Computer and Information Science
www.ccsenet.org/journal.html
Vol. 1, No. 4
November 2008

Multi-level Frontier based Topic-specific Crawler Design with Improved URL Ordering

Akilandeswari Jeyapal (Corresponding author)
Sona College of Technology
Salem - 636005, Tamilnadu, India
E-mail: akila_rangabashyam@yahoo.com

Gopalan Palanisamy
National Institute of Technology
Tiruchirappalli - 620015, Tamilnadu, India
E-mail: gopalan@nitt.edu

Abstract

The rapid growth of World Wide Web has urged the development of retrieval tools like search engines. Topic specific crawlers are best suited for the users looking for results on a particular subject. In this paper, a novel design of a topic specific web crawler based on multi-agent system is presented. The architecture proposed employs two types of agents: retrieval and coordinator agents. Coordinator agent is responsible for disseminating URLs from crawling frontiers to individual retrieval agents. The URL frontier is modeled as multi-level queues to implement tunneling and is populated with URLs by a rule based engine. The coordinator agent dynamically assigns URLs to retrieval agents to avoid downloading non productive and duplicate Web pages. The empirical results clearly depict the advantage of using multi-level frontier queues in terms of harvest ratio, time, and downloading highly relevant Web pages.

Keywords: Topic-specific crawler, Multi-agent system, Rule based system, Multi-level frontier queue

1. Introduction

The World Wide Web (WWW) or Web can be sighted as a huge distributed database across several million numbers of hosts over the Internet where data are stored as Web pages on Web servers. The logical relationship between these Web pages is represented by hyperlinks. Using the hyperlinks, users can navigate through the Web pages with the help of browsers which is time consuming. Another way to locate information is through search engines. As the size of WWW is colossal, search engines become a vital tool to search for information. Different search tools can be compared with the number of indexed pages, quality of returned pages, and response time. Google has gathered about 25 billion Web pages during 2006, but covers only 40% of the publicly available Web pages. Most of the empirical studies say that Google outperforms in both technology of gathering Web pages and indexing methods. Because of its huge collection most of the search results are irrelevant to the users. Now-a-days, people began to use more specialized search tools which will fetch only those URLs that are more important to them.

Web spiders or crawlers or robots are the main component of every search engine. The crawlers visit Web pages across the Web, following hyperlinks from site to site, storing downloaded pages as they visit, to build a searchable index. A crawler design is termed as good if it handles certain issues in an efficient way (Cho et.al., 1998). The design criteria mentioned below endorse the fact that every crawler should posses the techniques to handle them in an appropriate manner:

- avoid overloading Web servers or network lines
- deal with large volume of data
- should have unlimited resources and time (ideal requirement)
- decide on a particular link to follow
- decide on re-crawl policy to keep the database fresh

General purpose crawlers have the disadvantage of processing large portion of the Web in a centralized manner.

Currently, some of the general purpose search engines employ distributed crawlers and distributed indexes. A collection of distributed focused crawlers can collect more specialized topics in more depth and can maintain the freshness of the indexes since the re-crawl covers only less portion.

Unlike general-purpose web crawlers, focused crawlers are designed to gather pages on specific topic. A focused crawler tries to "predict" whether a target URL is pointing to a relevant and high-quality Web page before actually fetching that page. There are many works in focused crawling. Researchers have employed machine learning approaches to improve the crawler's functionalities in many aspects. Classification is one of the supervised learning algorithms that is used to determine the relevance of a Web page. Another important application of the above approach is to build and maintain Web directories or portals. In semi-supervised machine learning approaches like reinforcement learning, the agents learn progressively by interacting directly with the dynamic environment. Because of the dynamism, the agents are never told about the correct action, instead they are told about how good or how bad its action was.

As the scenario of information search in a large environment such as Web becomes tough, there is some assistance in the form of agents. The agents can act as autonomous computational entities for accessing, discovering and retrieving information on the Web. Unlike objects, an agent is defined in terms of its behavior. The characteristics of agents have made its use in several research areas in multi disciplinary domains. The properties like its autonomy, cooperativeness among other agents to accomplish a task, and its adaptability to the changing environment makes it suitable to Web search scenario.

Since a single agent approach may be inefficient and impractical for the large scale IR environment, most of the systems employ multi-agent systems. This paradigm has become more and more important as they represent a new way of analyzing, designing, and implementing complex software systems. In multi-agent systems, communication and organization enables the agents to cooperate and coordinate their actions. There are number of communication languages like KIF (Knowledge Interchange Format) (Genesereth et.al 1992), KQML (Knowledge Query and Manipulation Language) (Finin et.al 1994), and ACL (Agent Communication Language) (Labrou et.al 1999).

In this paper, an architectural framework is presented for crawling topic specific Web pages using multi-agent based system. The framework consists of multiple, parallel crawling agents and a master agent to coordinate them. In this prototype, the crawling agents are incorporated with intelligence which guides them in deciding the appropriate URL to download next. This feature is enabled with the help of an enhanced rule based system employing multi-level crawl frontier. The rule based system has the capability of harvesting a relevant URL after visiting some URLs with less relevance scores. This concept is called tunneling.

The remainder of this paper is organized as follows: section 2 discusses on the work related to Web crawlers. In section 3, the architectural framework of the crawler is described. In section 4, experimentation and evaluation details are discussed. Finally in section 5, conclusions are presented.

2. Related Works

The Web in many ways simulates a social network: links do not point to pages at random but endorses the page authors' idea of what other relevant or interesting pages exists. This information can be exploited to collect more on-topic data by intelligently choosing what links to follow and what pages to discard.

One of the pioneer approaches in ordering the URLs according to relevance is FISH SEARCH (De Bra et al., 1994). The system is query driven and considers only those pages that are matching the query and links that emanate from those pages. An improvement of Fish Search is proposed as Shark search (Hersovici et al.,1998). This algorithm uses weighted term frequency (TF) and inverse document frequency (IDF) measure to determine page relevance score. Another algorithm (Cho et.al) proposed a technique to reorder the URLs in the frontier queue according to various heuristics like page rank, in links count, back link count and combination of these attributes.

A soft focused crawler is proposed by Chakrabarti et al. (1999). This technique uses classifier to obtain a score. The main shortcoming of this technique is that it will not support tunneling i.e. following a path of off-topic pages.

A context-graph based crawler (Diligenti et al. 2000) and Cora's crawler (McCallum et al., 1999) use tunneling concept. Cora is a domain specific search engine whose spider employs reinforcement learning algorithm. There are other systems like Web Topic Management System (Mukherjea, 2000) that fetches only those pages that are parent, child or sibling to on-topic pages. Bingo! (Sizov et al., 2003) is a crawling system that eliminates the initial training step and employs progressive training of the classifier with high quality pages. Menzcer et al. (2003) presented a framework to evaluate focused crawlers and developed a crawler based on an evolutionary algorithm. An information integration framework ALII is presented by Barfouroushi et al. (2002) which uses active logic. ALII, employs a compact context representation and build a hierarchy model of query and web pages. The crawler also does limited backward crawling using general search engine indices.

An intelligent crawling architecture is presented by Aggarwal et al. (2001) based on predicates. It uses a self-learning mechanism that can dynamically adapt to the particular structure of the relevant predicate. Several factors were considered during the crawl to evaluate its effectiveness, building a composite crawler.

Page relevance to the user's need is computed using ontology-based algorithm by Ehrig et al. (2003). Entities, the words occurring in the ontology are extracted from the page and are counted. Relevance of the page is computed with regard to user selected entities using several measures on ontology graph like direct match, taxonomic and more complex relationships. This system has improved harvest rate when compared to baseline focused crawler that decides on page relevance by a simple binary keyword match. Case based BDI-agent (Olivia et al., 1999) is a domain specific search engine that uses case based reasoning (CBR) as its learning component. It uses past results and reuse that for answering future queries.

3. Architecture

The components of the crawler are shown in figure 1. The system proposed has two types of agents: coordinator agent and number of retrieval agents. The retrieval agents download Web pages and classify them as relevant or irrelevant. Coordinator agent disseminates URLs to different retrieval agents. The system should consider three key issues.

1. deciding on whether the downloaded pages are on-topic or not
2. deciding on which URL to visit next
3. avoiding multiple agents downloading same page

The first issue is considered by retrieval agents. The last two issues are considered by coordinator agent. Our system concentrates in using an intelligent multi-agent based system that mines the information contained in both hyperlinks and content. The advantages are two fold in using two different agents for collecting information. One is to reduce the network traffic load and another is to parallelize the computation. The crawling system is fed with the inputs such as seed URLs, and topic specific query terms. With these inputs the crawler must predict the next URL to download.

The coordinator agent functions as an interface between the crawler agents. The interface collects the seed URLs from a trusted search engine like Google. An expert intervention is necessary in this step to identify the most important seed pages from millions of search results from the search engine. The URLs are then placed in the frontier queue along with the relevant scores given by the expert. The coordinator agent maintains four levels of frontier queues. First level contains the URLs which are relevant to the topic of search. Second level contains the URLs that are less relevant but following a URL in that level will lead to a relevant page. Similarly the remaining levels of frontier queues are implemented to achieve tunneling.

Initially, the coordinator agent spawns number of retrieval agents equal to the number of URLs in the first level of frontier queue. The queues in the lower levels are empty. The retrieval agents start downloading pages, parse them, and are given as input to the Naïve-Bayes classifier. The classifier in each retrieval agent classifies the page and returns the relevance scores which are then updated in the frontier queue.

As in the rule based system exploiting inter-class relationships proposed by Altingövde et al. (2004), the relevance scores are computed based on associations among the classes. In this proposed system, rules are generated and normalized relevance scores are computed for each downloaded page as follows:

$$R[p] = NHyp_{i\rightarrow i} / NHyp_{i\rightarrow i}\ NHyp_{i\rightarrow j} \qquad (1)$$

where $NHyp_{i\rightarrow i}$ is number of hyperlinks in a page that belong to the same class and $NHyp_{i\rightarrow j}$ is number of hyperlinks that belong to other classes. Instead of considering the probabilities, the counts are taken into account. This computation gives 20% more efficiency in finding quality Web pages. The URLs with updated relevance scores and a value for level attribute is communicated to coordinator agent. It performs URL seen test and inserts the parsed URLs in the appropriate queues with the help of values in level attribute. Since the frontier queue is global and the assignment of URL is centralized, there are little chances for downloading duplicate pages by different crawler agents. The resultant pages that are classified as relevant are stored in the database for later processing.

4. Discussion

The architecture described above is implemented using JADE framework. JADE is one of the promising agent development frameworks supporting the deployment of multiple agents. Each of the agents can dynamically discover other agents and can communicate according to peer-to-peer paradigm (Nikraz et al., 2006). FIPA ACL is used as the communication language.

For training the Naïve-Bayes classifier, Yahoo! topic taxonomy is used. The performance of the proposed crawler design is compared with two baseline crawlers:

(i) baseline crawler without rule based system

(ii) baseline crawler with single level frontier queue

One of the parameter to evaluate the performance of the crawler design is the harvest ratio which is defined as the average relevance of all pages retrieved on a particular topic.

$$\text{Harvest ratio} = \frac{\sum_{i=1}^{N} R(p_i)}{N} \qquad (2)$$

The performance comparison of all the three crawlers is shown in Figure.2.

The graph clearly depicts that the proposed crawler initially has low harvest rate. Two reasons were observed by carrying over the experiments repeatedly. They are:

1. time consumed in initial training of the classifier
2. time consumed in initial coordination among all retrieval agents for receiving URLs to store in different levels of queues

Besides these two observations, the newly proposed crawler outperforms both.

Another observation is made on the crawler design which is the time taken by the crawlers to download upto 7000 pages. The graph is depicted in Figure. 3.

The Figure. 4 depicts the time taken for downloading 1000 pages with different levels of queues. The graph shows the advantage that is gained due to implementation of frontier queues in multiple levels. It is much reduced when using 4 levels of frontier queues. This is due to the fact that the retrieval agents need not wait for the master agent to dynamically assign URLs. Implementing tunneling in different levels also increases the number of relevant pages harvested. The observation recorded is shown in Figure. 5.

The experimentation is done on different levels of frontier queues. It was found that the relevance scores of the harvested pages increased till the number of levels was 4. Adding one more level to the queue decreases the relevance score as topic of the low quality pages drifted from the topic of search.

5. Conclusions

This paper has discussed the issues of designing a focused crawler. The fast improvements in the computational tools help in framing a novel architecture for locating relevant information in the rapidly growing Web. This paper proposed a novel framework of implementing multi-level queues as URL frontier to realize tunneling concept. The proposed system employed multiple agents communicating with each other to achieve the goal of finding quality results. The system also employs a mechanism to avoid duplicate downloading of Web pages. The empirical results suggest that the relevance scores of the pages downloaded is quiet high and the time taken to download those relevant pages is less. This system can be improved further to alleviate the bottleneck due to the centralized task allocation by the coordinator agent.

References

Paul De Bra, Geert-Jan Houben, Yoram Kornatzky, Reinier Post (1994), Information Retrieval in Distributed Hypertexts, *Proc. 4th Int'l Conf. Intelligent Multimedia Information Retrieval Systems and Management (RIAO 94)*, Center of High Int'l Studies of Documentary Information Retrieval (CID), pp. 481–491.

M Hersovici, M Jacovi, Y S Maarek, D Pelleg, M Shtalhaim, S Ur (1998), The SHARKSEARCH Algorithm—An Application: Tailored Web Site Mapping, *Computer Networks and ISDN Systems*, vol. 30, nos. 1–7, pp. 317–326.

J. Cho, H. Garcia-Molina, and L. Page (1998), Efficient Crawling through URL Ordering, *Computer Networks and ISDN Systems*, vol. 30, nos. 1–7, pp. 161–172.

S. Chakrabarti, M.H. Van den Berg, and B.E. Dom (1999), Focused Crawling: A New Approach to Topic-Specific Web Resource Discovery, *Computer Networks*, vol. 31, nos. 11–16, pp. 1623–1640.

M. Diligenti et al. (2000), Focused Crawling Using Context Graphs, *Proc. 26th Int'l Conf. Very Large Data Bases* , Morgan Kaufmann, pp. 527–534.

McCallum et al. (1999), Building Domain-Specific Search Engines with Machine Learning Techniques, Proc. *AAAI Spring Symp. Intelligent Agents in Cyberspace*, AAAI Press, pp. 28–39.

S. Mukherjea (2000), WTMS: A System for Collecting and Analyzing Topic-Specific Web Information, *Proceedings of the 9th International World Wide Web conference on Computer networks : The International Journal of Computer and Telecommunications Networking*, vol. 33, nos. 1–6, pp. 457–471.

S. Sizov, J. Graupmann, and M. Theobald (2000), From Focused Crawling to Expert Information: An Application Framework for Web Exploration and Portal Generation, *Proc. 29th Int'l Conf. Very Large Data Bases (VLDB 2003*), Morgan Kaufmann, pp. 1105–1108.

F. Menczer, G. Pant, and P. Srinivasan (2003), Topical Web Crawlers: Evaluating Adaptive Algorithms, *ACM Trans. Internet Technology*, vol. 4, no. 4.

A. Abdollahzadeh Barfouroushi, M.L. Anderson, H.R. Motahary Nezhad, D. Perlis (2002), Information Retrieval on the World Wide Web and Active Logic: A Survey and Problem Definition, Technical Report, pp 1-45.

Charu C. Aggarwal, Fatima Al-Garawi, Philip S. Yu (2001), Intelligent crawling on the World Wide Web with Arbitrary Predicates, *Proceedings of the 10th International Conference on WWW*, ACM Press, pp 96-105.

C. Olivia, C. Change, C. F. Enguix, A.K. Ghose (1999), Case-Based BDI Agents: An Effective Approach for Intelligent Search on the web, *Proceeding AAAI-99, Spring Symposium on Intelligent Agents in Cyberspace,* Stanford University, USA.

M. R. Genesereth, R. E. Fikes (1992), Knowledge interchange format, version 3.0. Technical Report 92-1, Stanford University, Computer Science Departament.

T. Finin, R. Fritzson, D. McKay, R. McEntire (1994),KQML as an Agent Communication Language, *Proceedings of the 3rd International Conference on Information and Knowledge Management* (CIKM'94), ACM Press, pp 456-463

Y. Labrou, T. Finin, Y. Peng (1999), The current landscape of Agent Communication Languages, *The current landscape of Agent Communication Languages, Intelligent Systems*, volume 14, number 2, March/April 1999, IEEE Computer Society.

Ismail Sengör Altingövde and Özgür Ulusoy (2004), Exploiting Inter-Class rules for Focussed Crawling, *IEEE Intelligent Systems*, Volume 19, Issue 6, pp 66-73.

M. Nikraz, G. Caire, and P. A. Bahri (2006), A Methodology for the Analysis and Design of Multi-Agent Systems using JADE, *International Journal of Computer Systems Science & Engineering, special issue on Software Engineering for Multi-Agent Systems*.

Marc Ehrig, Alexander Maedche (2003), Ontology-focused Crawling of Web Documents, *In Proceedings of the ACM symposium on Applied computing*.

Figures

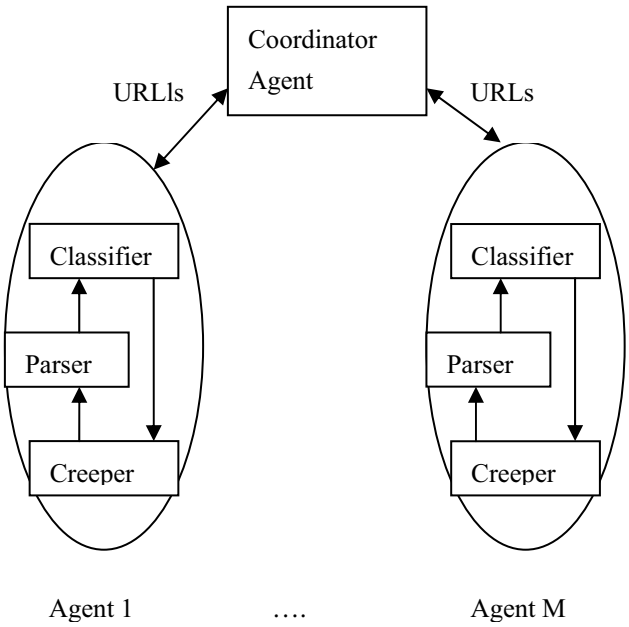

Figure 1. Architecture of the Web crawler

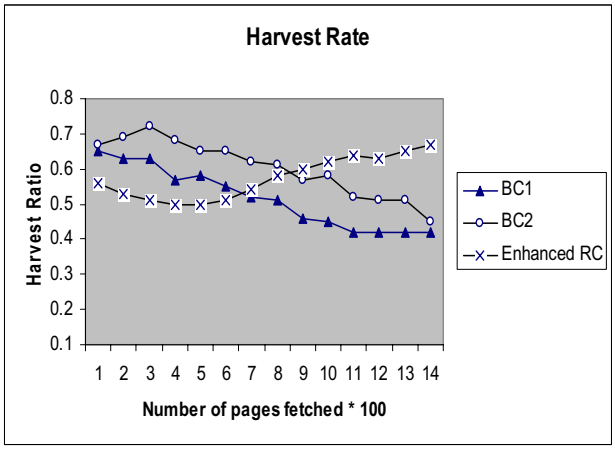

Figure 2. Harvest Ratio

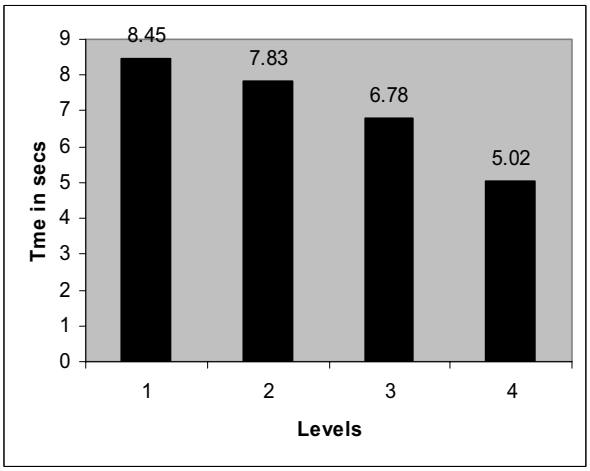

Figure 3. Time taken to download pages

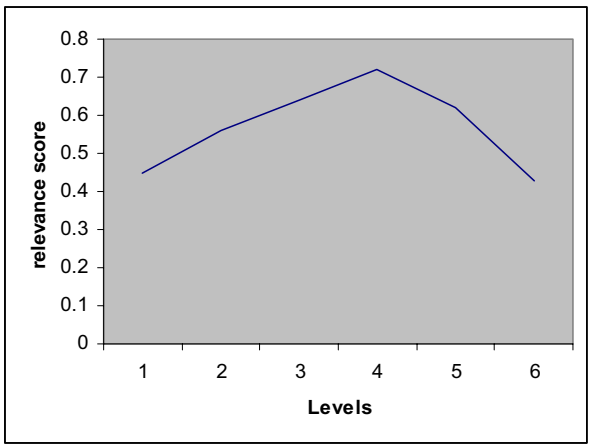

Figure 4. Number of levels in frontier Vs time taken to download 100pages

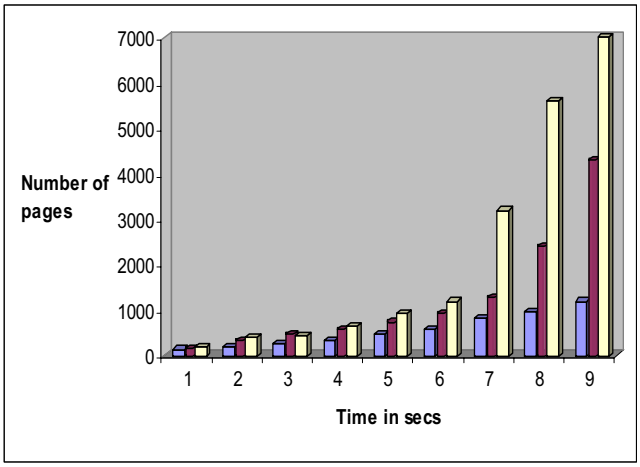

Figure 5. Number of levels Vs relevance scores