resolved to allow for real-life distributed search systems. These problems are discussed below.

4. RESEARCH PROBLEMS

The problems that need to be addressed for successful distributed search systems can be identified as follows:

1) *Query routing and result aggregation*: How to select the search engine(s) to which each particular user query should be sent and how to aggregate the results from multiple sources. Query routing requires some forward knowledge about search engine parameters, including its topic and service price. Therefore, we can differentiate between three sub-problems here:

   – Describing services provided by search engines (e.g. representing the engine's topic);
   – Using the above service descriptions to route queries;
   – Merging results from multiple search engines.

2) *Efficient construction of topic-specific databases*: For a non-topic-specific Web robot, all Web pages have equal value, since the search engine attempts to index all information on the Web without any particular preferences. For a topic-specific Web robot, only pages relevant to the engine's topic are valuable. An efficient topic-specific robot should avoid downloading irrelevant pages so as to minimise the cost of building the database;
3) *Service management*: How search engines should choose their service parameters, such as the topic or the service price. The complexity here arises from the mutual influence between search engines in the system. While each engine is independent in selecting its own service parameters, they are not independent in terms of the results achieved. The utility of any local parameter

change may depend on the actions of other search engines in the system.

4) *Information infrastructure*: One needs to decide what elements of the distributed architecture are responsible for query routing and result merging, as well as how information about search engines is disseminated in the system and announcements of new services are made.

Most of research to date has been concentrated on the first two problems. Little attention has been paid to the questions of service management and information infrastructure.

5. PREVIOUS WORK

In this section we discuss the previous work in the area grouped by the problem domains.

Distributed search architectures (DSA)

The DESIRE project created a distributed search system where most engines in the system provided a search service for a particular network or geographic domain [9]. A geographic or network domain based partitioning of the index means, however, that for most user queries each engine is equally likely to index relevant resources. This implies that most queries must be routed to all engines in the system leading to degradation of system performance.

In Harvest, the search engine functionality is split in two components: broker and gatherer [8]. Gatherers collect information about available resources. Brokers perform indexing and process user queries. Brokers can collect information from several gatherers and propagate user queries to other brokers. They can be both non-topic-specific and topic-specific. However, query propagation routes and possible broker specification are configured manually, which makes management of large Harvest-based systems very complicated [4].

An interesting example of distributed search architecture is the OASIS project [6]. OASIS service consists of topic-specific search engines (collections). Collections' parameters are described in a global OASIS directory. When a user issues a search query to one of the OASIS search engines, the engine can both search its own index and propagate the query to other collections that may contain potentially relevant information. Collection descriptions from the global directory are used for query routing. However, building of new collections,

selection of their topic specification, and managing other service parameters is done manually.

Query routing and result merging

Query routing between topic-specific collections of documents has received a lot of attention to date. Most approaches to specification of document topics in the textual information retrieval are focused on analysis of statistical text characteristics. One of the most popular models is the vector-space model [10]. In brief, the document topic is viewed as a vector where values represent "weights" of distinct terms in the document (e.g., based on the term frequencies within the document). A number of query routing methods exist that are based on the vector space model including GlOSS [11] and CORI [12].

GlOSS assumes that RDS consists of a central glossary server referencing multiple possibly topic-specific databases. This project concentrates on problems of propagating user queries to databases that are expected to contain more relevant results. CORI represents information about collections in the form of an inference network, which allows for very moderate storage requirements.

Similar problems are tackled in the Sheldon's work on content routing [13]. Content routers use compact descriptions of document collections called content labels to route queries. Discovery project uses the content routing and query refinement techniques to search a large number of WAIS servers [14].

Some recent results on the query routing and collection description are presented in Craswell *et al* [15], where the authors propose a new method for estimating relevance of databases by sending probe queries and analysing results. The same work also analyses methods for result merging identifying two basic categories: isolated and integrated merging.

Construction of topic-specific databases

The first crawlers were used to obtain non-topic-specific samples of the Web for general-purpose search engines [16]. A topic-specific Web crawler, on the other hand, is used to build a database of Web pages on a specified topic. The problem for a topic-specific Web crawler is to schedule document requests so as to retrieve the most relevant documents as soon as possible.

Early algorithms, such as Shark-Search [17], the Focused Crawler [7], and the OASIS Crawler [18] below were heuristic based. These algorithms provide limited levels of performance. Their

limitation stems from their inability to adapt to the complex and poorly understood document environment in which they operate.

Recent machine learning approaches, such as InfoSpider [19] and Cora Spider [20], have attempted to overcome these limitations by adapting their scheduling strategies to observations of real world documents. These algorithms, however, have been based on incomplete and oversimplified models of the problem. They do not take into account, for example, the need to make concurrent document requests to hide network latency, and the need to avoid overloading Web servers with rapid-fire-requests.

Service management

The lack of effective mechanisms for management of search engines in distributed search architectures is currently a serious problem and one of the main reasons why they are not widely deployed [4]. To the best of the authors' knowledge, no results have been published to date on service management in DSA for the Web. The closest related research was done in the area of distributed database systems. The Mariposa project [21] is worth a mention here.

Mariposa adopted a microeconomic approach to the problems of query execution and storage management in distributed database systems. In Mariposa, the distributed system consists of a federation of databases and query brokers. Query brokers partition queries into sub-queries that can be executed in parallel and then find a set of databases that can execute the sub-queries. Selection of databases is done via a bidding process, but only the databases having the necessary data may bid for a query. The databases can trade data fragments between themselves so that they can bid for profitable queries and thus maximise their revenues. This process resembles the management of topics for topic-specific search engines, but cannot be adopted in DSA design directly. Search engines can acquire data (Web pages) independently from other engines (i.e. do not have to buy them) and there is no exact match between Web pages and queries as there is between data fragments and database queries.

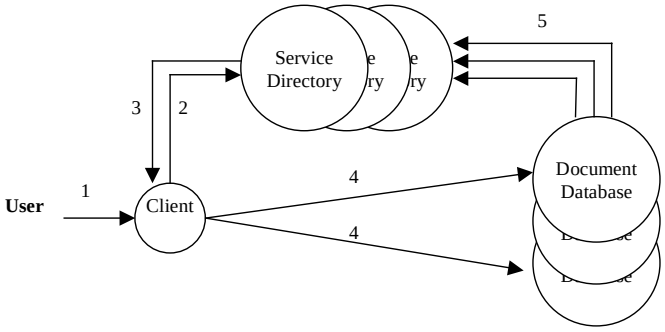
4. ADSA PROJECT

The ADSA (Adaptive Distributed Search and Advertising) project builds on the results of the OASIS project [6] and aims to develop a scalable Internet search system based on the concept of topic-specific distributed search architectures presented in this paper.

An important aspect that the ADSA system is going to address is the generation of revenue from distributed search services. Due to the now widespread expectation of free search services on the Internet, revenue generation from search services requires a new pay-for-placement business model. In this model, advertisers pay search engines to include their Web pages in search results on a per-click basis.

The advantages to the advertiser, search service provider, and end user in this model are considerable. The advertiser only pays for actual Web site visits that result from the advertisement and the service provider obtains revenue from the service. By clearly displaying the cost to the advertiser in the search results, the end user benefits from the transparency in the system. This is in contrast to the situation where advertisers pay third party companies to perform "behind the scenes" manipulation of keywords in their Web pages to improve their ranking in search results.

The ADSA architecture consists of three basic components, each of which can run on a separate computer system, communicate with other components over a network, and can have different owners. Figure 2 shows the ADSA components and their interactions. The main components are: the Document Database, the Service Directory, and the Client. The distribution of the system at the component level provides for much greater flexibility.



Legend:
1 - User submits a search or advertising request
2 - Request is sent to a Directory
3 - Directory returns list of Databases
4 - Request is sent to selected DBs
5 - Databases register themselves with Directories.

Figure 2: ADSA architecture

The following describes the functionality of each ADSA component:

- The *Document Database* is essentially a topic-specific search engine. It maintains an index of Web documents (or some other proprietary content in the case of "invisible" Web) and processes user search queries submitted to the engine.
- The *Service Directory* can be called a "database of databases": its purpose is to find the best Document Databases for a given user request whether it be a search or advertising request (that is request routing). To become known to the ADSA system, databases can advertise (register) themselves in Service Directories. Like the Document Database, the Service Directory has a "search engine style" interface: a Directory accepts requests in the same (or almost the same) form as a Document Database, but returns links to Databases rather than links to documents.
- The *Client* provides the system with a user interface and is responsible for request propagation and result merging. For each user request, the Client submits it to a Service Directory, retrieves a list of best matching services (Document Databases), and then propagates the request to the selected Databases. The search results from the database(s) are merged and returned to the user. For document advertisement (pay-for-placement) requests, there is no need to merge results, since results on advertising requests are visits by search users to advertised Web pages.

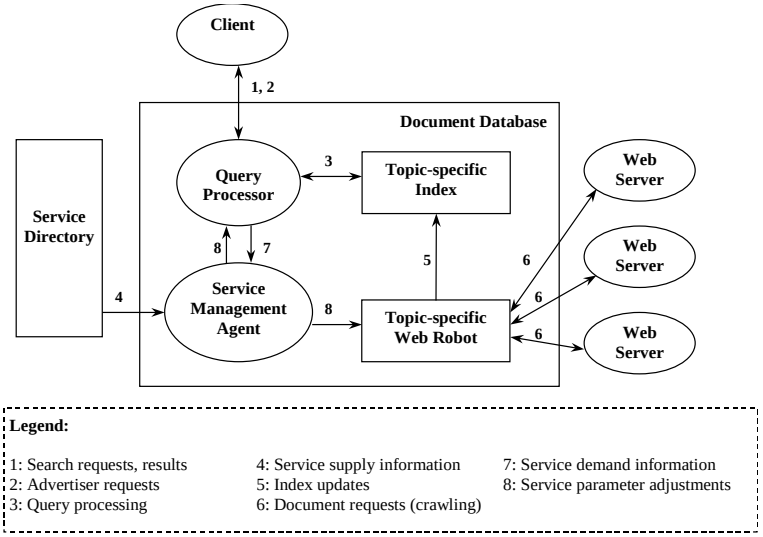


Figure 3: ADSA Document Database

A more detailed structure of the Document Database component is illustrated by Figure 3. The problems addressed by the ADSA project include:

- *Topic-specific Web robot algorithms*. These algorithms will be employed by the Web Robot to efficiently construct and maintain topic-specific Web indexes by automatically following hyperlinks to discover Web pages relevant to a selected topic. The latest research has identified a particular class of stochastic scheduling models as having the best potential to address the full topic-specific Web robot problem [22]. This work will investigate the application of neuro-dynamic programming techniques to solving these models. Neuro-dynamic programming techniques can generate high-performance solutions by training neural networks on observations of real world data [23].
- *Query routing algorithms*. The query routing algorithms require accurate forward knowledge of the Web index content in the Service Directory. The project will investigate the use of Latent Semantic Indexing techniques to generate index representations for the Service Directory and a decision-theoretic model for optimal database selection. These techniques have been identified as having a strong performance potential in this area [6]. Algorithms for routing of advertisement queries and producing advertisement service descriptions will also be investigated.
- *Decentralised service management strategies*. These strategies will be used by service management agents to control the service parameters of ADSA Document Databases, including the topic and service prices for search and advertising. The goal is to construct strategies so as to maximise the performance of a database in the presence of other databases in the system. The project will investigate game-theoretic approaches [24] to the problem, which have been shown to have a good potential [25].
- *Content-based document advertising*. Existing search systems, such as Overture (http://www.overture.com), that support document advertising (pay-for-placement) only support keyword-based advertising. Advertisers are required to bid for a set of key words associated with the advertised documents. When searchers submit these keywords in a query, the advertised documents are presented as results in the order determined by the relative bid sizes. This approach requires undue effort from the advertiser who must determine what query keywords to bid on and how much to

bid. In ADSA, advertisement of the documents will be based on the document content which will be automatically analysed. The position of the document in a result list will be determined by a combination of the document's relevance to the query and the advertisement payment associated with the document.

5. CONCLUSIONS

In this paper, we presented an overview of the concept of Distributed Search Architectures (DSA). We outlined what issues with existing search systems can be resolved using DSAs and described some key research problems associated with DSAs. We analysed the relevant research work in the area and proposed possible approaches to some of the problems in DSA. This analysis forms the research core of the ADSA project.

The main conclusion of the paper is that for search systems to scale with the Web growth they will have to adopt a distributed search model. In particular, a topic-oriented approach to building DSAs for the Web has the greatest potential to deliver a successful system. The authors believe that the results of the ADSA project will be important for the development of such systems for the Internet.

6. ACKNOWLEDGEMENTS

The support of the European Union INCO Copernicus Programme and the Informatics Research Initiative of Enterprise Ireland are gratefully acknowledged. All product names mentioned in this paper are the trademarks, registered trademarks or service marks of their respective owners.

7. REFERENCES

[1] C.A. Lynch, "Networked Information Resource Discovery: An Overview of Current Issues", *IEEE Journal on Selected Areas in Communications*, Vol. 12, No. 8, October 1995.

[2] R. Hock, *The Extreme Searcher's Guide to Web Search Engines*, 2nd edition, CyberAge Books, 2001.

[3] S. Lawrence and C.L. Giles, "Accessibility of information on the Web", *Nature*, Vol. 400, 1999.

[4] S. Brin and L. Page, "The Anatomy of a Large-Scale Hypertextual Web Search Engine", In *Proc. of the 7th International World Wide Web Conference*, Brisbane, Australia, April 14-18 1998.

[5] C. Sherman and G. Price. *The Invisible Web: Uncovering Information Sources Search Engines Can't See*, Independent Publishers Group, 2001.

[6] A. Patel, L. Petrosjan, W. Rosenstiel, *OASIS: Distributed Search System in the Internet*, St. Petersburg, Russia: St. Petersburg State University Published Press, 1999.

[7] S. Chakrabarti, M. Van den Berg, B. Dom, "Focused Crawling: A New Approach to Topic-Specific Web Resource Discovery", In *Proc. of the 8th International World Wide Web Conference*, Toronto, Canada. May 11-14 1999.

[8] Bowman, Danzig, Hardy, Manber, Schwartz, and Wessels, "Harvest: A Scalable, Customizable, Discovery and Access System", *Technical Report CU-CS-732-94*, Dept. of Computer Science, University of Colorado, March 1995.

[9] A. Ardo and S. Lundberg, "A Regional Distributed WWW Search and Indexing Service—the Desire Way", *Computer Networks and ISDN Systems*, Vol. 30, No. 1-7, 1998.

[10] G. Salton, *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer*, Reading, MA: Addison-Wesley, 1989.

[11] L. Gravano, H. Garcia-Molina, "GlOSS: Text-source discovery over the Internet", *ACM Transactions on Database Systems*, Vol. 24, No. 2, June 1999.

[12] J.P. Callan, Z. Lu, W.B. Croft, "Searching Distributed Collections with Inference Networks", in *Proc. of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ACM Press, July 1995.

[13] M.A. Sheldon, *Content Routing: A Scalable Architecture for Network-Based Information Discovery*, PhD Thesis, Department of Electrical Engineering and Computer Science, MIT, 1995.

[14] M.A. Sheldon, A. Duda, R. Weiss, and D.K. Gifford, "Discover: A Resource Discovery System based on Content Routing", In *Proc. of the 3rd International World Wide Web Conference*, Elsevier, North Holland, 1995.

[15] N. Craswell, P. Bailey, and D. Hawking, "Server selection on the World Wide Web", In *Proc. of the Fifth ACM Conference on Digital Libraries*, San Antonio, TX, USA, June 2–7, 2000.

[16] F. Cheong, *Internet Agents: Spiders, Wanders, Brokers, and Bots.* Indianapolis: New Riders, 1996.

[17] M. Hersovici, M. Jacovi, Y. S. Maarek, D. Pelleg, M. Shtalhaim, and S. Ur, "The shark-search algorithm. An application: tailored Web site mapping," *Computer Networks and ISDN Systems*, Vol. 30, Apr. 1998.

[18] I. Nekrestyanov, T. O'Meara, A. Patel, and E. Romanova, "Building topic-specific collections with intelligent agents," in *Proceedings of the Sixth International Conference on Intelligence in Services and Networks, IS'99*, Barcelona, Spain, April 27-29, 1999.

[19] F. Menczer and R. K. Belew, "Adaptive retrieval agents: Internalizing local context and scaling up to the Web," *Machine Learning*, Vol. 39, May 2000.

[20] A. McCallum, K. Nigam, J. Rennie, and K. Seymore, "Automating the construction of Internet portals with machine learning," *Information Retrieval*, Vol. 3, No. 2, 2000.

[21] M. Stonebraker, R. Devine, M. Kornacker, W. Litwin, A. Pfeffer, A. Sah, and C. Staelin. "An economic paradigm for query processing and data migration in Mariposa", In *Proc. of the Third International Conference on Parallel and Distributed Information Systems*, Austin, Texas, USA, Sept. 28–30, 1994.

[22] T. O'Meara and A. Patel. "A topic-specific Web robot model based on restless bandits", *IEEE Internet Computing*, Vol. 5, No. 2, March/April 2001.

[23] D. P. Bertsekas and J. N. Tsitsiklis, *Neuro-Dynamic Programming*. Belmont, MA: Athena Scientific, 1996.

[24] L. Petrosjan and N. Zenkevich, *Game Theory*, New Jersey: World Scientific, 1996.

[25] R. Khoussainov, T. O'Meara, and A. Patel, "Independent Proprietorship and Competition in Distributed Web Search Architectures", in *Proc. of the Seventh IEEE International Conference on Engineering of Complex Computer Systems*, Skovde, Sweden, 11-13 June 2001.