An extended model for effective migrating parallel web crawling with domain specific crawling

Mahesh Singh¹, Navita²
¹Mentor, ²Student
¹²Department of Computer Science and Engineering, Maharishi Dayanand University, Rohtak, India

Abstract: The size of the internet is large and it had grown enormously search engines are the tools for Web site navigation and search. Search engines maintain indices for web documents and provide search facilities by continuously downloading Web pages for processing. This process of downloading web pages is known as web crawling. In this paper we propose the architecture for Effective Migrating Parallel Web Crawling approach with domain specific and incremental crawling strategy that makes web crawling system more effective and efficient. The major advantages of migrating parallel web crawler are that the analysis portion of the crawling process is done locally at the residence of data rather than inside the Web search engine repository. This significantly reduces network load and traffic which in turn improves the performance, effectiveness and efficiency of the crawling process. The another advantage of migrating parallel crawler is that as the size of the Web grows, it becomes necessary to parallelize a crawling process, in order to finish downloading web pages in a comparatively shorter time. Domain specific crawling will yield high quality pages. The crawling process will migrate to host or server with specific domain and start downloading pages within specific domain. Incremental crawling will keep the pages in local database fresh thus increasing the quality of downloaded pages.

Keywords: Web crawling, parallel migrating web crawler, search engine.

INTRODUCTION

The Internet is a global system of interconnected computer networks. The searching and indexing tasks for the web are currently handled from specialized web applications called search engines. The modern search engines can be divided into three parts they are the publically available search engine, the data-base and the web crawling system. A web crawler is an automated program that browses the Web in a methodological manner. The process of traversing the web is called Web crawling. Web crawler starts with a queue of known URLs to visit. As it visits these pages, it scans them for links to other web pages. The Web Crawler consists of Crawler Manager, Robot, Txt downloader, Spider and Link Extractor. A Crawl Manager takes a set of URLs from Link Extractor and sends the next URLs to the DNS resolver to obtain its IP address. Robot.txt file are the means by which web authors express their wish as to which pages they want the crawler to avoid. Link extractor extract URLs from the links in the downloaded pages and sends the URLs to the crawler manager for downloading afterwards.

LITERATURE SURVEY

In [13], the author demonstrated an efficient approach to the “download-first process-later” strategy of existing search engines by using mobile crawlers. In [14] author has implemented UbiCrawler, a scalable distributed and fault-tolerant web crawler. In [15] author presented the architecture of PARCAHYD which is an ongoing project aimed at designing of a Parallel Crawler based on Augmented Hypertext Documents. In [16] the author studied how an effective parallel crawler is designed. As the size of the Web grows, it becomes imperative to parallelize a crawling process. In [17] the author proposes Mercator, which is a scalable, extensible crawler. [18] Presented Google, a prototype of a large scale search engine which makes heavy use of the structure present in hypertext. [19] Aims at designing and implementing a parallel migrating crawler in which the work of a crawler is divided amongst a number of independent.

MIGRATING PARALLEL WEB CRAWLER

The Migrating Parallel Crawler system consists of Central Crawler, Crawl Frontiers, and Local Database of each Crawl Frontier and Centralized Database. It is responsibility of central crawler to receiving the URL input from the applications and forwards the URLs to the available migrating crawling process. Crawling process migrated to different machines to increase the system performance. Local database of each crawl frontier are buffers that locally collect the data. This data
is transferred to the central crawler after compression and filtering which reduces the network bandwidth overhead. The central crawler has a centralized database which will contain the documents collected by the crawl frontiers independently. The main advantages of the migrating parallel crawler approach are Localized Data Access, Remote Page Selection, Remote Page filtering, remote Page Compression, Scalability, Network-load dispersion, Network-load reduction.

**ISSUES IN MIGRATING PARALLEL WEB CRAWLER**

The following issues are important in the study of a migrating parallel crawler interesting:

**Communication bandwidth:** Communication is really important so as to prevent overlap, or to improve the quality of the downloaded content, crawling processes need to communicate and coordinate with each other to improve quality thus consuming valuable communication bandwidth.

**Quality:** Prime objective of migrating parallel crawler is that to download the “important” pages first to improve the “quality” of the downloaded pages.

**Overlap:** When multiple processes run in parallel to download pages, it is possible that different processes download the same page multiple times. One process may download the same page that other process may have already downloaded, one process may not be aware that another process has downloaded the same page.

**COORDINATION IN MIGRATING PARALLEL WEB CRAWLER**

The coordination is done in order to prevent overlap and also crawling processes need to coordinate with each other so that the correct and quality pages are downloaded. This coordination can be achieved in following ways:

**Independent:** In this method Crawling processes may download pages independent of each other without any coordination. The downloaded pages may overlap in this case, but we assume that overlap is not significant enough.

**Static assignment:** In this method the Web is partitioned into sub partitions and each sub partition is assigned to each Crawling processes before the actual crawling process may begin. This type of arrangement is called static assignment.

**Dynamic assignment:** In this coordination scheme there is a central coordinator that divides the Web into small partitions and each partition is dynamically assigned to a Crawling processes for further downloading.

**PROPOSED ARCHITECTURE OF MIGRATING PARALLEL WEB CRAWLER**

A migrating parallel crawler consists of several multiple crawling processes which migrate to source of data for example Web server or host before the crawling process is actually started on that Web server or host. Migrating parallel crawlers move to resource and take the advantages of localized access of data. Migrating parallel crawler after accessing a resource migrates to the next host or server or to their home system. Each migrating parallel crawling process performs the tasks of single crawler that it downloads pages from the Web server or host, stores the pages in the local database, extracts URLs from the content of downloaded pages and follows the extracted URLs or links. When Crawling process run on the same local area network and communicate through a high speed interconnection network then it is known as intra-site migrating parallel web crawler. When Crawling process run at geographically distant locations connected by the Internet or a wide area network then it is known as distributed migrating parallel web crawler. The architecture consists of central coordinator system and crawling.

1. **Central Coordinator System**

Central coordinator system consists of Central Crawl Manager, Crawling Application, Web Cache, Central Database or Repository, URL Dispatcher, URL Distributor, Domain Selector. Central crawl manager is the central part of the crawler all other components are started and register with the central crawl manager to offer or request services. It acts as the core of the system. Central Crawl Manager’s objective is to download pages in the order specified by the crawling application. The central crawler has a list of URLs to crawl. After getting the URLs of files, central crawl manager queries the DNS resolvers for the IP addresses of the host or servers. The central manager then takes into consideration robots.txt files in the root directory of the web server. The central crawler a set of available crawling process which have registered themselves with the central crawl manager which logically migrated to different domain specific locations. The central crawl manager assigns different URLs to all the crawl process in domain specific manner and in turn the crawling processes begin to crawl the received URL in breadth first fashion and pages are downloaded by multi-threaded downloaded. It is the responsibility of the crawling application to check for duplicate pages. The
crawling application considers in this paper is a breadth-first crawl strategy. Breadth first search gives high quality pages thus improving the quality of downloaded pages. The crawling process actually crawls in breadth first manner. The DNS resolver uses the GNU and asynchronous DNS client library to access a DNS server usually collocated on the same machine. A web cache stores web resources in anticipation of future requests. Web cache works on the principle that the most popular resource is likely to be requested in the future. The advantages of Web Cache are reduced network bandwidth, less user perceived delay, less load on the origin servers. Web cache provides quick response time. When the information resides in web cache, the request is satisfied by a cache, the content no longer has to travel across the Internet. Central database or Repository stores the list of URLs received and downloaded web pages which are collection of the documents downloaded by crawling process. Web pages can be stored in compressed form. Each document is provided with a unique number called document identifier denoted by doc_id. The documents are stored along with doc_id, length of document and URL. This helps with data consistency and makes data storage much easy and efficient. Data is stored in the central data base in hierarchical tree manner. Each node or the parent node of the tree is the domain and the child node of the parent node is the sub domain. Each node consists of domain name and its corresponding URLs. The database is continuously updated for future use by adding new URL. Central Database communicates with the database of the search engine. URL dispatcher reads the URL database and retrieves the URL from the index table maintained in the central repository and forwards it for crawling. URL distributor classifies the URLs on the basis of domains. It then distributes the URL to the concerned domain specific crawling process for crawling. The URL distributor balances the load on the crawling process. Domain selector identifies the domains of the links which are extracted by the link extractor. The links belonging to specific domain are stored in the corresponding domain table. The domain selector also forwards the domain specific URL to the URL dispatcher for further crawling.

2. Crawling Process

Crawling process consists of Scheduler, New ordered Queues, Site ordering module, URL Queues/ Known URLs, Multi-threaded Downloader, URL Collector, Link Extractor, Link analyzer. Scheduler provides the set of URLs to be downloaded based on the modified page rank. The URLs are stored in new ordered queues. New ordered Queues stores the set of URLs based on modified page rank. Site ordering module provides the modified page rank of the page. Known URLs are the set of already known URLS. They can be treated as seed URLs. Multi-threaded downloader takes a URL from URL collector and downloads the corresponding webpage to store it in the local repository or database. The downloader component fetches files from the web by opening up to connections to different servers. URL collector keeps the URL from the downloaded pages. Link Extractor extracts the link from the downloaded pages. Link analyzer checks the extracted URLs by the link extractor. If there is similarity in the URL then such URLs are discarded and not further forwarded for processing. When the crawling process runs it requires some memory space to save the downloaded web pages. Each crawling process has its own local storage.
The crawling process saves the downloaded web pages in this database. It is the storage area of the machine on which the crawling process is running. The Ranking Module constantly scans the known URLs and the local database to make the refinement decision. The Ranking Module refines the local database. The Ranking Module discards the less important page from the local database to make space for the new page. Locally collected URLs are the set of URLs in the local collection. The Update Module maintains the local database fresh. In order to maintain the local database fresh, the crawler has to select the page that will increase the freshness most significantly this decision is known as update decision.

**Result**

The crawled pages will be domain specific. Fresh pages will be downloaded as change frequency is taken into consideration. High quality of pages will be downloaded as crawling processes are performing in breadth first manner. Breadth first crawling improves the quality of downloaded pages

**Conclusion**

In this paper we proposed an Extended Model for Effective Migrating Parallel Web Crawling with Domain Specific. Domain specific crawling will yield high quality pages. The crawling process will migrate to host or server with specific domain and start downloading pages within specific domain. Incremental crawling will keep the pages in local database fresh thus increasing the quality of downloaded pages. The research directions in migrating parallel crawler include:

- Security could be introduced in migrating parallel crawlers
- Migrating parallel crawler could be made polite
- Location awareness could be introduced in migrating parallel crawlers

This future work will deal with the problem of quick searching and downloading the data. The data will be collected and analyzed with the help of tables and graphs.

**References**