SIMULATION BASED APPROACH TO EVALUATE A DISTRIBUTED SEARCH ENGINE

Thomas Phelan, Ahmed Patel, and Séamus Ó Ciardhuáin

Computer Networks and Distributed Systems Research Group, Department of Computer Science, University College Dublin, Belfield, Dublin 4, Ireland

ABSTRACT

We present a simulation model used to evaluate a distributed search engine by modelling its performance. The model is designed to aid performance-driven evolution of the system. We present the results of simulations investigating the architecture of the system and how it will behave when scaled up, and show how the simulation gave insights into the behaviour of the real system.

KEYWORDS

Simulation, Search Service Management, Search Engine Performance

1. INTRODUCTION

The goal of this paper is to present a simulation tool for modelling our Adaptive Distributed Search & Advertising search engine (ADSA) (Khoussainov et al., 2001a).

The ADSA project has as its aim the development of a distributed search system that can scale with Web growth. A distributed system of topic-specific search engines has been developed. Such an architecture partitions its information databases into multiple smaller databases in an attempt to relieve the scalability problems of centralised search architectures, and also has the potential to incorporate privately indexable material into a logically unified database. Such a system can scale with Web growth since each engine will process only pages and queries relevant to a selected topic(s). It can also provide comprehensive coverage of selected topics, sophisticated query processing, and lower costs due to the smaller index sizes and query loads at each engine. Distributed topic-specific search will enable both large and small organisations to participate in a global search services market by providing specialist services in selected topic domains. A pilot service has been established and is expected to run until the end of 2003 (http://www.cnds.ucd.ie/adsa). To support this development, we have also built a simulation system which can be used to investigate the properties of ADSA with larger numbers of databases and users than are available in reality.

While distributed architectures are not an innovative idea, ADSA is unique in so far as it provides independent ownership of individual databases that dramatically reduces the entry cost for an organisation or even an individual into the global information search market (Khoussainov et al., 2001b). Anybody can install a small, private search engine without the prohibitive costs of maintaining large monolithic databases, and yet be a part of a global search system that can match the growth of the Internet for the foreseeable future. Independent ownership is especially attractive in the context of providing search service for privately indexable information. Providers can retain full control over indexing and information access, yet at the same time integrate themselves into a wide-scale search engine.

Due to the proprietary nature of many search systems very little research has been published on their performance evaluation (Floyd and Paxson, 2001; Lorenz, 1997; Unger et al., 1998). With this in mind existing simulation models cannot be applied to measure its performance and hence we developed our own unique model to assist future research in the area of workload modelling of distributed information retrieval.
With this in mind, we have built a simulation harness which allows us to investigate different aspects of the ADSA distributed search engine. In this paper we report the results of using it to investigate performance issues. In future work, we intend to use it to investigate further aspects of the system such as service management.

2. SEARCH ENGINE SIMULATION MODEL

The ADSA system consists of many topic-specific Document Databases containing descriptions of web documents on the selected topic. Queries to the system are initially sent to a Service Directory, which ranks the Document Databases in order of their relevance to the query. Queries are then routed to the relevant Document Database(s) which perform a search and return the results to the user. The Document Databases are populated by a topic-specific Web robot. The prototype implementation uses MySQL as the database.

The simulation was implemented as a multi-threaded, event-driven (for exponential time testing) and continuous (for capacity and performance testing) application (Lin, 2000). The “pull” model of document acquisition was implemented, i.e. a Web robot is used to retrieve documents from Web servers by recursively following hyperlinks. The robot component simulates retrieval of documents from the Web and the subsequent population of a document database (Figure 1). It updates the current database content description in the same way as if a retrieved document were added to the database. Essentially, this process simulates the functioning of an “ideal” Web robot. Documents are downloaded by the robot from a specified “seed” location and a profile of each document is created, compared against the database content filter, and a score assigned to the document based on similarity between the content and the filter. The score indicates the relevance of a document with respect to the selected topic. The robot then checks to see whether the document is in the database already. If not, it is added to the database.

The Web robot simulation provided us with the ability to simulate different parameter adjustments, notably RFR (Rapid-Fire-Requests, i.e. the amount of requests sent by a robot to the same web-server in close succession) and bandwidth levels, in order to assess what impact these would have on total query time while document downloading was occurring concurrently.

Once all databases have been populated with documents, the simulation makes a database selection. The database selection function is run when user requests are submitted to the Service Directory component. Database selection for a request is performed in two steps. Firstly, the desired content description is compared to the contents of all databases registered within the Service Directory, and the relevance scores are calculated. The Collection Retrieval Inference (CORI) network algorithm is used. The rest of the parameters (time, number of results and price) are calculated and an overall utility is computed using a request parameter weighting scheme. This utility value is then used for ranking of the databases in order of appropriateness for the request.
Once the best database has been selected, a vector space model for information retrieval is used to extract the relevant documents. Weighted term vectors represent incoming requests from simulated Search Clients. Simply, this is assigning a weight to a term in some document. It was implemented as a simple, geometric interpretation of retrieval. In addition it provides a unifying basis for a very wide range of retrieval operations including indexing, relevance feedback and document classification. There are several processes in the execution of this simulation.

1) Word separators are defined. These are specific characters which are treated as marking a division between distinct terms in a query, e.g. a space is always treated as a word separator.

2) A “Stop Word Removal” function is then called to remove any common words that are frequently used in order to save space and ultimately speed up searches.

3) The Data Dictionary is loaded only once for the entire simulation process. It contains a list of all files in the database, the number of records in each file, and the names and types of each field. The Data Dictionary is required because we need more advanced database schemas, especially when tables are connected with external keys and key columns. When accessing data from several tables, the database schema will be needed in order to find joining data elements and in complex cases to find proper intermediate tables.

4) A query is then constructed and passed as a MySQL statement to the selected database.

The search engine database tables contain specific fields that are paramount for information retrieval of specific requests. These include the weights (relevance scores) assigned to each word contained in the title of each document contained within the database. The success and number of relevant returned documents based on any given query is determined by the similarity between the Word_ID (Word_ID’s are simply numbers representing words; they help reduce access times and space consumption) for a particular query and the (combined) weightings for those words stored within the tables.

Figure 2 below shows a high-level view of the full search system simulation model.

![Search Engine Simulator Diagram](image)

Figure 2. Search engine simulator

The simulation model was validated by comparing the results obtained from it to measurements of the prototype ADSA distributed search system, which is being run as a pilot service with a limited number of Document Databases and Service Directories. It was found that the simulation model’s predictions agreed very well with the prototype's behaviour, and this gives us confidence that the performance predictions described below are an accurate basis for assessing the behaviour of the system when deployed on a large scale.

3. PERFORMANCE MODELLING

Performance modelling, in particular load balancing, is extremely useful to see how well the components would deal with the very large volumes of data that would be encountered in the real world. The results can help to predict future behaviour of the real system and identify possible system bottlenecks before they actually happen. In our simulation model, we are able to examine:
• Total search engine time
• Average query response time
• Average number of documents returned per query
• Database throughput rate

In practice, queries/requests do not arrive at evenly-spaced intervals. To address this, we implemented an M/M/1 event queue simulation, which was used to estimate system performance levels. M/M/1 queuing theory assumes a Poisson arrival process. This assumption is a very good approximation for the arrival process in real systems where the number of customers in the system is very large, the impact of a single customer on the performance of the system is very small (i.e. a single customer consumes a very small percentage of the system resources) and all customers are independent (i.e. a customer’s decision to use the system is independent of other users). These assumptions are relevant in the case of the ADSA system for the following reasons:

• The number of user requests coming into the system can be very large.
• A single request has little or no impact on the overall system performance, i.e. it consumes a very small percentage of the system resources.
• All requests are independent of each other.
• ADSA supports a single server environment (one process running on one server) e.g. a single request queue is not feeding two load-sharing communication links to the same destination.

Using this queuing model, we were able to investigate the following system properties:

• Average number of queries in the network
• Average number of queries in the database queue
• Average network-waiting time for queries
• Average database queue waiting time
• Number of queries serviced
• Throughput rate
• Database utilisation
• Network delay

The simulation performance model allows us to assess how the specific metrics identified above would change/evolve in time under the influence of events occurring at irregular time intervals. It was important to model the search engine as a discrete event system because such systems abound in real-world applications, and also because they provide a framework for future sensitivity analysis and system optimization.

4. PERFORMANCE EXPERIMENTS AND RESULTS

Query logs were obtained from an ISP and used as the inputs to the simulation. Approximately 500,000 queries collected over a period of eight months were used, giving a representative sample of user queries. In general when analyzing configurations statically, care must be taken in choosing workloads to ensure they are representative of the users usage. However, as we were simulating the system from a robustness and performance point of view we did not need to worry about by logging the usage of the actual pilot system to generate a set of inputs for the trials, or to guide the production of synthetic workloads.

Initially, we investigated the performance impacts of two important architectural questions in the design of the ADSA system. First, we considered whether the updating of the document database by the robot concurrently with processing user queries would adversely impact the system’s performance. We simulated using the robot to populate the database starting with 1,000,000 seed URLs pointing to local documents (to avoid undesirable impact on remote servers) while also processing the full query stream. Queries were tested against a 20,000 document database and robot usage accounted for approximately 1% of total query time. Tests were carried out with different levels of network bandwidth available to the robot. It was found that there was little or no impact on query response times at any level of robot activity (see Figure 3). This shows that the robot is not CPU dependent but rather network resource intensive.
The second architectural issue was the impact of the need for all queries to pass through the Service Directory. The project requirements specified that the Service Directory component of ADSA should be capable of supporting 1,000 databases each containing 200,000 documents. We could not test this with the prototype system so the simulated Service Directory was populated with 800 database descriptions and the full query stream processed. It was found that the SDIR as designed and implemented was fully capable of supporting the requirement (Figure 4).

Next we turned our attention to the performance of the Document Database component of ADSA. The prototype system was encountering poor response times which our earlier work had indicated were not due to either the robot or the Service Directory. Because all the application components resided on the same server for testing purposes, other issues such as network bandwidth, latency etc. did not contribute to the overall processing time of the queries. This suggested that the behaviour of the Document Database was the most likely cause of poor response times, especially the information retrieval algorithms used. We investigated two possible reasons for performance delays:

- database size
- query structure
Figure 5 Average query time vs database size

Figure 5 above shows that as the size of the database grows (in terms of documents) so does the average query time. This is in accordance with our expectations of the sizing issues which might arise with the use of the MySQL relational database for this application.

Figure 6. Average query time vs database size (complex queries)

We turned our attention to the types of query and identified certain types that might impact on the overall response time. These were queries which used Boolean operators and had a larger number of search terms. Using a query stream consisting of these complex queries, we obtained the results shown in Figure 6. These results indicate that the structure of the query also has an impact on the response time.

The biggest workload we could feed our simulation was by using the Data Dictionary as the input query stream to the system. This we believed was the ultimate test of algorithm performance because for every word in the data dictionary there is at least one corresponding document in the database, thus forcing the retrieval algorithm to match all possible queries to all relevant documents. The results concluded that the performance time increased significantly (Figure 7). This gives a worst-case upper bound on query time. The sharp increase in query time for larger databases is probably due to the behaviour of the cache system used; this will be the subject of further research.
5. DISCUSSION

The most important result from the simulation is that the basic ADSA architecture using a Service Directory to route queries to the Document Databases gives acceptable performance. When combined with the better performance observed for smaller databases, this indicates that the ADSA concept of using many smaller topic-specific databases working together in a distributed search system is very effective.

It should be noted that these results were obtained using a standard relational database (MySQL) rather than a custom-designed database as is required for monolithic search engines. Up to 200,000 documents the relational DB gives acceptable performance; this database size is reasonable in a topic specific system because it is not necessary for the database to index the entire Web, but only a small part of it.

6. FUTURE WORK

The success of the ADSA system will depend greatly on its behaviour with large numbers of smaller topic-specific databases. Future work will focus on:

- evaluating behaviour with many databases
- identifying improved information retrieval algorithms and how they may be used to improve performance with larger databases (over 200,000 documents)
- management of the databases, in particular the parameters used for database ranking

7. CONCLUSIONS

The simulation model has proved to be a valuable tool for understanding the behaviour of a distributed search engine system. By building a simulator we have been able to confirm that the architecture of the system does not adversely affect its performance. We have also been able to investigate the behaviour of the system in large configurations than could be implemented in practice, and have gained important insights into the behaviour of our prototype system. The simulation has pointed to possible scaling issues which will form the basis of future research work.
ACKNOWLEDGEMENTS

The ADSA research project is funded by Enterprise Ireland under the Informatics Research Initiative.

REFERENCES


