International Journal of Computer Theory and Engineering, Vol. 1, No. 1, April 2009
1793-8201

Compounded Uniqueness Level: Geo-Location Indexing Using Address Parser

M. Shoaib Jameel and Tejbanta Singh Chingtham

***Abstract*—Geo-location searching is an important feature for any search engine and research in this field is not new. The only issue that remains is how a search engine know whether a web page belongs to India or the USA? URLs ending with *[.in]* are the ultimate choice for India but not all web sites from India end with *[.in]*. This paper describes a technology known as the address parser. The address parser searches for patterns in a web page that communicates address information. The address parser does not parse every web page of a website for extracting the address but only works on those URLs where the probability of finding an address of the website owner is maximum, thereby eliminating false positives. A central knowledge base is built manually, which contains information like States of a country followed by their city names and other relevant information that may help the address parser do precise local indexing. It was observed that the address parser was not only able to recognize the address patterns in the web pages but also indexed them to city specific information. As a result, a person located in *Gangtok, Sikkim, India* searched for *[universities]*; the searching module showed the link of *[Sikkim Manipal University]* first, followed by other links from India. This work also focuses on the importance of the terms contained in the URLs for geographical based indexing and searching.**

***Index Terms*— Address Parser, Geo-location Indexing, Information Retrieval, Localized Searching**

I. Introduction

Geographical based searching plays a vital role in web search. A user in a particular region uses search engine to search for *[restaurants]* or *[book shops.]*, will be more interested to know which *restaurants* or *book shops* are close to his locality. Geo-location [58] searching not only helps a person get acquainted with the entities present very close to the person's locality but it also helps in giving meaningful and relevant information to the user. Imagine a search engine giving results of the *hotels* in the USA to a user sitting in India? Not all web pages convey their geographic information. For example, a web site *[http://www.amity.edu]* belongs to a university in India, but how can a search system know about its geographic information? Similarly, *[http://www.tatasteel.com]* [57], which is an Indian steel company, does not convey any information that it belongs to India.

This paper deals with a very robust implementation of understanding a web page's geo-location information. The implementation then indexes the web page to the city specific indexing. Current search technologies do not show results that are very close to the user's location. By entering *[hotels]* from *Jamshedpur, India*, search engines give results of the hotels in India but none of them reflect *hotels* of city from where a person has issued the query. This is what this paper addresses. What search engines have prospered to do is that most of them give results that are in the user's own country. This paper focuses on the design and working of an "Address Parser". The address parser tries to confirm patterns in the web page that convey postal address scheme. The parser carries out several confirmatory tests to conclude that the text in the web page is an address. Subsequently, the address parser indexes the website to city specific level from the location information provided in the address.

Many IP-address locators [29], [30], [31] and [32] provide the geo-location of the URLs. What they provide is the address of the domain owner under whose name the domain is registered. If this utility is used in determining the geo-location of the web pages, even then the need for an address parser arises to index the domain to city specific indexing. Moreover, in some cases address retrieved from the IP-address locators may be old and have changed. These locators can be useful in situations when there is no geographic information present in the web page or their URLs. The main focus in this paper is to detect addresses present in a web page and index the page to the city specific level.

The algorithm uses a knowledge base that is manually maintained. This knowledge base is huge, which contains information of the colleges, universities, industries, cities, states, famous landmarks etc. of a particular country. Using information from the knowledge base and web pages' META and TITLE tags, the algorithm indexes a site based on its geographical location. Our research work focuses only on India[3].

After the description of the indexing scheme, the paper describes the searching and presentation of search results. Through console based application (written in C) it has been shown that the algorithm provides search results that are very close to the person's location.

Manuscript received November 16, 2008.

M. Shoaib Jameel was with the Department of Computer Science and Engineering, Sikkim Manipal Institute of Technology, Majitar, Rangpo, East Sikkim - 737132 INDIA. He is now with the Department of Research and Development/Scientific Services, Tata Steel Limited, India (corresponding author to provide phone: +919234502858;

Tejbanta Singh Chingtham is with the Department of Computer Science and Engineering, Sikkim Manipal Institute of Technology, Majitar, Rangpo, East Sikkim - 737132 INDIA.

[3] Due to familiarity of the postal address schemes prevalent in India, this research paper lays more stress on the addresses from India.

DOI: 10.7763/IJCTE.2009.V1.5

The work addresses the following questions:

- Can city specific searches be given to the user?
- Where can exact location information be extracted from a web page? How to design an algorithm that does not consider the addresses which are false positives?
- Is it possible to maintain such a huge central database? Data in the central database might become outdated and hence need arises to update the database. Can automated approaches be designed to automate the process of update of the central database?
- Many web pages contain more than one contact addresses. How to parse and index those web pages?

II. RELATED WORKS

This research is a part of The Thinking Algorithm [1], which is a web search engine algorithm that adds the following question with every query: *"Sitting in a particular region, WHY has the person entered such a query?"* Compounded Uniqueness Level is a part of the algorithm that aids in geo-location searching and solves the first part of the above question. Study related to geo-location searching is not new. Many research works have been undertaken previously that focus on unearthing the geo-location information of web pages. Sanderson and Kohler [15] verified that about 18% of the queries submitted to Excite [16] contain geo-graphical related terms. Moreover, at least 20% of the web pages included one or more easily recognizable location identifiers like postal addresses. These pages are locally very relevant [17] [18] [19] and [20]. Our work closely relates to [2] and [7], which sample web pages to find area codes and other location specific information that may aid in geo-location searching. However, the address parser of the Thinking Algorithm goes beyond such sampling and enhances recognition techniques to find more accurate information from the web pages. The address parser only considers certain classes of web pages where the probability of finding an address in web pages is maximal. Moreover, this algorithm also applies several heuristics in order to conclude that the text in the web page is an address or just normal text with city or PIN[4] code (false positive). The position where an address is present in the web page is also considered as one of the parameters by the address parser. Other works that deal with geo-location searching are [21], [22], [23], [24], [25], [26], [38], [39] and [40]. Larson [27] defines Geographic Information Systems (GIS) and Information Retrieval (IR). GIR is concerned with indexing, searching, retrieving and browsing geo-referenced information sources and the design of systems that carry of these tasks effectively. Ding et al [4] also studied the technique of extracting the geo-graphical information from the web pages. The work by Ding et al proposed two methods one related to the technique based on the distribution of HTML links to the web page and the second based on the technique of geographical references to the web page. Commercial search engines like Google [5], Yahoo [6] and Live [11] have given a lot of focus in giving localized search results to the user. None of the search engines give results, which are very close to the person's locality. What they give are the URLs from the country where the person is currently situated. Other prototypes of geographic search engines include Northern Light [8] and Overture [9].

The central knowledge base described here has some similarity to the Gazetteer in [28] and [37]. However, in order to make indexing more precise, we have made the central database more comprehensive with information relating to the universities, industries etc. Moreover, our choices of source information like PIN, cities, states name was such that those online sources are human maintained and are updated regularly. So, regular crawling keeps the central database updated.

III. COLLECTING AND PROCESSING URLS

Terms contained in the URLs convey important information, which can be tapped for geographic IR. A URL of the *[.in]* domain can be automatically indexed in the indexes of the Indian region. Similarly, the URL like *www.educationinfoindia.com/* [13] can be automatically indexed in the indexes of the Indian region owing to the fact that the URL contains the word *India.*

We collected a comprehensive list of cities, landmarks etc. in the central database. 1000 URLs were collected and stored in a text file. The URL parsing module of the address parser (written in C programming language) began to scan the URLs one by one to find terms like *[.in]* or *[India]*, city names and other entities from the central database. This helped in indexing the URLs to city specific level. URLs satisfying the condition were copied to another text file.

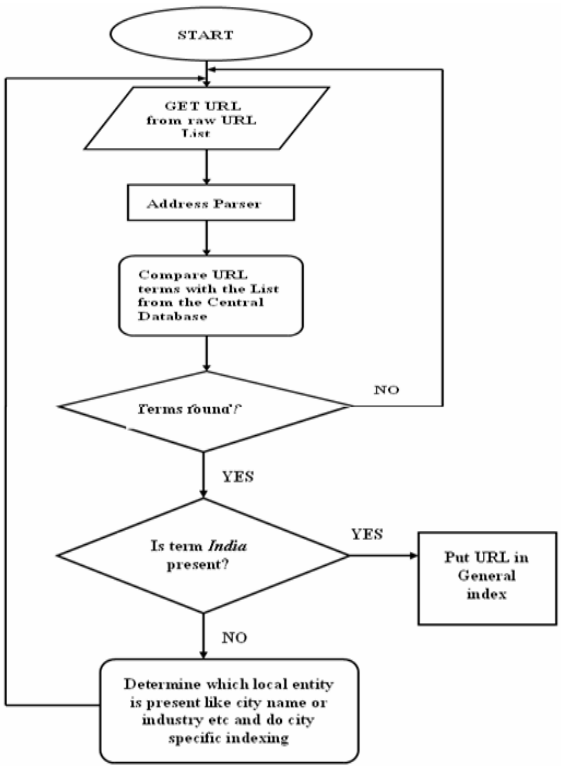


Fig. 1 Flowchart describing the URL parsing process for identification of geo-graphical terms.

After an in-depth study, we observed that most of the websites contained the address of the website administrators, which can help in geo-location indexing. The notion is that, if an administrator has a website, the administrator gives his/her contact address on the site (in one of the web pages).

[4] PIN stands for Postal Index Number. It is similar to the notion of ZIP codes in the United States and elsewhere.

Our target was to look for those web pages where the probability of finding an address in the web page was maximum and also with least false positives. It was observed that most of the sites had the URLs like *www.amwayindia.com/htmls/**contactus_locations.html**, http://www.abc-site/**contact_us.html**, http://www.abc-site/**address.html** or http://www.nitt.edu/home/students/facilitiesnservices/ComputerSupportGroup/**contact%20address/**, where the probability of finding an address is maximum because these web pages contained mostly the contact addresses of the owner of the website. These pages were downloaded manually and stored in a single directory for geo-location identification by the address parser. Web pages of the Indian universities, industries and famous landmarks were also collected and stored in similar directory as rest of the pages. These web pages did not convey any information about their geographical location from the terms contained in the URL because the URLs of these sites ended with either *.edu* or *.com*. Moreover, some of these sites did not have any contact information on any of its web pages which the address parser could parse and identify the location.

Some web pages, which neither had any *contact.html* link on their website but had an address in the bottom or top of the web page, were also downloaded. The main reason was to check how efficiently the address recognition program worked on such web pages?

IV. BUILDING THE KNOWLEDGE BASE

This was the most tedious and time-consuming task. A lot of time was spent on browsing the encyclopedias and Indian government sites in order to build this huge knowledge base. The knowledge base consisted of the following information:

- States of India along with their cities and Postal Index Numbers. This list is huge covering virtually every city of India.
- Famous Landmarks of India.
- Industries of India.
- Colleges and Universities of India along with their location information like city or state where they are located.
- A standard dictionary as a text file. This dictionary played an important role in finding some of the Proper Nouns in the web page.

We collected this information from various sources like [33], [34], [35] and [36]. The idea was that using such information, the parsing program can continuously consult the knowledge base for the occurrence of the city names or state names in the web page. It might be argued that building such a huge knowledge base was of no importance as just parsing the address and knowing the pattern could help to decipher the address in the web page. Moreover, consulting and finding the information from such a huge knowledge base involved lot of CPU processing power.

The reason for building such a huge database was that we wanted the address parser program to be very precise while searching for an address in the web page. We also wanted our indexing scheme to be very robust and perfect. Moreover, the knowledge base acted as an instructor to the parsing module, which gave the module most of the information that the address parsing module required. If the address parser found the name of the city and PIN in the web page, the parser consulted the knowledge base whether city name and the PIN for that corresponding place matched. This made the parser module more accurate in identifying an address in the web page.

Consider the URL, *http://www.jaipur.org.uk/* [12], though the indexing module would index the URL in the indexes of the UK country domain. However, the URL contained the word *Jaipur*, which is a city in India. Without the knowledge base, it was obvious that the parser module would not know that *Jaipur* is a city in India and hence the URL cannot be indexed in the Indian indexes. However, with the help of the knowledge base the indexer was able to index this site in the indexes of India and more preciously in the index of *Jaipur* in *Rajasthan*. There were so many other such information that we were able to tap from the META and TITLE tags of the web pages like indexing *http://www.amity.edu* [14] in the indexes of India because the knowledge base had the information that *amity.edu* is a university in India. This was the reason why building such a huge knowledge base proved fruitful. However, due to the spamming of META and TITLE tags [52], [53], [54] and [55], they cannot be relied upon. We have not addressed the spamming problem in this paper.

V. THE DESIGN OF THE ADDRESS PARSER

Web pages were manually downloaded and stored in one directory. We then parsed the HTML pages and extracted the textual contents. TITLE and META tags were also extracted. Programming of the address parser was done in C programming language owing to the efficiency reasons.

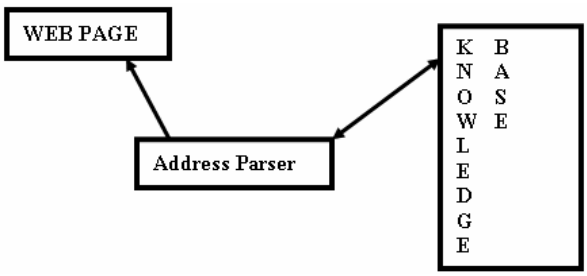


Fig. 2 An illustration of the working of an address parser

The address parsing program is the pattern-searching program that looked for patterns in a web page. Fig. 2 depicts the working of an address parser. It continuously consulted the knowledge base during the course of its execution. Addresses in India normally have a common pattern like *Number, Road Number/Name, City-PIN, State or Number, Name, City-PIN State or Number, Name, City, State PIN*. This is what the parsing program looked for in the web page. Consider the following example of an address

IndiaMARTIN InterExample

B-21, Sector - 14, Lucknow

Uttar Pradesh, INDIA. PIN - 201301

Phone: +(91)-(220)-4069600, 3961508

Fax: + (91)-(120)-2444743

The address parser passed through several phases in web page processing and address pattern detection. In the first phase, the parser processed words from the web page one by one and compared the word with the terms in the knowledge base. If none of the words in the web page and knowledge base match, then it was certain that the web page contained no information that may be processed because neither the words in the web page have any information that may lead to geo-location indexing nor it was feasible to proceed further with any other further processing. Such web pages were discarded and were not indexed geographically.

If the words in the web page matched with any of the words in the knowledge base, then the address parser did the following:

Step 1: Determine which word from the knowledge base had matched with that from the web page. For example, *city* name would match with the words of city list from the knowledge base. Consider an example, a web page has a word *national*, this word matched with the word in the colleges and universities list. However, the words following *national* could be anything like *flag* or *anthem*, the parser module extracted the next few words from the knowledge base and the web page and compared them to find a match. Like *National Institute of Technology,* the parser took one word at a time and compared the words sequentially from that of the knowledge base. If the words from the knowledge base and the web page matched in sequence, then further processing was done else not.

Step 2: If the word was from the States list and a corresponding *PIN* and *city* had been found near the *State*, then chances were immense that an address might be present. Consider an example, *Jamshedpur, Jharkhand – 831001*. The parser with aid from the knowledge base acquainted itself that *Jamshedpur* is a city in the state of *Jharkhand*, moreover it also came to know from the knowledge base that *831001* is the *PIN code* of *Jamshedpur*. This test was among the confirmatory tests in address pattern detection.

Step 3: When the State name had been found in the web page and compared with that of the knowledge base, then the file pointer was moved, currently set in the web page, forward to search for the city name in that state and compared the extracted names to the city names in the central knowledge base. Also, it has to be mentioned that the comparison for the city name will only take place in the list of cities within that State of India, which has been found in the web page. Also, check for the match in the PIN number in the web page. The PIN number should correspond with the PIN of the city or the place in the knowledge base.

Step 4: In case there was no match, move the file pointer back to few words and again begin searching for information as in Step 3.

Step 5: If there was a match between the *city* and its *PIN* number, then there was a high probability that the pattern of text might be an address.

Step 6: In that, area of the text where PIN, city and State had been matched the file pointer was moved forward and backward and search for some Proper Nouns. This process was accomplished by comparing the words from the web page with a dictionary. If the word in the web page did not match with any word in the dictionary, then there were some Proper Nouns in the text then we were close to confirming that the text in the web page was an address. Though during the testing phase we did not come across any address where Proper Noun was not present, we were able to extract at least one Proper Noun from the web page, in the vicinity of the PIN, state and city. This test was not among the confirmatory tests for address detection. This was just to enhance efficiency of the address parser and increase its detection precision.

Step 7: Search for some numbers, words like *"road"*, *"street"*, *"colony"*, *"nagar"*, *"marg"* etc. for more confirmation. In addition, search for some alphabets like "A", "Z", etc and for some special characters in the vicinity of the text like '.' or ', ' and '-'.

The reason for the above step was that some addresses like the one in the example cited before contained *B-21* i.e. an alphabet and a number. Consider another example, which throws more light on Step 6 and Step 7:

Jamia Millia Islamia (A Central University)

Maulana Mohammad Ali Johar Marg

Jamia Nagar, New Delhi - 1100 25, India

The address contains some Proper Nouns and some words as in Step 7.

Step 8: After immense survey, we also concluded that it would be better to check for first alphabet of every word in the address text. Normally, people write an address by beginning every word with a capital letter. We noticed this pattern in about 99.99% of the addresses that we came across.

Step 9: Look for patterns like *Phone* or *Fax* or *Ph* near the text where the presence of address was detected. If there was a match, it was confirmed that the web page had an address and since the web page belonged to a website that had *"contact us"* term in the URL, we were sure that the web page has a contact address of the administrator.

Step 10: The indexer module indexed the web page to the city based indexing.

Step 11: If the web page was from the *contact.html*[5] page, then no prediction of the position of the address pattern in the web page was required, otherwise, if the web page had an address, compute the location of the address pattern in the web page. Either the address has to be at the bottom or at the top.

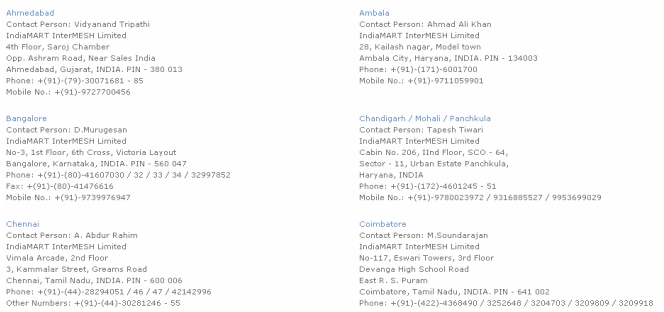
People normally write an address at the bottom or at the top of the web page. Sometimes, address is found at the left-centre of the web page. It was rare to see that an address was found at the centre of the web page followed by text above and below it. These web pages were not considered as good candidates for address parsing.

Step 12: If no address in the web page was found, words from the META, TITLE and URL of the web page were

[5] Here *contact.html* applies to all the forms of the web page where the probability of finding an address is maximal like *address.html*, *contact_us.html*, etc.

checked and compared with the words from the knowledge base.

Step 13: Presence of multiple addresses in the web page.


Ahmedabad
Contact Person: Vidyanand Tripathi
IndiaMART InterMESH Limited
4th Floor, Saroj Chamber
Opp. Ashram Road, Near Sales India
Ahmedabad, Gujarat, INDIA. PIN - 380 013
Phone: +(91)-(79)-30071681 - 85
Mobile No.: +(91)-9727700456

Ambala
Contact Person: Ahmad Ali Khan
IndiaMART InterMESH Limited
28, Kailash nagar, Model town
Ambala City, Haryana, INDIA. PIN - 134003
Phone: +(91)-(171)-6001700
Mobile No.: +(91)-9711059901

Bangalore
Contact Person: D.Murugesan
IndiaMART InterMESH Limited
No-3, 1st Floor, 6th Cross, Victoria Layout
Bangalore, Karnataka, INDIA. PIN - 560 047
Phone: +(91)-(80)-41607030 / 32 / 33 / 34 / 32997852
Fax: +(91)-(80)-41476616
Mobile No.: +(91)-9739976947

Chandigarh / Mohali / Panchkula
Contact Person: Tapesh Tiwari
IndiaMART InterMESH Limited
Cabin No. 206, IInd Floor, SCO - 64,
Sector - 11, Urban Estate Panchkula,
Haryana, INDIA
Phone: +(91)-(172)-4601245 - 51
Mobile No.: +(91)-9780023972 / 9316885527 / 9953699029

Chennai
Contact Person: A. Abdur Rahim
IndiaMART InterMESH Limited
Vimala Arcade, 2nd Floor
3, Kammalar Street, Greams Road
Chennai, Tamil Nadu, INDIA. PIN - 600 006
Phone: +(91)-(44)-28294051 / 46 / 47 / 42142996
Other Numbers: +(91)-(44)-30281246 - 55

Coimbatore
Contact Person: M.Soundarajan
IndiaMART InterMESH Limited
No-117, Eswari Towers, 3rd Floor
Devanga High School Road
East R. S. Puram
Coimbatore, Tamil Nadu, INDIA. PIN - 641 002
Phone: +(91)-(422)-4368490 / 3252648 / 3204703 / 3209809 / 3209918

Fig. 3 Web page containing multiple addresses

Explanation: During data collection phase, several web pages were downloaded which had multiple contact addresses like the one shown in Fig. 3. The question arose, where to index the web page in case of presence of multiple addresses in that web page?

We felt that indexing web page in all the cities listed in that web page would be an ideal case as the web page would be important to the people of every city listed in the contact address of that web page.

Let's describe how the solution was devised. Suppose there are ten addresses in the web page W: Address1, Address2, Address3 till Address10 for regions Region1, Region2, Region3 till Region10 for corresponding addresses. Let's assume that a person from Region1 searched for geographical based entities through the search system. Now, if that entity was present in the web page W, then the URL was displayed to the user as the first search result. Again, if another user from Region9 searched for the same entity which was present in the web page W, then the same URL was displayed to the person in Region9. This way the same web page with multiple addresses became important to all the regions whose address the web page contained.

Therefore, the address parser signaled the indexer to index the web page in all the cities listed in that web page. In case of multiple addresses in a web page, the address detection technique followed the same set of rules as those discussed above.

A. *OBSERVATIONS*

- Most of the web pages contained addresses that the address parser was successful to parse and subsequently index. Out of 1000 web pages, the address parser was able to correctly identify addresses in 989 of them. All the web pages in this 1000 set had postal addresses of India. This made the detection rate somewhat near to 99%. Some of the web pages (out of this 1000) were indexed geo-graphically based on the TITLE and META tags. They had addresses in the web pages, but owing to the META and TITLE detector algorithms, they were indexed without the address parser parsing them. The addresses that the parser failed to detect did not follow the semantics of our pattern detection. We included them because we wanted to test the efficiency of the address parser. Some of the addresses did not even contain PIN numbers but satisfied other semantics. We believe this is the area where our parser has to be improved for better address detection.
- We also downloaded 300 web pages that contained no postal addresses. Neither any information could be known from the META, TITILE tags or the URLs. The addresses parser did not detect any address in those web pages. Hence, they were not indexed.
- Most of the web pages contained contact details of its main corporate office and the addresses of the subsidiary companies on the same web page. In such a case, the parser parsed and confirmed all the addresses and the indexing algorithm indexed the same URL in all the city indexes, which were parsed and detected by the address parser.
- It has to be mentioned that those web pages were targeted, where the probability of finding an address is maximum, this is what gives us high success in address parsing.

VI. GEO-LOCATION DETECTION WITHOUT THE USE OF ADDRESS PARSERS

The algorithms designed not only worked on understanding the address patterns present in the web page but also rely a lot on the knowledge base for geo-location indexing. URLs like ***http://www.tata.com*** [56] or ***http://www.tatasteel.com*** [57] or ***http://www.amity.edu***, were directly geo-graphically indexed because they were present in the knowledge base, in one of the categories, like Tata Steel was present in the category of Indian industries. Some of the URLs had *contact us* URL, but with knowledge base, which indicated to the indexer, that for example ***http://www.amity.edu*** is a URL of a college in India. This was enough for the indexer to index the site in the indexes of India. This helped us to save processing power as the address parser was not involved in pattern searching. The knowledge base helped to index most of such websites, which conveyed no address or location information from their web pages.

VII. GEO-LOCATION INDEXING

After location has been confirmed using above methods, the indexer module of the address parser indexed the web pages based on their geo-graphical presence. It has to be mentioned that the indexer not only indexed a single web page, but the entire web site. The indexer indexed the web pages to the city-specific indexing. It made individual directories of each of the cities of India. Accordingly, web pages were put in the city specific directories.

Consider an example, suppose we have a web site *W*. Its *contact.html* web page is written as ***http://www.W.com/contact.html***. When the geo-location of the web page has been confirmed from the *contact* page, the indexer indexed the entire ***http://www.W.com***, to the city specific level. This was how the geo-location information of the entire website came to be known.

After the address parser parsed the address and found that the address belonged to *Jamshedpur, Jharkhand*; the indexer module created a directory *Jamshedpur*, which is a city in the state of *Jharkhand*. Therefore, the parent directory was *Jharkhand* while the city names formed the child directories. This made it much uncomplicated in city specific searches. With this indexing feature a person could find what all entities related to his/her search query are present nearby to the person's locality.

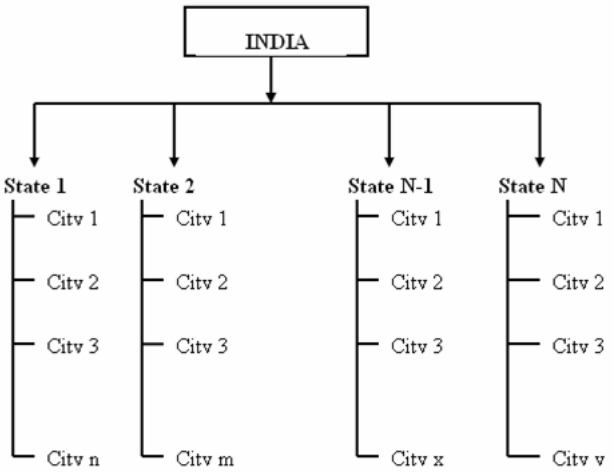


Fig. 4 Hierarchical view of the indexing mechanism. Here country is at the parent directory. State1, State2 etc are the child directories, within State1, State2 etc, city directories were placed.

Websites like ***http://www.indiaedu.com/***, which neither had any contact URL nor contained any information about its exact location; such were put in the *General* index. Such pages mostly contained those web pages that had the term *India* in their URLs. Web pages, which had the name of the city in their URL like ***http://www.jaipur.org.uk***, were put in city-based indexes.

VIII. EXPERIMENTS AND RESULTS

The main aim of this research was to study geo-location indexing of a search engine right to the city-level indexing. This would enable a searcher to locate some entities very close to his/her region. The research focused on extracting the address from certain classes of web pages where the probability of finding an address is maximal and correct.

Not all queries can be treated as geo-location based. For example, queries like *[data mining]*, *[online encyclopedias]*, etc. would be processed by the C.U.L algorithm of The Thinking Algorithm. They fall in the category of *global* queries. Therefore, classifying queries as *local* or *global* is important in such a case. Query classification is beyond the purview of this paper, so has not been dealt with. Some related works, which describe the detection of geography based on user queries, include [41], [42], [43], [44], [45] and [46]. It has been assumed in this paper that all the queries that are entered into our system are geo-location based queries.

Applications were developed in the C programming language. No Graphical User Interface (GUI) applications were developed. The main aim was to work on a "proof-of-concept".

No IP-to-location detection technique was applied in the console applications. The user passed an identifier stream of the form *query_stream#city#state*

We randomly downloaded 1000 (plus 300 extra, which had no address in them) web pages from different sources like company sites, university sites, web sites of Indian states, Indian government sites, tourism sites of India etc. Though the scale of 1000 web pages is very low in WebIR, however, our main purpose was to test the efficiency of the address parser and how preciously it detected address patterns in the web pages. Other major metric was to identify whether the CUL algorithm gave search results to city specific level. Some web pages contained contact information of the administrator or the company and some did not. Some web pages, which contained the expression *India* in their URL like http://www.indiaedu.com, were downloaded.

A. RESULTS

The console applications accepted input stream of the form *query_stream#city#state.* Consider a query *[hotels#Mumbai#Maharashtra];* the algorithm did not make use of the IP-to-location detection technique so, during program execution user specified the location from where the query was being entered. As output, the console application printed the URL of the most relevant web page.

Snapshots of the results (queries were entered by the authors) are pasted below, followed by a brief explanation of the results.

Fig. 5 Output 1: Query = purchase Microsoft windows; Location = Jamshedpur, Jharkhand

Fig. 6 Output 2: Query = colleges; Location = Delhi

Explanation: Fig. 5 and Fig. 6 are the snapshots of the applications that were written in C. We have developed different modules in The Thinking Algorithm. It has to be mentioned that these query streams were not passed through the following module:

- Ambiguity Removal Algorithm: Ambiguities do exist in user queries [47], [48], [49], [50] and [51]. In [3] Jansen and Spink showed that queries are shorter and therefore more ambiguous. However, work described here does not apply the Ambiguity Removal algorithm [1]. The main idea was to show the implementation of the C.U.L. algorithm i.e. location based searching and indexing.

In Fig 5, which is output 1, is an implementation that solves the inherent question of The Thinking Algorithm i.e. "Sitting in a particular region, why has the person entered

such a query?" From the query terms the algorithm computed that sitting in *Jamshedpur India*, a user wished to purchase the Microsoft Windows software. The algorithm then gave to the user that page that was very close to the user's location from where the person can go and buy the original copy of the Microsoft Windows software. The algorithm had given due credence to the terms contained in the URL (*Jamshedpur* is one of the terms in the URL).

It is now clear from the outputs of these console applications, that the results thus obtained are indeed very close to the person's current location. Consider the other query in Fig. 6, where the searching module displayed to the user the links, which were in the user's own city. Some other results that the searching algorithm produced are given below.

Example 1: Query = *furniture* and Location = *New Delhi*
Result: www.woodartindia.com/
www.delhifurniture.com/
Example 2: Query = *hotels* and Location = *Delhi*
Result: www.oberoihotels.com/
http://www.indiamart.com/hotelajanta/
Example 3: Query = *education in India* and Location = Mumbai, Maharashtra
Result: www.indiaedu.com/

In example 3, it has to be admitted that the searching module should give results very close to the user's region but in this case, the presence of *India* + *education* made the searching module give more priority to the URL name, as the URL *indiaedu* contained the terms that were in the query. The URL parser compared the terms of the query and the URL, character by character and this made it give more weight age to *indiaedu* for the words ***edu**cation in **India***. The other reason for this result was that *education* is a general term, which the algorithm cannot relate to the name of the college or the name of an industry or any place of interest in India. Hence, the algorithm gave more priority to the terms contained in the URL.

IX. Conclusion and Future Work

We've seen that web pages convey information about their location. What is needed are the ways of tapping those information and using them intelligently. What we have shown in this paper is a very robust and accurate method of extracting the geo-location information from the web pages. By crawling and processing certain classes of web pages from a website, the address parser was able to extract the location information from the web page. The drawback of the application is the huge knowledge base that it consulted during the course of location detection. It was also studied that even if a web site does not contain any contact information; using the knowledge base, geo-location indexing was possible. The only pain was finding those resources for the central database that is human maintained and is updated regularly in the internet. This implementation offered to the user the entities that are very close to the user's current location.

Scaling this implementation and testing with larger text corpora will be our main future aim. Moreover, we will develop separate address parsers for other countries too.

References

[1] S. Jameel, A. Akshat, and Singh Ch. Tejbanta, *Enhancements in Query Evaluation and Page Summarization of The Thinking Algorithm.* In the Proceedings of the Third International Symposium on Information Technology, Kuala Lumpur, Malaysia, vol. III, pp. 1979-1987
[2] O. Buyukkokten, J. Cho, H. Garcia-Molina, L. Gravano and N Shivakumar, *Exploiting Geo-graphical Location Information of Web Pages.* WebDB (Informal Proceedings), pp. 91-96, Jun 1999.
[3] B.J. Jansen, A. Spink, (2006), *How are we searching the World Wide Web? A comparison of nine search engine transaction logs.* IP&M, 42(1), 248-263.
[4] J. Ding, L. Gravano, and N. Shivakumar. *Computing geographical scopes of web resources*, In Proceedings of the Twenty-sixth International Conference on Very Large Databases (VLDB'00), 2000.
[5] Google Search Engine [Online]: Available: http://www.google.com
[6] Yahoo! Search Engine [Online]: Available: http://www.search.yahoo.com
[7] A. Markowetz, T. Brinkho, B. Seeger, *Geographic Information Retrieval*, 3rd International Workshop on Web Dynamics, 2004
[8] Divine inc., Northern Light GeoSearch, [Online]: Available: http://www.northernlight.com/geosearch.html.
[9] Overture Services, Inc., Local Search Demo [Online]: Available: http://localdemo.overture.com
[10] Microsoft's Live Search Engine [Online]: Available: http://www.live.com
[11] Information about Tourism in Jaipur [Online]: Available: http://www.jaipur.org.uk/index.html
[12] Education in India [Online]: Available: www.educationinfoindia.com/
[13] Amity University [Online]: Available http://www.amity.edu
[14] M. Sanderson, and J. Kohler, *Analyzing Geographic Queries.* In Proc. of the ACM SIGIR Workshop on Geographic Information Retrieval. Sheffield, UK, 2004, pp. 1-2.
[15] Excite Search Engine [Available]: Online http://www.excite.com/
[16] E. Amitay, N. Har'El, R. Sivan, and A. Soffer, *Web-a-Where: Geotagging Web Content.* In Proc. of the 27th Annual Int'l ACM SIGIR Conf. on Res. and Develop. in Information Retrieval. Sheffield, UK, 2004, pp. 273-280.
[17] K.A.V. Borges, "Use of an Ontology of Urban Places for Rec-ognition and Extraction of Geospatial Evidences on the Web (in Portuguese), " PhD Thesis, Federal University of Minas Gerais: Belo Horizonte (MG), Brazil, 2006.
[18] T.M. Delboni, K.A.V. Borges, A.H.F. Laender, and C.A. Davis Jr., *Semantic Expansion of Geographic Web Queries Based on Natural Language Positioning Expressions*. Trans-actions in GIS, 11(3): 377-397, 2007.
[19] H. Himmelstein, *Local Search: The Internet is the Yellow Pages*. IEEE Computer, 38(2): 26-35, 2005.
[20] K.S. McCurley, *Geospatial Mapping and Navigation on the Web*. In Proc. of the Tenth Int'l World Wide Web Conference (WWW10). Hong Kong: ACM, 2001, pp. 221-229.
[21] K.A.V. Borges, A.H.F. Laender, C.B. Medeiros, A.S. Silva, and C.A Davis Jr., *The Web as a Data Source for Spatial Databases.* In Proc. of the V Brazilian Symp. on GeoInformatics. Campos do Jordão (SP), Brazil, 2003, pp. CD-ROM.
[22] G. Fu, C.B. Jones, and A. Abdelmoty, *Building a Geo-graphical Ontology for Intelligent Spatial Search on the Web*. In Proc. of the IASTED Int'l Conf. on Databases and Applica-tions. Innsbruck, Austria, 2005, pp. 167-172.
[23] C.B. Jones, Purves, A. Ruas, M. Sanderson, M. Sester, M. V. Kreveld, and R. Weibel, *Spatial Information Re-trieval and Geographical Ontologies: an overview of the SPIRIT project.* In Proc. of the 25th Annual Int'l ACM SIGIR Conf. on Res. and Develop. on Information Retrieval. Tam-pere, Finland, 2002, pp. 387-388.
[24] M.J. Silva, B. Martins, M. Chaves, N. Cardoso, and A.P. Afonso, *Adding Geographic Scopes to Web Resources.* Computers, Environment and Urban Syst., 30: 378-399, 2006.
[25] C. Wang, X. Xie, L. Wang, Y. Lu, and W. Ma, *Detecting Geographic Locations from Web Resources.* In Proc. of the 2nd Int'l Workshop on Geographic Information Retrieval. Bremen, Germany, 2005, pp. 17-24.
[26] W. Zong, D. Wu, A. Sun, E. Lim, and D.H.G. Goh, *On As-signing Place Names to Geographic Related Web Pages*. In Proc. of the 5th ACM/IEEE-CS Joint Conf. on Digital Librar-ies. Denver, Colorado, USA, 2005, pp. 354-362.
[27] R.R. Larson, *Geographic Information Retrieval and Spatial Browsing*. In Geographic Information Systems and Libraries: Patrons, Maps, and Spatial Information, Smith, L.C. and Gluck, M., Eds. 1996, Un. of Illinois: Urbana, IL. p. 81-123.

[28] K.A.V. Borges, A.H.F. Laender, C.B. Medeiros, A.S. Silva, and C.A. Davis Jr., *Discovering Geographic Locations in Web Pages using Urban Addresses*. Workshop On Geographic Information Retrieval Proceedings of the 4th ACM workshop on Geographical information retrieval, Lisbon, Portugal Pages 31-36 Year of Publication: 2007 ISBN:978-1-59593-828-2
[29] IP Address Locator [Online]: Available: www.ipaddresslocator.org
[30] IP Address Locator [Online] : Available: http://www.analysespider.com/ip2country/lookup.html
[31] Webmaster Toolkit IP Address Locator [Online]: Available : http://www.webmaster-toolkit.com/ip-address-locator.shtml
[32] Ip2Location [Online]: Available: http://www.ip2location.com/free.asp
[33] Indian Colleges Directory [Online]: Available: http://www.123careers.net/
[34] Landmarks in India [Online]: Available: http://en.wikipedia.org/wiki/Category:Landmarks_in_India
[35] PIN codes and cities in India [online]: Available: http://www.indiavilas.com/indiainfo/pincodes.asp
[36] List of companies in India [Online]: Available: http://en.wikipedia.org/wiki/List_of_Indian_companies
[37] E. Amitay, N. Har'El, R. Sivan, and A. Soffer, *Web-a-Where: Geotagging Web Content*. In Proc. of the 27th Annual Int'l ACM SIGIR Conf. on Res. and Develop. in Information Retrieval. Sheffield, UK, 2004, pp. 273-280.
[38] F. Bilhaut, T. Charnois, P. Enjalbert, Y. Mathet, *Geographic reference analysis for geographic document querying.* Proceedings of the HLT-NAACL 2003 workshop on Analysis of geographic references, p.55-62, May 31, 2003
[39] K. S. McCurley, *Geospatial mapping and navigation of the web.* Proceedings of the 10th international conference on World Wide Web, p.221-229, May 01-05, 2001, Hong Kong, Hong Kong
[40] W. Zong, D.Wu, A. Sun , E. Lim, D. H. Goh, *On assigning place names to geography related web pages.* Proceedings of the 5th ACM/IEEE-CS joint conference on Digital libraries, June 07-11, 2005, Denver, CO, USA
[41] Z. Zhuang, C. Brunk, C. L. Giles, *Modeling and visualizing geo-sensitive queries based on user clicks.* Proceedings of the first international workshop on Location and the web, p.73-76, April 22-22, 2008, Beijing, China
[42] N. Cardoso, M. J. Silva, *Query expansion through geographical feature types.* Proceedings of the 4th ACM workshop on Geographical information retrieval, November 09-09, 2007, Lisbon, Portugal
[43] J. Kohler. Analysing *Search Engine Queries for the Use of Geographic Terms*. Master's thesis, University of Sheffield, 2003.
[44] B. Martins, M. J. Silva, S. Freitas, and A. P. Afonso. *Handling Locations in Search Engine Queries.* In Proceedings of the 3rd ACM Workshop on Geographical Information Retrieval, GIR'2006, Seattle, Washington, USA, 10th August 2006.
[45] L. Gravano, V. Hatzivassiloglou, R. Lichtenstein, *Categorizing web queries according to geographical locality*. Proceedings of the twelfth international conference on Information and knowledge management, November 03-08, 2003, New Orleans, LA, USA
[46] M. Sanderson, Y. Han, *Search words and geography*. Proceedings of the 4th ACM workshop on Geographical information retrieval, November 09-09, 2007, Lisbon, Portugal
[47] R. Song , Z. Luo , J.R. Wen , Y. Yu , H.W. Hon, *Identifying ambiguous queries in web search.*. Proceedings of the 16th international conference on World Wide Web, May 08-12, 2007, Banff, Alberta, Canada
[48] M. Sanderson, *Ambiguous queries: test collections need more sense.* In the Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval, Singapore, Pages 499-506, ISBN:978-1-60558-164-4
[49] R. Krovetz , W. B. Croft, *Lexical ambiguity and information retrieval.* ACM Transactions on Information Systems (TOIS), v.10 n.2, p.115-141, April 1992
[50] K. Spärck-Jones, S. E. Robertson, M. Sanderson, *Ambiguous requests: implications for retrieval tests, systems and theories.* ACM SIGIR Forum, v.41 n.2, p.8-17, December 2007
[51] C. Silverstein, M. Henzinger , H. Marais, and M. Moricz, *Analysis of a very Large Web Search Engine Query Log*. SIGIR Forum. 33(3), 1999. Originally published as DEC Systems Research Center Technical Note, 1998
[52] Z. Gyongyi, H. Garcia-Molina, *Spam: It's not just for inboxes anymore.* Computer, Volume 38, Issue 10, Oct. 2005 Page(s): 28 - 34
Digital Object Identifier 10.1109/MC.2005.352
[53] Z. Gyongyi, H. Garcia-Molina, and J. Pedersen. *Combating web spam with TrustRank.* Technical report, Stanford University, 2004.
[54] Open Directory Project. Open directory editorial guidelines: Spamming [Online]: Available http://dmoz.org/guidelines/spamming.html.
[55] A. Perkins. The classification of search engine spam [Online]: Available: http://www.ebrandmanagement.com/whitepapers/spam-classification/
[56] The Tata Group [Online]: Available www.tata.com/
[57] Tata Steel Limited [Online]: Available www.tatasteel.com/
[58] Geo-location [Online]: Available http://en.wikipedia.org/wiki/Geolocation