Site-To-Site (S2S) Searching Using the P2P Framework with CGI

Wan Yeung Wong
Department of Computer Science and Engineering
The Chinese University of Hong Kong
Shatin, Hong Kong
wywong@cse.cuhk.edu.hk

ABSTRACT

Peer-To-Peer (P2P) networks like Gnutella improve some shortcomings of Conventional Search Engines (CSE) such as centralized and outdated indexing by distributing the search engines over the peers, which maintain their updated local contents. But they are designed for sharing and searching the contents in personal computers instead of websites. In this work, we propose a novel web information retrieval method called Site-To-Site (S2S) searching, which uses the P2P framework with CGI as protocol. It helps the site owners to turn their websites into autonomous search engines without extra hardware and software costs. In this paper, we introduce S2S searching with some related work. We also describe the system architecture and communication protocol. Finally, we summarize the experimental results, and show that S2S searching works well in one thousand sites.

Categories & Subject Descriptors: H.3.3 [**Information Search and Retrieval**]: Search process

General Terms: Algorithms, Performance

Keywords: Search Engine, Web Information Retrieval, Site-To-Site (S2S), Peer-To-Peer (P2P), Distributed System

1. INTRODUCTION

Conventional Search Engines (CSE) like Google and AltaVista have three shortcomings, which are (1) centralization of resources used, (2) no control over information shared by the content owners, and (3) lack of relevant feedback from the users. In this paper, we propose Site-To-Site (S2S) searching in order to distribute the search engines over websites based on Peer-To-Peer (P2P) paradigm without extra hardware and software costs. It improves the three aforementioned shortcomings of CSE.

1. Centralization of Resources Used: CSE are centralized which require powerful servers to handle search requests. And they need large storage space to store the crawled contents, which are not always up-to-date as the web pages are being updated [1]. To achieve high performance, the hardware cost is heavy. On the other hand, S2S search engines are decentralized. So they need less powerful machines to handle search requests, and less storage space to store the local index. Each site maintains its own local index, which is always up-to-date.

2. No Control over Information Shared: CSE crawl all published contents on the web, and make them become searchable without their owners' permissions. The site owners are also unable to alter the ranking strategy for their prioritized contents. On the other hand, S2S search engines allow the site owners to selectively disable their published contents to be searchable. They could also prioritize their contents in order to advertise and rank the results in a more customized way.

WWW 2004, May 17-22, 2004, New York, NY, USA.
ACM 1-58113-912-8/04/0005.

3. Lack of Relevant Feedback: CSE ignore the intentions, interests, and preferences of their users, as the search engines always return the same search results with the same keywords for every user [3]. On the other hand, S2S search engines provide relevant feedback by monitoring the actions of the users to the search results. Users could express their preferences by giving scores to a particular link (assume there is no cheating).

Related Work: Unlike crawler-based search engines, P2P networks like Gnutella [2] and YouSearch [4] offer a real-time information retrieval based on most updated contents. Gnutella is designed for sharing and searching the contents in personal computers, which is not optimized for the web search. YouSearch is designed for searching in the network of personal web servers, which is also not optimized for searching in ISP web servers. It improves the query flooding problem of Gnutella, by using the centralized registrar to summarize the local index for each peer. On the other hand, S2S searching is designed to share and search the contents in websites, which are hosted by ISPs. Currently, it does not prevent the query flooding, which would be improved by integrating some query routing algorithms in the future.

2. ARCHITECTURE AND PROTOCOL

The S2S paradigm makes each website, which joins the S2S network, becomes an autonomous search engine. The site owners only need to install the S2S software to their websites, which is a set of CGI programs written in Java Servlet. Then the users could use the search form to start searching the web contents. The query requests are propagated from site to site, which are limited by the Time-To-Live (TTL) value. Finally, all the search results are propagated back to the requester, and displayed to the users.

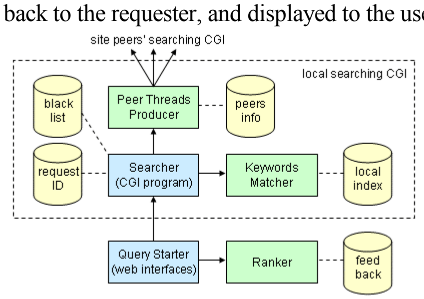

Figure 1. System Architecture

System Architecture: Figure 1 shows the system architecture of S2S search engines. When the *query starter* receives a query request from the search form, it generates a unique request ID, which is passed to the *local searching CGI* together with the keywords and other parameters in the search form. Inside the *local searching CGI*, the *searcher* checks if the requester is in the black list by its IP address, and the current request is a repetitive request by its request ID. The next step is to check if the TTL value from the CGI parameters is greater than zero. If it is, then the *searcher* asks the *peer threads producer* for spawning threads to broadcast the query request to the adjacent sites. Each thread calls a distinct site's *searching CGI*, and waits for its return. During the waiting period, the *searcher* asks the *keywords matcher* to match the keywords by looking at the local index. Once they match, the similarities are calculated and the results are returned. Usually, the *keywords matcher* is able to utilize the full CPU resource, as the *peer threads producer* is idle for waiting other sites' *searchers* to return. This makes the searching process highly distributed and efficient. After some time, the *searcher* gathers the results and returns to the *query starter*, which forwards the results to the *ranker*. It ranks the results based on the four values of the documents. They are the priority value (*priority*) which is assigned by the site owner, click proportion (*click*), average users' scores (*score*), and similarity (*sim*) which is calculated by the *keywords matcher*. The final ranking value (*rank*) is calculated by

$$rank = p \times priority + q \times click + r \times score + s \times sim ,$$

where p, q, r, s are the adjustable ranking parameters. Finally, the *ranker* sorts the search results in descending order by the ranking values, which are then displayed to the users.

S2S Communication Protocol: S2S searching targets on those site owners, whose websites are hosted by ISPs. So they have limited site administration privilege. It is a challenge to make S2S search engines plug into the websites easily, which does not require any system administrator to install special software and open specific firewall. Taking these into consideration, CGI is a good choice. There are four CGIs for the S2S communication protocol. (1) *Starting CGI* is called by the search forms for starting the search requests. It also generates a unique request ID, and calls the local *searching CGI* for obtaining the search results. (2) *Searching CGI* is called by local or other sites for searching the target information. It also calls the *searching CGIs* of other sites to broadcast the query requests. (3) *Pinging CGI* is called by other sites for querying the information about the current site like the response time. (4) *Joining CGI* is called by other sites for requesting the current site to join another site.

3. EXPERIMENTS AND DISCUSSIONS

There are two experiments which measure the (1) performance of S2S searching and (2) dependence of searching time. Table 1 shows the configurations of three different types of computers, which overall performances are much lower than those dedicated web servers.

Table 1. Computer Configurations

	Model	RAM	Network	Overall
X	Sun Blade 1000	2GB	100Mbps	Fast
Y	Sun Ultra 5/400	512MB	100Mbps	Medium
Z	Sun Ultra 1/140	64MB	100Mbps	Slow

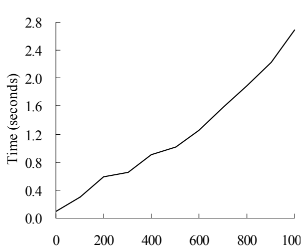

Figure 2. Experiment 1

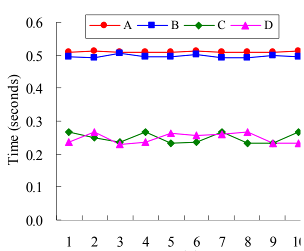

Figure 3. Experiment 2

1. S2S Searching Performance: This experiment is to measure the performance of S2S searching. The simulation is done by two computers X. The local searching time of each site is fixed to 0.1 second. Figure 2 shows the relationship between the total searching time and the number of sites in the S2S network. We measure the worst case by using the *worst connection structure*, which adjacency matrix *adj* is defined as

$$adj_{ij} = \begin{cases} 1 & \text{if } |i-j| = 1 \\ 0 & \text{otherwise} \end{cases} .$$

2. Searching Time Dependence: This experiment is to show that the total searching time depends on the slowest site in the S2S network. Five websites with 441KB HTML documents each are connected by using the *worst connection structure*. Figure 3 shows the total searching time in ten trials. Line A is obtained by using four computers Y and one computer Z. Line B is obtained by searching locally in the computer Z. Line C is obtained by using four computers Y and one computer X. Line D is obtained by using five computers Y.

Discussion: From the first experiment, we observe that S2S searching is quite efficient in a large scaled S2S network. It is due to the highly distributed and parallel searching process. The searching time is further improved if a better keywords matching algorithm is used. From the second experiment, we demonstrate that the total searching time depends on the slowest site in the S2S network. It is because the query request is first broadcasted to all sites. Then each site performs a local searching. Those fast sites, which finish their searching, always wait for those slow sites to return. So if there are very slow sites that join the S2S network, the total searching time may be unacceptable. On the other hand, S2S search engines circumvent this problem by applying the timeout mechanism in order to skip those slow sites.

4. CONCLUSIONS AND FUTURE WORK

In this paper, we present a novel web information retrieval method called S2S searching, which helps the site owners to turn their websites into autonomous search engines without extra hardware and software costs. Finally, we show that S2S searching works well in one thousand sites. Since S2S technology is a relatively new topic, there is still much research that could be done. We plan to improve the current query flooding problem by integrating some algorithms to S2S searching. We also plan to extend S2S searching to include multimedia information retrieval in the future.

5. ACKNOWLEDGMENTS

This work is supported in part by the Earmarked Grant, CUHK #4351/02E, from the Research Grants Council (RGC) of Hong Kong Special Administrative Region.

6. REFERENCES

[1] D. Fetterly, M. Manasse, M. Najork, and J. Wiener. A Large-Scale Study of the Evolution of Web Pages. In Proceedings of 12th International World Wide Web Conference, 2003.

[2] Gnutella website. http://www.gnutella.com

[3] J. Pujol, R. Sanguesa, and J. Bermudez. Porqpine: A Distributed and Collaborative Search Engine. In Proceedings of 12th International World Wide Web Conference, 2003.

[4] M. Bawa, R. Bayardo, S. Rajagopalan, and E. Shekita. Make it Fresh, Make it Quick – Searching a Network of Personal Webservers. In Proceedings of 12th International World Wide Web Conference, 2003.