Implementation of Web Crawler for Mobile Search Using Mobile Agent

Mona, Monika, Prof. Ela Kumar

M.Tech (CSE) student, Indira Gandhi Delhi Technical University, India

monali.mona08@gmail.com

M.tech(CSE)student, Indira Gandhi Delhi Technical University, India

monika09.1992@gmail.com

Professor of Computer Science and Engineering Department, Indira Gandhi Delhi Technical University, India

ela_kumar@rediffmail.com

Abstract— In the present world, presence of billions of web data on WWW poses a huge challenge for web search engines in terms of efficiency and reliability. Web Crawler is the heart of a search engine. Web crawler is a software program that traverses web, collects web data available on internet and index the crawled data. The indexed data can be further retrieved by the user depending upon the query entered. Search engines need to have their indexes and data repositories updated as per the web resources hosted in web servers. According to CiscoVNI forecast [1] there is a rapid progress in global mobile data traffic exceeding 2.5 exabytes at the end of year 2014. To crawl this huge amount of mobile data an efficient mobile web crawler is introduced to fetch data to be viewed on mobile handheld devices like tablets, smartphones etc.

In this paper the architecture of our mobile web crawler that fetches the relevant data to be viewed on handheld terminals is presented. Mobile web crawler also known as mobile agent is implemented using java aglets introduced by IBM. The major advantage of using mobile agent for crawling is that it can transit from one platform to another platform depending upon where the relevant data is present and brings back the crawled result to the host machine.

Keywords— Web Crawler, World Wide Web, mobile agent, handheld device, Java aglets, search engine, server.

INTRODUCTION

The enormous size, dynamic nature and decentralized control over its web content are the three main reasons for the success of World Wide Web. Unluckily, the same issues become drawbacks when the concern is locating relevant information in desired time. This is because the quality information i.e. relevant data is highly decentralized as compared to rest of the web content available. Thus to manage the huge amount of data available on web and to retrieve the relevant information in a more efficient manner, a software known as search engine has been designed. A search engine is a highly sophisticated piece of software that can be accessed through a page on a website. The search engine allows user to search the web by entering the search query in the search box. It uses keyword matching technique to search the data and then display the result according to the relevancy of the information that was searched for. Web search engines make use of a web crawler to download pages from WWW. These downloaded web pages are stored and indexed in search engine's data repository for efficient data retrieval.

Because of the dynamic nature of web, its contents change every second. Hence, to maintain up to date indexes, a crawler needs to traverse the web many times. More the revisiting of web more will be the chance of internet overloading and hence sometimes a website gets collapsed. According to a study [1] the current web crawlers have navigated, downloaded and indexed billion of web pages and is responsible for 41% internet traffic and bandwidth spending.

In this paper a mobile web crawler is introduced that collects data to be viewed on mobile handheld devices like smart phones, tablets, PDAs. A report from market research firm global Web index [2] says that nearly 80% of people worldwide now own a smart phone and almost all smart phone users are using their devices to access the internet. Mobile Web (red dots and curve [3], Figure 1) may catch up with desktop Web (blue dots and curve, Figure 1) by July or August 2015, according to areppim's forecast [4] based on the currently available data. Mobile Web's 30% world market share may seem unthreatening compared to the 69% share of desktop Web. In reality it is moving up very fast, at the rate of 5.93% per month (doubling in size every 12 months), while desktop Web is steadily losing ground at the average monthly rate of 0.52%.

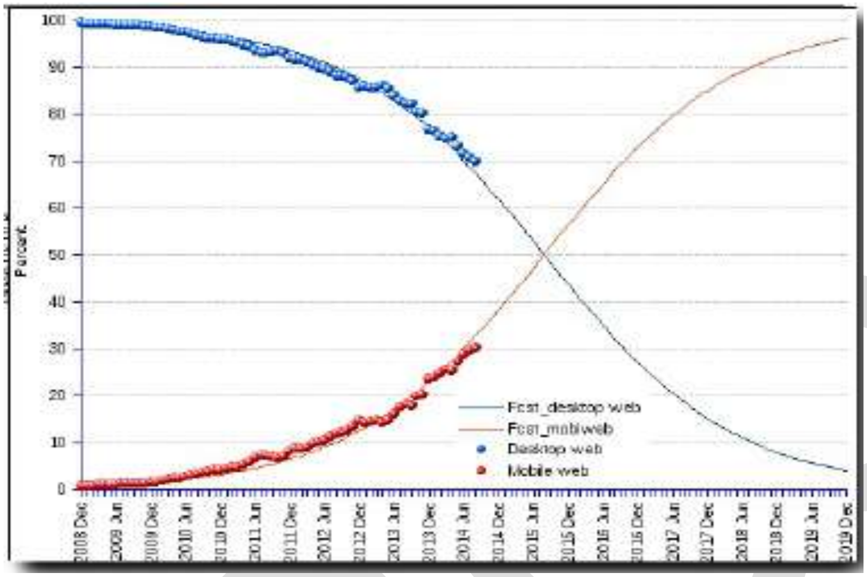


Figure 1: Mobile and desktop Web may have 50% market share each by mid 2015 [3].

Hence we are concentrating on a specialized crawler that collects mobile compatible data for viewing on mobile terminals efficiently. This specialized mobile web crawler is based on mobile agents implemented using Java aglets introduced by IBM. Benefits of using mobile agent are that it is capable of reducing network load and bandwidth wastage caused by traditional web crawlers.

RELATED WORK

A traditional search engine has four major modules named as crawler module, indexing module, ranking module, querying module. Crawling module is an inevitable part of a search engine. Web crawler is a program that spiders the web on behalf of search engine. A web crawler starts with a list of seed URLS, identifies all the hyperlinks in the currently downloaded page and add them to the list of URLs known as crawl frontier for further visiting. Now, the crawler will extract URLs from the crawl frontier and crawls them in a recursive manner according to a set of rules. A crawler follows links to reach different web pages and download them to save in page repositories. The indexing module uses page repositories to strip contents from the downloaded web pages and extract key elements like title tag or description tags etc. There are two main responsibilities of a crawler, one is downloading the new web pages and another is refreshing the earlier downloaded pages. To maintain up to date indexes a web crawler needs to revisit websites recursively and many times. Due to huge number of revisits properties like network bandwidth, disk space etc get increased and thus overloading the internet.

To reduce network load and to save network bandwidth we have implemented a web crawler using mobile agent i.e. mobile crawlers. A mobile agent has the ability to migrate from one platform to another in an autonomous manner. The unique property of mobile agents to do the selection and filtration of web pages at the server side rather than search engine side reduces network load and increases search efficiency.

ARCHITECTURE OF SEARCH ENGINE

Search engines are needed because with over eight billion web pages available, it would not be possible to search for relevant information. This is why search engines are used for filtering the information and transform it into results that increase ease of information access for the user. A Crawler based search engine creates their listings automatically. These types of search engines employ a "spider" or a "crawler" to search the Internet. The crawler digs through individual web pages, extracts keywords and then appends the pages to the search engine's database. If changes are made to web pages, crawler-based search engines eventually find these changes, and further change the URLs listings.

Every search engine comprises of six main modules but among those crawler module is the inevitable part because it helps to provide best possible results to the search engine. Other basic modules of a search engine are cloud, page repository, indexing module, query module and ranking module. These components are described in the following [8].

- Cloud: This represents the WORLD WIDE WEB.

- Crawler module: It sends, crawlers or spiders to web for crawling the websites and extract data back and put them in page repository. Crawler is software programs that starts by fetching few web pages and then follows the links on those pages and fetches the pages they link to and so on.

- Page Repository: Web pages retrieved by the crawler are stored in web page repository. It stores the web pages temporarily and these web pages remain there until they send them to indexing module.

- Indexing Module: This module strips the content from web pages in page repository and a particular key i.e. pieces of contents (key element are title tag, description tag, images or internal links) provides content summary of each page. Indexing modules pushes data in the form of indexes. These indexes can be of different kinds like content, video, image index.

- Query Module: When a query is typed, it is send to query module which breaks down this query into a language that search engine can understand. It will extract thousands and thousands of results from indexes and pass all these results to ranking module.

- Ranking Module: This module has the job of filtering and putting results in ranks according to the relevancy. The ranking module works on different algorithm to extract the content and then looks up the popularity score and combines those together and send back to search engine page.

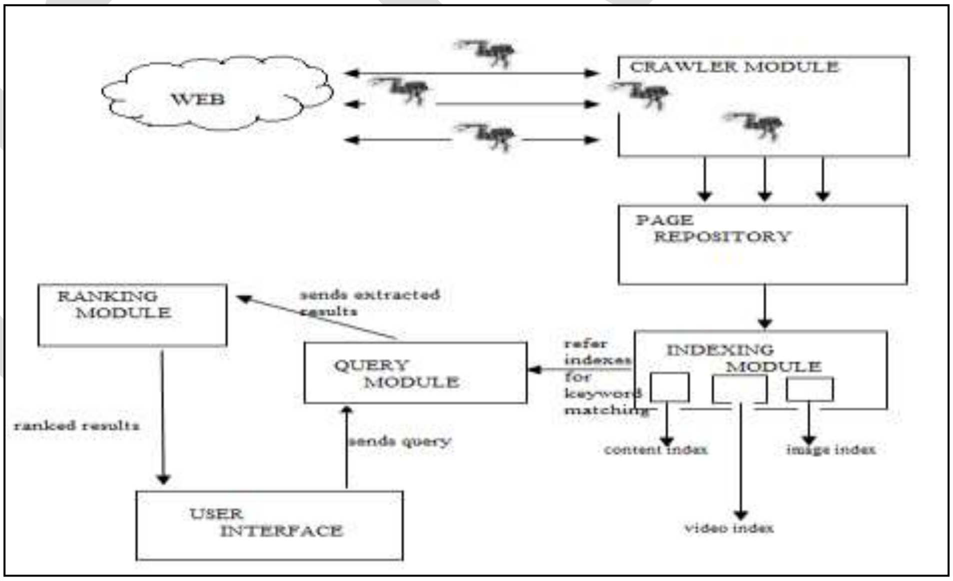


Figure 2: Working of a search engine.

MOBILE AGENT

Mobile agent is a software programs having the ability to halt its execution on one platform and automatically migrate to another platform where it resume execution to complete its task.

Mobile agents also known as mobile crawlers migrate to the data source before the crawling process actually get started on that particular server. After accessing the relevant data, mobile crawler either move to the next resource or else to the host platform carrying the result back in filtered and compressed form. Mobile agents perform selection and filtration of web pages at the server side rather at the search engine side. This local accessing of data reduces network load due to HTTP request caused by traditional web crawlers. Details of mobile crawler are discussed by [9].

Mobile Agent System [10] consists of two main components: mobile agents and mobile agent platforms. Mobile agent platforms are the execution environments on different platforms for mobile agents. Mobile agents is an autonomous software program having the ability to migrate from one platform to another platform for local data accessing hence reducing traffic load due to HTTP request. In our work mobile agent is implemented using IBM's Java Aglets.

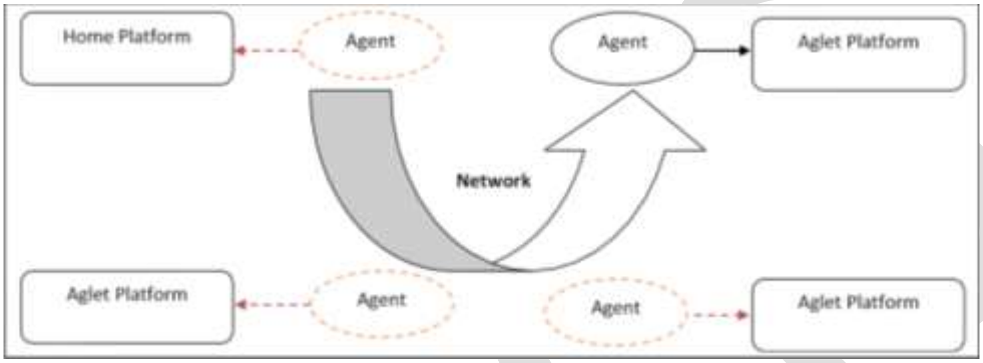


Figure 3: Mobile agent system [10].

AGLET LIFECYCLE MODEL

An aglet is a Java-based autonomous software agent. A software agent is program that can suspend its execution on one platform, migrate itself to another platform on the network and resume its execution at the new platform. Aglets are autonomous because they are independent to decide where to go and what to do. An aglet carries its current state wherever it goes and eventually returns back to its host platform carrying the data along with state. A java aglet is similar to an applet, the only difference is that applets are stateless whereas aglets are stateful.

The different events in aglet lifecycle model [5], [10] are as follows:

- **Created:** A brand new aglet is born,- its state is initialized, its main thread starts executing
- **Cloned:** A twin aglet is born - the current state of the original is duplicated in the clone
- **Dispatched:** An aglet travels to a new host - the state goes with it
- **Retracted:** An aglet, previously dispatched, is brought back from a remote host - its state comes back with it
- **Deactivated:** An aglet is put to sleep - its state is stored on a disk somewhere
- **Activated:** A deactivated aglet is brought back to life - its state is restored from disk
- **Disposed:** An aglet dies - its state is lost forever

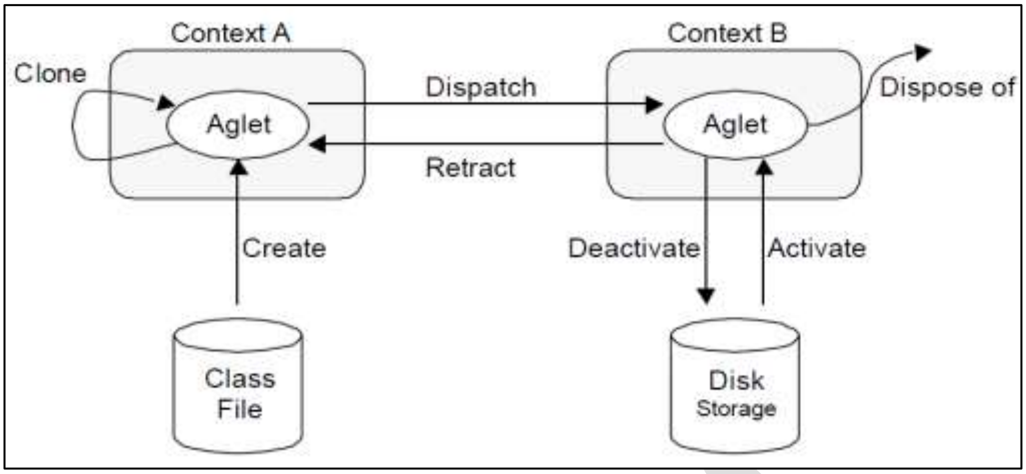


Figure 4: State Diagram of Aglet Lifecycle [16].

ADVANTAGES OF MOBILE AGENT

The main advantage of using mobile agent is its distributed crawling functionality. Other advantages [9] of mobile agents are as follows:

1. Localized Data Access: Use of mobile crawlers reduces HTTP requests overhead by migrating itself to the source of data. The crawler then issues all HTTP requests locally with respect to the HTTP server. This approach reduces network traffic and saves bandwidth.

2. Remote Page Selection: Traditional crawlers downloads complete database before issuing queries to search relevant information. This traditional crawling approach increases redundancy and thus network traffic. In contrast to this, mobile crawlers move to the source site, executes query remotely and then sends only the query result over the network thereby eliminating processing part at search engine side.

3. Remote Page Filtering: A mobile crawler filter out irrelevant information remotely keeping only the relevant data. This feature of mobile agent saves network bandwidth and reduces web traffic by transmitting the relevant data only.

4. Remote Page Compression: A mobile crawler introduces data compression feature at the remote site. Mobile crawler applies data compression algorithm like gzip to reduce the size of data to be transmitted over the network. This feature reduces network bandwidth hence making it an attractive approach over traditional crawling method.

DISTINGUISHING MOBILE COMPATIBLE DATA

Each content provider has its own specifications to represent mobile content. To identify mobile content to be viewed on mobile handheld devices like smart phones, PDAs etc. we have used “viewport” meta tag. Space within the browser window which is affected by monitor resolution is known as viewport. Viewport (window) control the viewport’s size and scale.

A typical mobile compatible website contains viewport meta tag as follows:

<meta name = “viewport” content =”width=device-width, initial-scale=1, maximum-scale=1, user-scalable=0”>

- Size of viewport is controlled by width property

- When the page is loaded first time, the zoom level is controlled by initial-scale property.

- Users hold to zoom the page in and out is controlled by properties like minimum-scale, maximum-scale and user-scalable.

IMPLEMENTATION OF CRAWLER FOR MOBILE SEARCH

A. Architecture of Web Crawler for mobile search using mobile agent

- A local server machine is being setup and web server software is loaded on it and our local server machine makes its services available to internet using 8000 port.
- Mobile web crawler is implemented using mobile agent that makes use of IBM Java Aglets for crawling.
- Mobile crawler allows search engine to send its representative i.e. an aglet to the data source.
- In our application the programmer instruct the crawler to migrate from web server at 5000 port to a web server at 5001 port in order to collect the relevant data.
- The special purpose of this specialized mobile web crawler is to provide high quality searches in academic domain and provides the data to be viewed on mobile terminals i.e. data must be mobile compatible.
- The relevant data crawled by the mobile crawler depending on the crawler specifications is stored in the local server available at 8000 port.
- The data is for viewing on mobile terminals, an android mobile application working as a client fetch data from local server available to the internet at 8000 port.

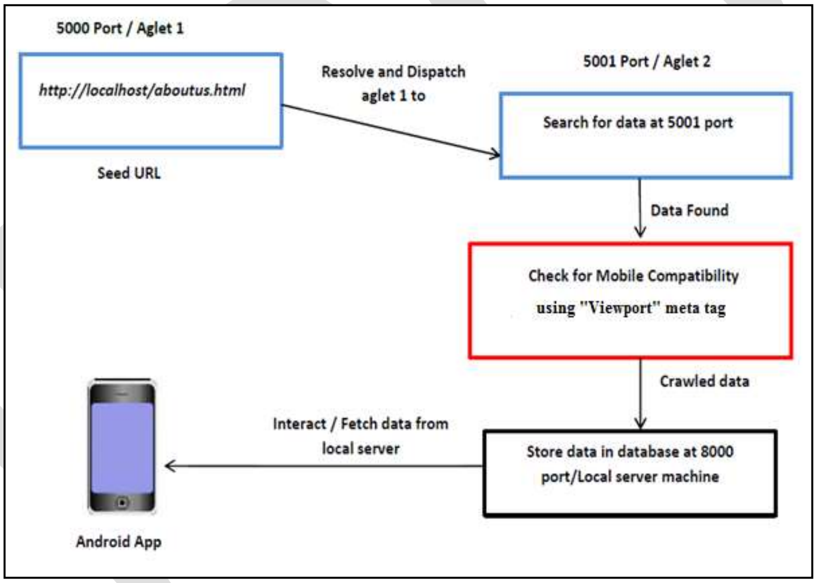
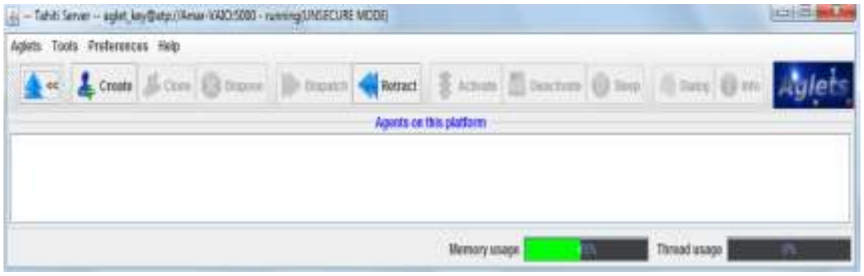

5000 Port / Aglet 1
http://localhost/aboutus.html
Seed URL
Resolve and Dispatch aglet 1 to
5001 Port / Aglet 2
Search for data at 5001 port
Data Found
Check for Mobile Compatibility using "Viewport" meta tag
Crawled data
Store data in database at 8000 port/Local server machine
Interact / Fetch data from local server
Android App

Figure 5: Architecture of crawler for mobile search

B. Implementation

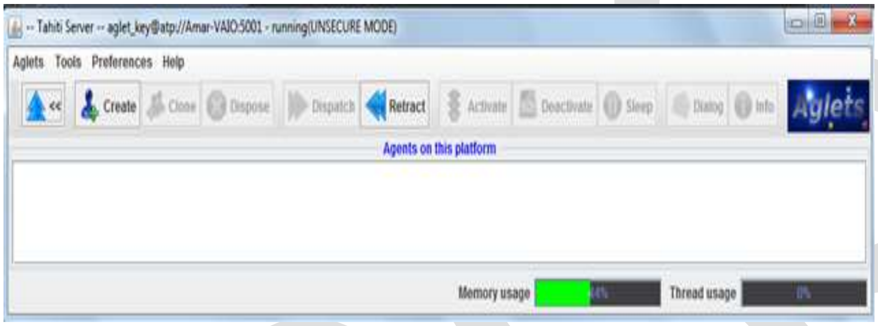
In this paper we have represented the design and working of a web crawler for searching mobile compatible data. This web crawler is implemented using mobile agent which is an autonomous software agent.

- First a local server machine is setup.
- Then we run an aglet application known as Tahiti. We can run multiple servers (Tahiti) on a single computer by assigning them different ports.



-- Tahiti Server -- aglet_key@atp://Amar-VAIO:5000 - running(UNSECURE MODE)

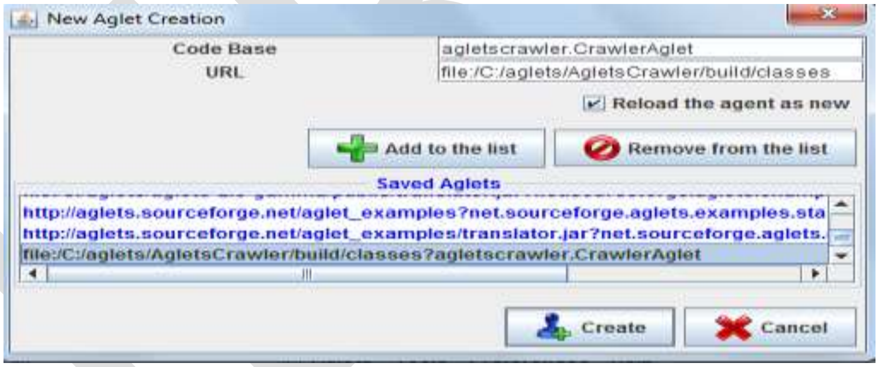
Figure 6: Aglet application Tahiti running at port 5000.



-- Tahiti Server -- aglet_key@atp://Amar-VAIO:5001 - running(UNSECURE MODE)

Figure 7: Aglet application Tahiti running at port 5001.

- Create an aglet



New Aglet Creation

Code Base: agletscrawler.CrawlerAglet

URL: file:/C:/aglets/AgletsCrawler/build/classes

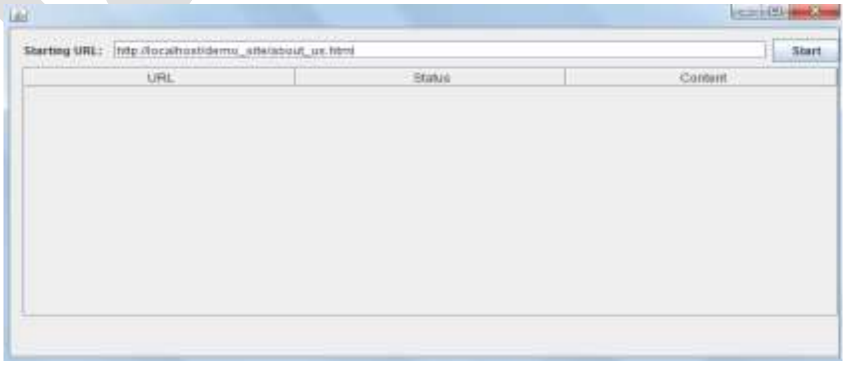
☑ Reload the agent as new

Saved Aglets

http://aglets.sourceforge.net/aglet_examples?net.sourceforge.aglets.examples.sta
http://aglets.sourceforge.net/aglet_examples/translator.jar?net.sourceforge.aglets.
file:/C:/aglets/AgletsCrawler/build/classes?agletscrawler.CrawlerAglet

Figure 8: Aglet Creation

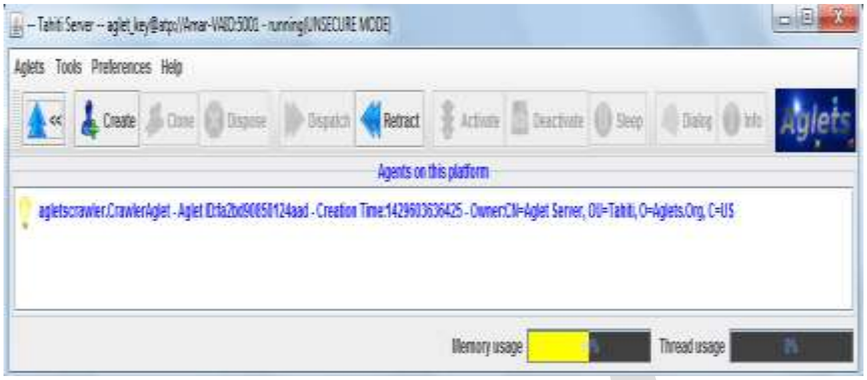
- Enter the seed URL and dispatch it.



Starting URL: http://localhost/demo_site/about_us.html

Figure 9: User Interface to enter seed url.

- Programmer instructs the crawler to migrate from web server at 5000 port to a web server at 5001 port.

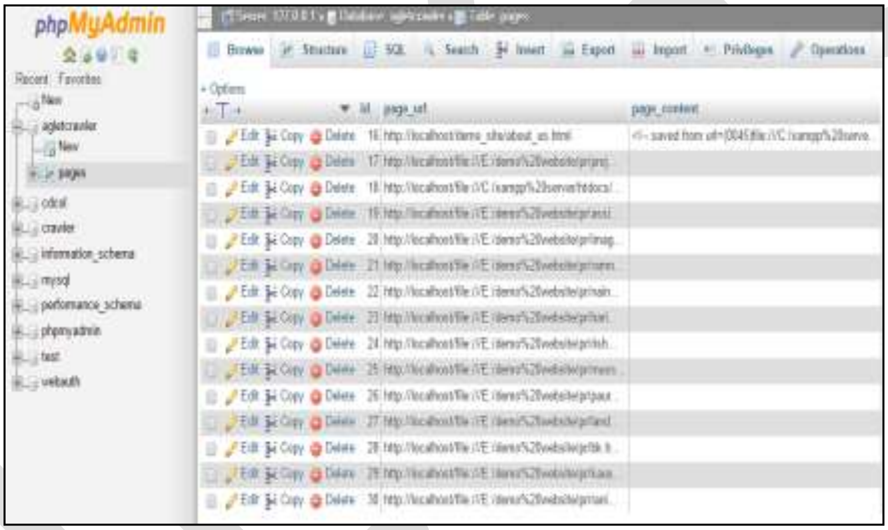


agletscrawler.CrawlerAglet - Aglet ID:fa2bd90850124aad - Creation Time:1429603636425 - Owner:CN=Aglet Server, OU=Tahiti, O=Aglets.Org, C=US

Figure 10: Aglet dispatched.

- The relevant data crawled by the mobile crawler is stored in the local server

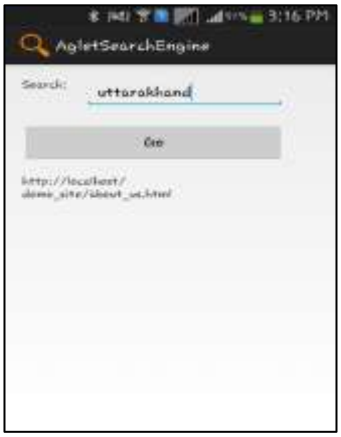
- Here we can see that a list of fetch web pages is created.



	id	page_url	page_content
Edit Copy Delete	16	http://localhost/demo_site/about_us.html	<!-- saved from url=(0045)file:///C:/xampp%20serve...
Edit Copy Delete	17	http://localhost/file:///E:/demo%20website/pr/proj...	
Edit Copy Delete	18	http://localhost/file:///C:/xampp%20server/htdocs/...	
Edit Copy Delete	19	http://localhost/file:///E:/demo%20website/pr/rasi...	
Edit Copy Delete	20	http://localhost/file:///E:/demo%20website/pr/imag...	
Edit Copy Delete	21	http://localhost/file:///E:/demo%20website/pr/amm...	
Edit Copy Delete	22	http://localhost/file:///E:/demo%20website/pr/main...	
Edit Copy Delete	23	http://localhost/file:///E:/demo%20website/pr/hari...	
Edit Copy Delete	24	http://localhost/file:///E:/demo%20website/pr/rish...	
Edit Copy Delete	25	http://localhost/file:///E:/demo%20website/pr/mans...	
Edit Copy Delete	26	http://localhost/file:///E:/demo%20website/pr/paur...	
Edit Copy Delete	27	http://localhost/file:///E:/demo%20website/pr/land...	
Edit Copy Delete	28	http://localhost/file:///E:/demo%20website/jetbk it...	
Edit Copy Delete	29	http://localhost/file:///E:/demo%20website/pr/kau...	
Edit Copy Delete	30	http://localhost/file:///E:/demo%20website/pr/nani...	

Figure 11: List of fetch URLs stored in local server.

- An android mobile application working as a client fetch data from local server



Search: uttarakhand

http://localhost/
demo_site/about_us.html

Figure 12: Fetching of data from local server using android application.

CONCLUSION

A Crawler is an inevitable part of a search engine. Web crawler is a program that makes searches on behalf of search engine. The traditional web crawlers navigate and download billions of web pages recursively many times to keep indexes updated thus responsible for huge internet traffic and bandwidth spending.

In this thesis work a web crawler for searching mobile compatible data to be viewed on handheld devices like smart phones, PDAs etc is designed and implemented. This crawler is implemented using mobile agent i.e. java aglets introduced by IBM. Aglet is a java based autonomous software having an ability to halt itself and ship to another resource on network, resume its execution there to complete its task. After completing its tasks an aglet moves to its host resource carrying result in its memory. As mobile agent has the property to migrate to the source and thus have local data access thereby reducing network traffic due to HTTP request. This crawler issues HTTP request locally, saving internet bandwidth. The web crawler is specifically designed to search mobile compatible data as mobile web is overpowering desktop web with a huge rate.

REFERENCES:

[1] http://www.cisco.com/
[2] B. Kahle, “Archiving the Internet,” Scientific American, 1996.
[3] http://tech.firstpost.com/news-analysis/more-than-half-of-the-worlds-population-owns-a-smartphone-report-249361.html
[4] StatCounter Global Stats [http://gs.statcounter.com/].
[5] Mobile to desktop web substitution forecast, 11 August 2014, available at: http://stats.areppim.com/
[6] http://www.javaworld.com/article/2077639/core-java/the-architecture-of-aglets.html
[7] http://bookofzeus.com/articles/html/quick-guide-to-mobile-devices-meta-tags/
[8] Brin S. and Page L., “The Anatomy of a Large Scale Hypertextual Web Search Engine,” Technical Report, Stanford University, Stanford, CA, 1997.
[9] Gautam Pant, Padmini Srinivasan, and Filippo Menczer, “Crawling the Web ” available at http://dollar.biz.uiowa.edu/~pant/Papers/crawling.pdf.
[10] Crawlers available at http://en.wikipedia.org/wiki/Web_crawler#Examples_of_Web_crawlers
[11] Christopher Olston and Marc Najork, “Web Crawling”, Foundations and TrendsR in Information Retrieval Vol. 4, No. 3 (2010) 175–246, 2010.
[12] Junghoo Cho and Hector Garcia-Molina, “Parallel Crawlers”, Proceedings of the 11th international conference on World Wide Web WWW '02”, May 7–11, 2002, Honolulu, Hawaii, USA, ACM 1-58113- 449-5/02/0005.
[13] Fiedler J. and Hammer J., “Using Mobile Crawlers to Search the Web Efficiently,” International Journal of Computer and Information Science, vol. 1, no. 1, pp. 36-58, 2000.
[14] A. Gulli, A. Signorini, "The indexable web is more than 11.5 billion pages"; ACM Press. pp. 902–903. doi:10.1145/1062745.1062789, 2005.
[15] Ioannis Avraam, Ioannis Anagnostopoulos; “A Comparison over Focused Web Crawling Strategies”, 2011 Panhellenic Conference on Informatics, IEEE.
[16] Md. Abu Kausar, Md. Nasar and Sanjeev Kumar Singh. “Maintaining the repository of search engine freshness using mobile crawler”, International Conference on Microelectronics, Communication and Renewable Energy (ICMiCR-2013), IEEE.
[17] Aglet Software link < aglets.sourceforge.net>
[18] Md. Abu Kausar, V S Dhaka and Sanjeev Kumar Singh. “Web Crawler: A Review.” International Journal of Computer Applications 63(2):31-36, February 2013. Published by Foundation of Computer Science, New York, USA.
[19] Md. Abu Kausar, V S Dhaka and Sanjeev Kumar Singh. “Web crawler based on mobile agent and java aglets” *I.J. Information Technology and Computer Science*, 2013, 10, 85-91**,** September 2013.
[20] Danny B. Lange and Mitsuru Oshima “Mobile Agents with Java: The Aglet API” is based on a chapter of a book by Lange and Oshima entitled

Programming and Deploying Mobile Agents with Java (Addison-Wesley)-1998