2015 International Conference on Computational Intelligence and Communication Networks

Framework for Distributed Semantic Web Crawler

Naresh Kumar
AIIT, Amity University, Noida , India
naresh.dhull@gmail.com

Manjeet Singh
YMCA University of Science & Technology,
Faridabad, India
mstomer2000@yahoo.com

Abstract: **Relevant information retrieval from the www mainly depends on the technique and efficiency of a crawler. So crawlers must be capable enough to understand the text and context of a link which they are going to crawl. Anchor text contains a very useful information to know about the target web page. Because knowledge about the target web page content helps the crawlers to decide their preferences of crawling the particular page. In this paper we have presented a design of distributed semantic web crawler capable of crawling both HTML and semantic web pages written using owl/RDf. In our crawler a component called page analyser is used to understand the theme of content of page and context of anchor tag in the page. The output of the page analyser is used to make crawling decisions. Our approach have revealed the great improvement in extracting the information from the links and guide the crawler for more relevant domain specific crawling.**

***Keywords*-Semantic Web, RDF, OWL, Semantic web Crawler, Distributed System**

I. INTRODUCTION

With the advent of semantic web meaningful information extraction gained the popularity in the recent years. Semantic web which is a extension of current web enable the machines and human to work together [2].With the help of technologies like RDF and OWL information in the world wide web is represented in the form of ontologies. Moreover the semantic information is increasing at a fast pace on the www. But still most of the information is in the form of text in the traditional web. To navigate through the www links called as anchor text are also provided in natural language text in traditional web pages. So to make the search more relevant more efforts are required to understand the text data in the traditional web pages. Crawler is a program which extracts the web pages from the www web. If crawler is able to extract more relevant pages search engine is able to provide more relevant results to the user query. To crawl the relevant pages different architectures are developed for semantic web pages ontology based crawlers like [11, 10, 9, and 1] are developed. On the other hand to crawl relevant pages from traditional web, techniques of link context extraction has been used by many researchers [3, 4, and 5]. The context of link provides the semantic information related to target web page on that link so that decisions can be made based on that semantic information whether the target page is to be crawled or not. In this paper we have proposed and implemented the distributed semantic web crawler. Our crawler is able to crawl semantic as well as traditional web pages. To understand the meaning of anchor text in this paper we have used the Stanford Parser Based Approach for Extraction of Link- Context from Non-Descriptive Anchor-Text in our architecture. We have also used Jena framework to understand the semantic web pages, this framework create a model for the fetched OWL/RDF.The model created by jena is stored for later used to make advanced, quality related filtering, ranking and analysis of the collected semantic content contained in the OWL/RDF page, which is matched with the domain specific stored ontology to make the crawling decisions for semantic web pages. Rest of the paper is organized as follows: Related work is discussed in section 2.Section 3 described the proposed scheme in detail. In section 4 implementation is described with Conclusion and future work in section 5.

II. RELATED WORK

A continuous effort has been done by researchers to make the crawl more relevant pages. A Rule-Based Approach for Extraction of Link Context from Anchor-Text Structure has been used by authors in [3] they have used a rule based approach to extract the context of link using LR parser. In [4] authors have used a finite state automata to recognize the named entities in a given text this system shows a great improvement for extracting named entities from the text of web page which can also be utilize for understanding the structure of anchor text. As major portion of the web pages on www contains the links on non-descriptive anchor text so authors in [5] have implemented an Stanford parser based approach for link context extraction of non-descriptive anchor text this approach provides a great help to crawlers who are making a blind search of links given on non-descriptive anchor text. Even this approach is appropriate for descriptive anchor text because it considers and analyse the text in the vicinity of anchor text to determine the context of the link. Various techniques of ontology based focussed crawlers were given by different authors [6,7,8,]. An ontology-focused crawler were proposed by [6] [7]. In their approach by using a domain specific ontology users can first define, crawling target to limit the crawling scope. Then focused crawler is used to retrieve the data from websites based on the ontology and crawling scope. Once the data is fetched relevancy is calculated between the ontological concepts and the crawled data by means of TF-IDF algorithm. In [8] the web pages are retrieved by linking them to topics by focused crawler. In this also terms of

978-1-5090-0076-0/15 $31.00 © 2015 IEEE
DOI 10.1109/CICN.2015.272

ontology is used to define each topic. Topical page rank algorithm [6] is used to compute relevance values between the terms and document texts for both global (the whole database) and local (the topic ontology) perspective. A pipeline architecture for crawling both traditional as well as semantic web pages is given by [9].The main goal of their pipelined architecture is to obtain large amounts of semi structured data from the web. Authors in [10] have proposed a methodology to extract Meta tags and metadata from web pages stored in knowledge base by using a web crawler.

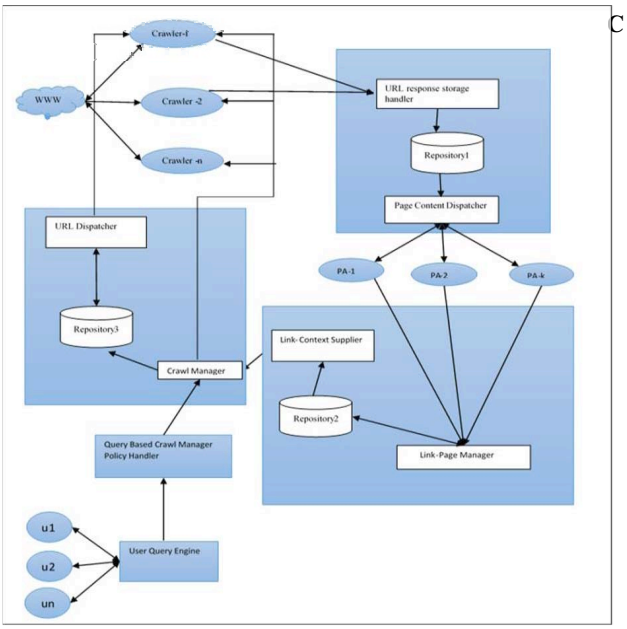


Fig: 1

We have divided our system into six major components each component is implemented on a separate machine. This complete system works as client server architecture. All the different components can communicate with each other in a client server mode. When one component wants to communicate with other component a socket is created using ip address and port address of the machine running that component. Our proposed architecture consists of following components as shown in Fig: 1.

1. Crawl Controller
2. Crawler
3. Page Manager
4. Page Analyzer(PA)
5. Analyzed page content supplier
6. Query Based Crawl Manager Policy Handler

A. Crawl Controller

This component is responsible for maintaining all the crawl policies. It controls all the crawlers running and maintain their states through message passing. Two main part of this component are:

1) Crawl Manager

The crawl manager is the main component for managing the policies for running the crawlers and page to be fetched. List of policies which are used by crawl manager are:

(i) Where and when to start and stop a crawler.

Normally the crawler will start from the initial see pages. However, depending on the user queries sometimes it start with specific domain and from somewhere more central to the site.

(ii) Concurrent Pages

It is the number of pages crawler will crawl concurrently this number is also decided by crawl manager.

(iii) Pause Between Pages

It is the waiting time a crawler must wait before moving to next page.

(iv) Save Log

Maintain the state of all the crawlers. If any crawler stops in between start from the same point.

(v) Context specific crawling

2) URL Dispatcher

This component is an interface between Repository 3 and various instances of crawlers. It provides a list of URLs as response to the request made by different crawlers.

B.Crawler

This is the second component of our distributed semantic web crawler. There are separate crawlers running on different machines they create a connection with crawl manager and page manager when they are started

1) World Wide Web

It is a global system of interconnected pages via hyperlinks based on Internet. It is a source of information for search engines. Search Engines download web pages from www which are then indexed.

2) Crawler

These components (each executing on separate machine) of our distributed search engine use input URLs to automatically download corresponding web pages from the Web. Individual instance of crawler starts with a set of seed URLs (for example, http: //www. yahoo. com). It makes a request to the server specified by a URL, and then downloads the corresponding page or receives an error. The downloaded pages or errors are then stored in a repository, a data structure shown in Repository 1 through an interface called URL Response Storage Handler (URSH) using client server architecture between crawler(as client) and URSH(as server). While designing the crawler instance, the scheduling policies are maintained by crawl manager. The crawler will crawl web pages as per the policy decided by Crawl Manager.

C. Page Manager

This component is responsible for responding the crawlers request to store the fetched page and then provide the stored content to page analyzer for further analysis. Two sub components of page manager are:

1) URL Response Storage Handler (URSH)

This sub component acts as interface between repository 1 (Table 1) and different instances of crawler. URSH receive request containing URL and corresponding page content form Crawlers and stores them in repository. The time of receiving page is also stored. If error are received then the status of errors are stored so that later the corresponding URL may be considered.

2) Page Content Dispatcher

This is the another sub component of page manager which is working as an interface between repository (Table 1) and instances of a component called Page Analyzer. It dispatch pages constituted at respective URLs to instances of Page analyzer.

D. Page Analyzer (Pa)

One instance of Page analyzer extracts links from the page content and their contexts using techniques based on statistical analysis and natural language processing. In our architecture of page analyzer for link context extraction we have used the technique of Stanford-Parser based approach for link context extraction of non-descriptive anchor text implemented by us in our previous research. The detail algorithm is given in next section. Page analyzer also extracts the theme of page. The separate instance of page Analyzer are executing on separate machine. This extracted information is stored in data structure shown in repository 2 through Link-Page Manager.

E. Analyzed Page Content Supplier

This component act as an interface between Page Analyzer and Crawl Manager. It receives the theme of the page and context of all the links in the page from Page Analyzer and provide the context to Crawl Manager through Link-Page Manager.

1) Link-page Manager

This component acts a interface between Page Analyzer and Repository 2 to store the extracted URLs from a page, their context, and the theme extracted from the page

2) Link-Context Supplier

It is an interface between repository 2 and Crawl Manager. This interface is passing URLs (Links) and context of URLs to Crawl Manager.

F.Query Based Crawl Manager Policy Handler

This component act as an interface between crawl manager and user queries so that crawl manager can decide the crawling policies based on user queries.

IV. PROPOSED SCHEME

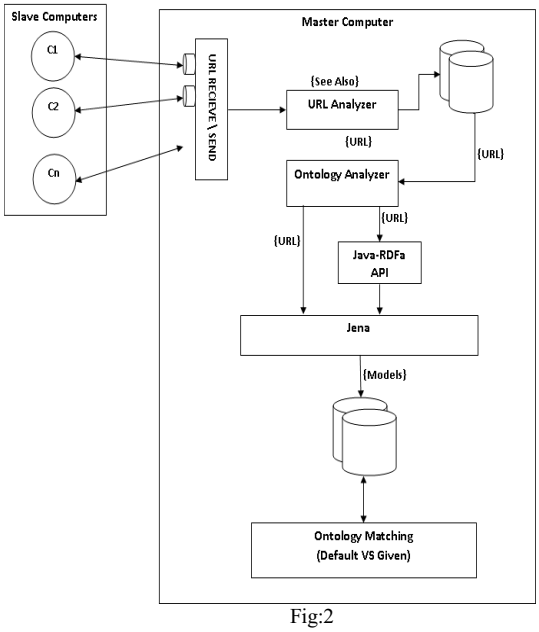


Fig:2

Crawler is the main component of any search engine because it is responsible for fetching the data from the World Wide Web and make it available for storing to indexer. If crawler is intelligent enough to make decisions whether the page it is going to crawl is relevant to its domain and can be used for indexing most of the problems of irrelevant search can be resolved and processing becomes faster. In our proposed scheme we try to implement a distributive semantic web crawler which makes its crawling decisions by understanding the concepts and context of links to be fetched so that it is able to avoid useless crawling of pages. In our scheme we have used an architecture of distributed web crawling, in which many independent machines are used to crawl, download, extract and index the semantic web via web crawling. We have developed a component called page analyser shown in fig 1.which is implemented on separate machine. It is responsible for analysing and extracting the links form the page sent by the crawler. It also verifies about the type of page if it is an OWL/RDF page the content is given to ontology analyser which using the Jena extracts the semantic information and creates a model to be stored in the database for further processing. If it is a html web page all the links will be extracted and stored in My Sql database to be used by link context extractor to extract the information from the all type of anchor text. The output of page analyser which is context and theme of all the links and text in the page is utilized by Link context manager. Link context manager works two folds it provide the links and context of links to Link context supplier which is an interface between the crawl manager and Link context manager. Second Link context manager provide the theme of page to indexer for storing the semantic information. Mainly all the policies of our

crawling framework is handled by crawl manager this component is responsible for managing all the crawler running on different machines. Our implemented architecture shows a great improvement for fetching the relevant pages. Also this due to distributed in nature it is robust kind of system which provides maximum reliability.

IV. IMPLEMENTATION

The system is implemented using Java as a front end and Oracle 11 I as backend. We also use Jena and Jsoup for model extraction of semantic web pages.Stanford parser based approach for link context extraction of Non-Descriptive anchor text is used for extraction of link context by page analyzer.

A. Crawl Controller

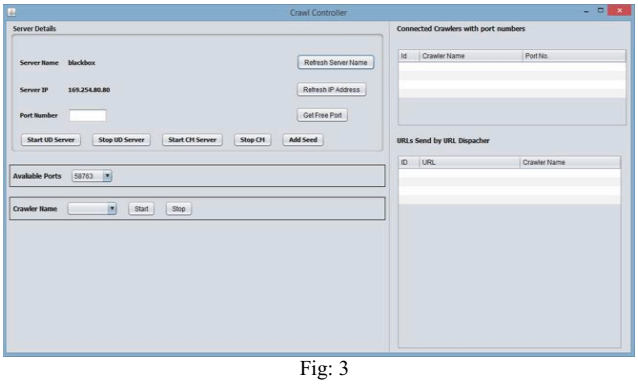


Fig: 3

1) Pseudo code for Crawl controller

```
private void jButton1ActionPerformed ()
String sql1 = "Select * from seeds where url = textbox_text
ResultSet rs = db.runSql(sql1);
If (seed already added)
    showMessage ("seed already added");
 else
    String sql = "INSERT INTO seeds (seed details )";
    PreparedStatement stmt = db.conn.prepareStatement(sql);
 If ( stmt! = empty)
    Execute statement;
 else
    Show message (" statement not executed")
    Refresh() // to fetch seed details from db
```

Fig: 4

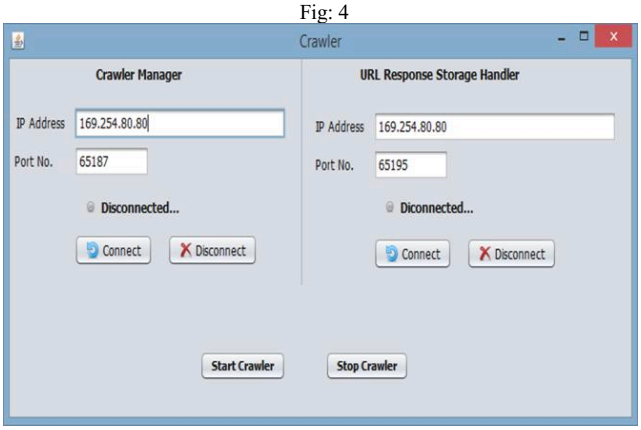


Fig: 5

B. Crawler

Crawler Client connects to the server started by crawl manager and fetches the resources upon which model extraction is done.

C. Page Analyzer

Fig: 7

The page analyzer acts as client for both page content dispatcher and link page manager. It extract hyperlinks from the RDF or HTML pages and extract models i.e. Subject Predicate and Object from the given page. To extract hyperlinks from the HTML PA use JSoup API in which it sees the HREF tag and extract the link. To extract hyperlink from the RDF and OWL PA uses JENA API.

1) Pseudo code for Model Extraction

```
extractModels(id, uri, html, langType)
{
    Model model =ModelFactory.createDefaultModel ().read (uri);
while (statements present)
        Statement s = nextStatement
        Resource r = s.getSubject
        Property p = s.getPredicate
        RDFNode o = s.getObject
        String res = r.getURI
        String prop = p.getURI
        String obj = null;
        if (object is URIResource)  then
          obj = o.asResource().getURI

        if (langtype != html) then
          if (subject does not contain # and is not null ) then save subject into database

              if (predicate does not contain # and is not null) then
            save predicate into database

          if (predicate does not contain # and is not null) then
            save object into database

  }
```

Fig: 8

2) Link Context Extraction Of The Anchor Text In The Page

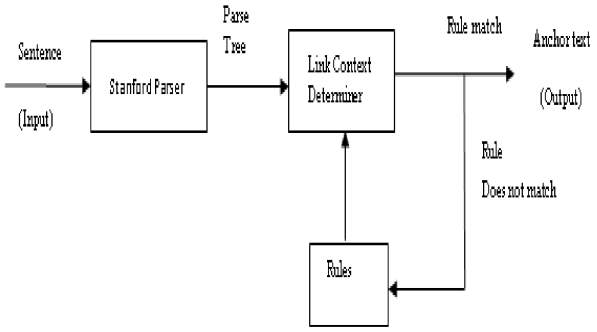
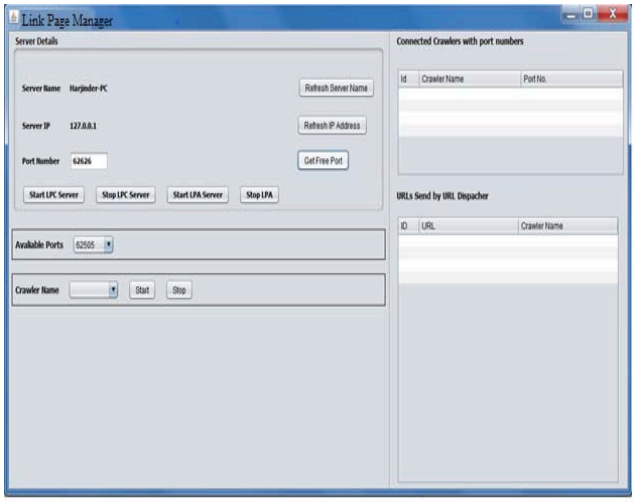


Fig: 10 [5]

For link context extraction we have used the technique given by us [5].All the links in the page and content of page is given to the SPLCA. The output of this SPLCA is given to the Link Page Manager.

D. Analyzed Page Content Supplier



The Link page manager stores the URL or filter the URL which are ready to be fetched. The work of this controller is only to get data from multiple PAs and filters the URL and store them into database PA acts as client and LPM acts as server.

V. CONCLUSION AND FUTURE WORK.

In our distributed semantic web crawler we have designed and implemented a scheme of distributed semantic web crawling which is able to crawl pages from both traditional as well as semantic web. We try to make our architecture more robust and reliable by using a distributed computing architecture. We have used a scheme of SPLCA which help our crawlers to fetch more relevant information. Jena framework is also used so that pages written in RDF/OWL are easily handled by our system which can be later used to implement the ontology based crawling.

In this paper we have implemented distributed semantic web crawling using the link context of the anchor text in future work we further enhance this architecture for ontology based crawling also.

REFERENCES

[1] Slug: A Semantic Web Crawler Leigh Dodds, leigh@ldodds.com, February 2006

[2] T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. Scientific American, 284(5):34–43, 2001.

[3]]. Suresh Kumar, Naresh Kumar, Manjeet Singh, Asok De “A Rule-Based Approach for Extraction of LinkContext from Anchor-Text Structure” Intelligent Informatics Volume 182 of the series Advances in Intelligent Systems and Computing pp 261-271.

[4]. Naresh Kumar, Manjeet Singh “Domain independent named entity recognition system based on finite state automata” ijacit Volume 2 Issue 4, 2013 - August Issue pp 77-87 .

[5] Naresh Kumar, Manjeet Singh “Stanford Parser Based Approach for Extraction of Link- Context from Non-Descriptive AnchorText” 978-1-4799-6896-1/14 ©2014 IEEE explore Reliability, Infocom Technologies and Optimization (ICRITO) (Trends and Future Directions), 2014.

[6] M. Ehrig and A. Maedche, "Ontology-focused crawling of web documents," in ACM Symposium on Applied Computing (SAC 2003), Melbourne, 2003.

[7] M. Ehrig, A. Maedche, S. Handschuh, L. Stojanovic, and R. Volz, "Ontology-focused crawling of web documents and RDF-based metadata," in Intenational Semantic Web Conference 2002 (ISWC 2002), Sardinia, 2002.

[8] A. Ardö, "Focused crawling in the ALVIS semantic search engine," in 2nd European Semantic Web Conference (ESWC 2005), Heraklion, 2005.

[9] Anthoniraj Amalanathan, Senthilnathan Muthukumaravel “Semantic Web Crawler Based on Lexical Database” IOSR Journal of Engineering Apr. 2012, Vol. 2(4) pp: 819-823

[10] Li Ding, Rong Pan, Tim Finin, Anupam Joshi, Yun Peng, and Pranam Kolari “Finding and Ranking Knowledge on the Semantic Web” Proceedings of the 4th International Semantic Web Conference, Galway IE, November 2005, Springer-Verlag.

[11] MultiCrawler: A Pipelined Architecture for Crawling and Indexing Semantic Web Data

Andreas Harth, Ju¨rgen Umbrich, and Stefan Decker

[12] G.S. Tomar, S. Verma & Ashish Jha; “Web Page Classification using Modified naïve Baysian Approach”, IEEE TENCON-2006, pp 1-4, 14-17 Nov 2006

[13] Gautam Pant “Deriving Link-context from HTML Tag Tree” DMKD03: 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery, 2003.