Profile Based Search Engine

Amruta Mantri[1], Priyanka Nawale[2], Trupti Pardeshi[3], Rajeshwary Shisode[4], Reena Pagare[5]
CSE, MAEER's MIT College of Engineering, University of Pune
Kothrud, Pune, India

***Abstract*- Huge amount of information available on internet makes it difficult for the user to get the exact search results according to his preferences. In this paper, we attempt to solve this problem to certain extent by extending the NUTCH open source search engine using personalized information of user. The user's information will be extracted from the social networking sites like Facebook. The search keywords given by user will be input to the NUTCH search engine. The results returned by NUTCH search engine will be further refined using our own Profile Biasing Algorithm.**

***Keywords*— Search engine, NUTCH, Crawling, Indexing, Profile Biasing, Profile Aggregator**

I. INTRODUCTION

Search is the first thing people use on the Web now, and there are fewer and fewer alternatives. There are many search engines available on the internet like Google, Yahoo and Bing. Google uses Pagerank algorithm [7] whose search is based on query keywords entered by the user mainly. Google takes a "trust us" approach to search [6]. Normally, search engine searches all sort of information even if not suitable for users' minors. The usefulness of a search engine depends on the relevance of the results it gives back. While there may be millions of Web pages that include a particular word or phrase, some pages may be more relevant, popular or authoritative than others. Most search engines employ methods to rank the results to provide the "best" results first [5]. How a search engine decides which pages are the best matches, and what order the results should be shown in, varies widely from one engine to another. The methods also change over time as Internet usage changes and new techniques evolve. To give more relevant and refined results according to user's interest we are building Profile Based Search Engine. We are extracting user's profile information from social networking site Facebook for finding relevant results.

II. RELATED WORK

A. Search Engine

A web search engine is designed to search information on World Wide Web. The World Wide Web (WWW) contains large amount data. Even in the day-to-day process we use search engine frequently. This is the reason for the increasing popularity and necessity of search engine. Search engine crawls the web and parses the WebPages and the URL's, which contains the user query keywords matched with the content of that specific webpage, are given as a results to the user.

Search engine helps to find information stored on a computer system such as the World Wide Web, or a personal computer.' Search engines use regularly updated indexes to operate quickly and efficiently. Basically through search engine we can access WebPages efficiently and easily.

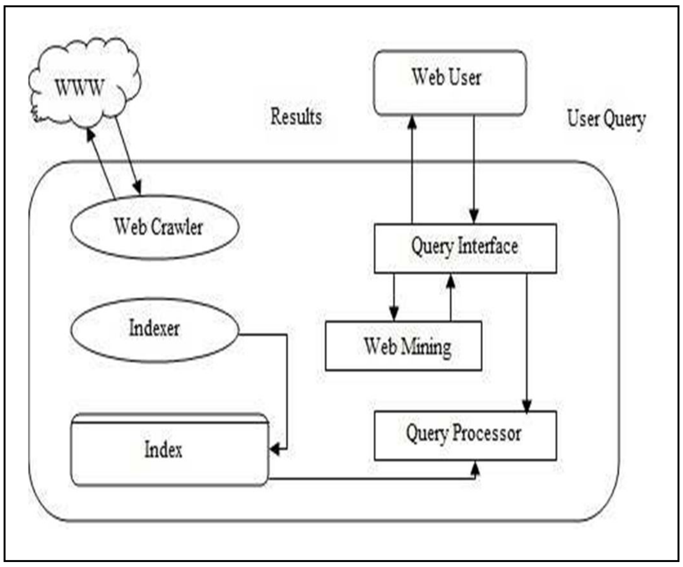


Fig 1: Architecture of Search Engine [10]

Crawler, Indexer and web mining these are the three important components of search engine. The crawler is also called as a robot or spider that traverses the web and downloads the web pages. The downloaded pages are sent to an indexing module that parses the web pages and builds the index based on the keywords in those pages. An alphabetical index is generally maintained using the keywords. When a user types a query using keywords on the interface of a search engine, the query processor component match the query keywords with the index and returns the URLs of the pages to the user. But before presenting the pages to the user, a ranking mechanism is done by the search engines to present the most relevant pages according to query keywords, at the top and less relevant ones at the bottom [9].

B. NUTCH

NUTCH is one of the open source search engine, so anyone can extend it. With Nutch, the indexing and page-ranking technologies are all open and visible [6]. It has different vital properties like flexibility, scalability, transparency [2], highly modular, understandability. Nutch is plug-in based. There are several other research projects built on NUTCH. Its implementation is done in Java. It provides all of the tools you need to run your own search engine [3].

a) Architecture of NUTCH

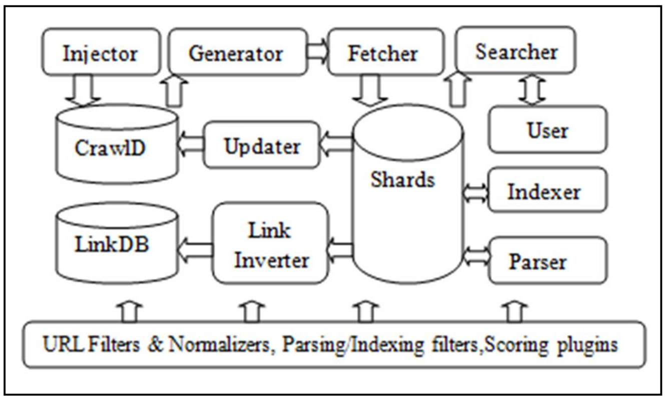


Fig 2: Architecture of Search Engine [10]

Steps in NUTCH Search Engine (Crawl and Index) [4]

i) Create a new CrawlDB (admin db -create).
ii) Inject seed URLs into the CrawlDB (inject).
iii) Generate a fetchlist from the CrawlDB in a new segment (generate).
iv) Fetch content from URLs in the fetchlist (fetch).
v) Update the CrawlDB with links from fetched pages (updatedb).
vi)Update segments with scores and links from the CrawlDB (updatesegs).
vii) Index the fetched pages (index).
viii) Eliminate duplicate content (and duplicate URLs) from the indexes (dedup).
ix) Merge the indexes into a single index for searching (merge).

1) Crawl DB

Maintains information on all known URL-s:

- Fetch schedule
- Fetch status
- Page signature
- Metadata

2) Link DB

For each target URL keeps info on incoming links, i.e. list of source URL-s and their associated anchor text

3) Segments

i) Collection of pages fetched and indexed by the crawler in a single run

ii) One segment dir for each crawl-fetch-update cycle at a particular depth.

iii) Contains raw text and parsed data of the files crawled

iv) The fetchlist for a segment is a list of URLs for the crawler to fetch, and is generated from the CrawlDB [1].

4) Parser is used to parse a new type of document.

5) Searcher uses inverted index to answer queries.

6) Indexer

The index is the inverted index of all of the pages the system has retrieved, and is created by merging all of the individual segment indexes. NUTCH uses Lucene for its indexing. The fetcher output is the data retrieved from the pages in the fetchlist. The fetcher output for the segment is indexed and the index is stored in the segment [1].

7) Link Inverter

The dependency graph is created among the pages by considering their outlinks. And then this links are inverted by the Link Inverter.

8) Scoring Plugins

Scoring is implemented as a filter plug-in, i.e. an implementation of the Scoring Filter class.

III.ARCHITECTURE OF PROFILE BASED SEARCH ENGINE

Profile Based Search Engine gives more refined and relevant results according to user's interest. Architecture of it consists of various components as explained below.

1) User Interface

User will interact with the system through the user interface provided. The NEW USER is required to REGISTER in order to use the system. During the registration the user need to provide his/her Facebook username. Then the Facebook API will be displayed to the user granting for the access to his/her Facebook profile. And once the user allows the access then all his/her

Facebook profile information is then stored into our local database. The OLD / REGISTRED USER can LOGIN into the system. User can now fire the query into the search input box provided. User when searches without LOGIN then he/she will get the search results provided by NUTCH.

2) Facebook Application

We have created one Facebook application which is used to extract his/her basic information and likes from Facebook profile. It asks user whether to allow Facebook application or not. This is done during registration process. Extracted information contains name, Facebook ID, location, likes, email ID.

3) Profile Aggregator

The user Facebook profile information stored into our local database needs to be up to date for the better relevancy of the results displayed to the user. Hence the Profile Aggregator is doing this work periodically.

4) Web or Internet data

This part of the architectural diagram represents the all available web pages over the internet. This whole web data will be further used by NUTCH.

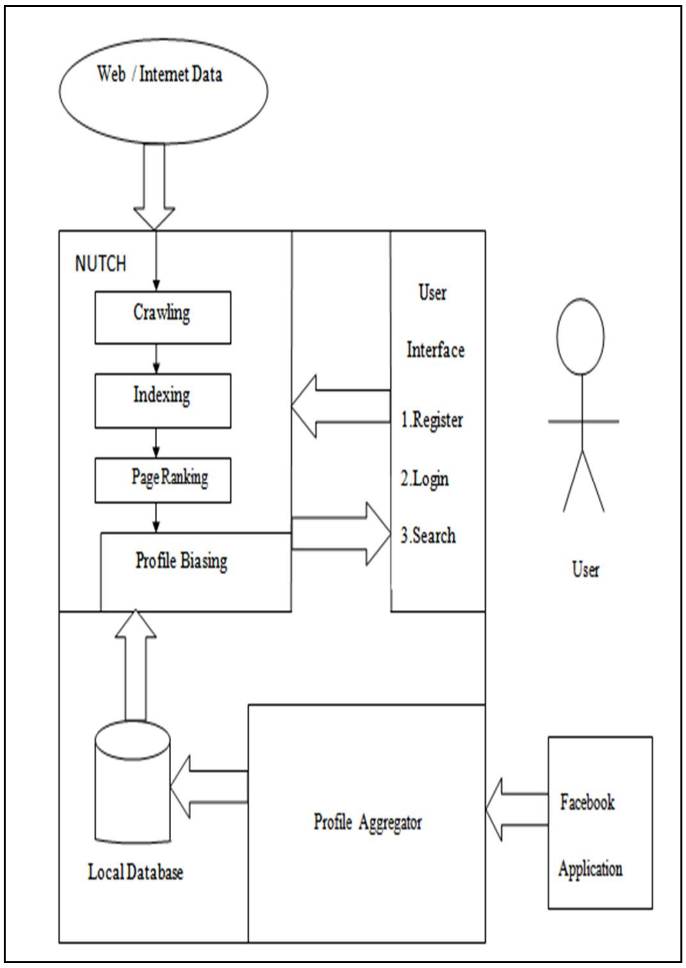


Fig 3.Architecture of Profile Based Search Engine

5) Crawling

Crawling is provided by the NUTCH. Crawling is done on the entire web by NUTCH. Crawling is a process or one type of software agent which starts with a list of URLs to visit, called the *seeds*. As the crawler visits these URLs, it identifies all the hyperlinks in the page and adds them to the list of URLs to visit. And again this procedure is repeated for those URLs and so on. Thus in this way it covers the entire web after crawling.

6) Indexing

NUTCH search engine maintains a copy of all the content it finds during the crawl process, and store it in an index for easy retrieval. Indexing is mainly carried depending on the keywords found in the web pages.

7) Page Ranking

The NUTCH applies the page ranking on the web pages crawled. The search result is provided by the NUTCH after the page ranking is performed by the NUTCH. The search result provided by the NUTCH will be further used for processing. But when the user doesn't want to LOGIN into the system then these results only are displayed to the user.

8) Profile Biasing

The Profile Biasing is the part of the architecture which is calculating the Profile Biasing Index (PBI) for web pages provided by the NUCTH. PBI calculation is based on the number of keywords matched in the user database i.e. the hit keywords (H_n). Then, the final rank will be computed for the web pages using the PBI. And accordingly the result will be displayed to the user on user interface.

9) Local Database

We will need local database for storing the user's Facebook profile information, user's registration details along with the all possible keywords describing the Genre per Facet.

A.STATE DIAGRAM

The overall flow of our system is represented by using state diagram as shown in Fig 4.

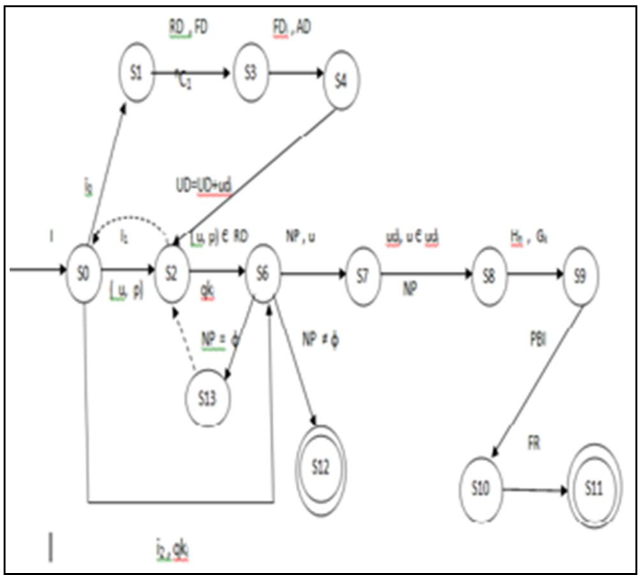


Fig 4. State Diagram of Profile Based Search engine

Let S be our system which is defined in the following manner as

S = {AD, UD, FD, CP, NP, FP, Qk, I, RD, Hn, Gk}

Where,

1) AD is the set of all admin databases.

2) UD is the set of all user database which is matched to the facebook profile and admin database.
3) FD is the set of facebook profile data.
4) CP is the set of all crawled pages by nutch.
5) NP is the set of pages given after page ranking by nutch.
6) FP is a final set of pages in which user profile keywords are matched.
7) I is a set of inputs defined as $I=\{i_0, i_1, i_2\}$
   i_0= New user registration
   i_1= Log in for already existing user
   i_2= User wishes for no recommendations
8) RD is a set of all users registration database.
9) Hn is a number of hits in genre keywords.
10) Gk is a total number of genre keywords.

At the initial state S_0, user is provided with three options viz. register, login and default search. During registration process user's facebook profile data is fetched into local database through facebook application as shown with transitions $S_0 \to S_1 \to S_3 \to S_4 \to S_2$. After login ($S_0 \to S_2$), NUTCH is providing the results on the user query keywords which is further refined by using Profile Biasing Algorithm as explained in section IV ($S_2 \to S_6 \to S_7 \to S_8 \to S_9 \to S_{10} \to S_{11}$). If the user chooses the third default search option then the results provided by the NUTCH itself are displayed to the user.($S_0 \to S_6 \to S_{12}$)

IV. PROFILE BIASING ALGORITHM

This algorithm is used for computing final rank for each page using Profile Biasing Index (PBI) and Page Rank (PR) as given by NUTCH.

Let's see working of PBI Algorithm:

Say, our Facet is Book and genres under this facet are Technical, Academic, and Fiction etc. Keyword set may contain "the", "best", "book", "technical" etc.

If user enters a query "Technical Books" Then result provided by NUTCH using Solr are like:

1. JAVA
2. C
3. C++
4. VB

Then we will take users' profile into an account for matching keywords. If users' profile contains keywords like "VB" and "C++". We will then calculate the no. of keywords matched in each webpage given by NUTCH. For each matched keyword within particular facet, we will increment no. of hit keywords i.e. Hn. Then we take into consideration of genre keywords of that facet as Gk. We calculate PBI by the formula:

$$PBI = PBI + ((Hn/Gk) * FMF)$$

Where FMF is Facet Multiplying Factor decided by us. Calculate final rank of web page by the formula:

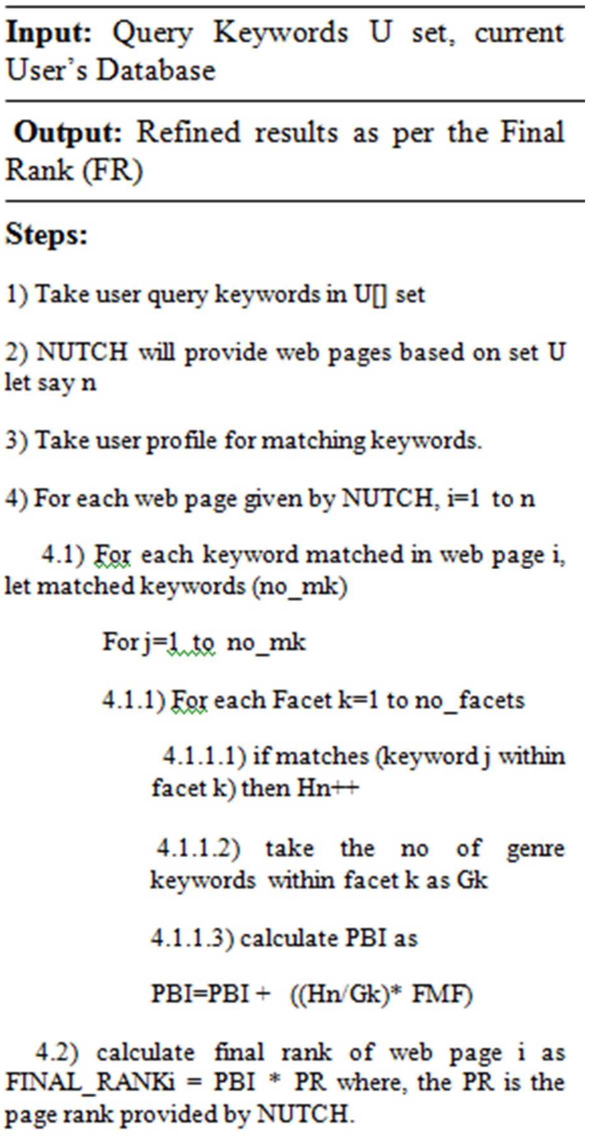
$$FINAL\_RANKi = PBI * PR$$

Where, the PR is the page rank provided by NUTCH.

Results of Profile Based Search Engines will be:

1. VB
2. C++
3. JAVA
4. C

Hence, User will get his results of interests on the first page and he/she will be happy at the end



Input: Query Keywords U set, current User's Database

Output: Refined results as per the Final Rank (FR)

Steps:

```
1) Take user query keywords in U[] set
2) NUTCH will provide web pages based on set U let say n
3) Take user profile for matching keywords.
4) For each web page given by NUTCH, i=1 to n
    4.1) For each keyword matched in web page i, let matched keywords (no_mk)
        For j=1 to no_mk
        4.1.1) For each Facet k=1 to no_facets
            4.1.1.1) if matches (keyword j within facet k) then Hn++
            4.1.1.2) take the no of genre keywords within facet k as Gk
            4.1.1.3) calculate PBI as
            PBI=PBI + ((Hn/Gk)* FMF)
    4.2) calculate final rank of web page i as FINAL_RANKi = PBI * PR where, the PR is the page rank provided by NUTCH.
```

V. CONCLUSION

There are many search engines available but they do not provide the results according to the user interest. The result may be few clicks away. Profile based search engine will retrieve results according to user's interest. This is achieved by refining the results returned by open source search engine, NUTCH. The PBI algorithm when applied to the results returned by NUTCH, it gives more relevant results.

VI. FUTURE WORK

In future we are proposing to implement PBI algorithm and compare the results returned by NUTCH with our Profile Biasing Search Engine. Instead of searching based on only user's profile we can extend our system to be more socialized by considering profiles of friends of user, family of user and friends of friends.

ACKNOWLEDGEMENT

This work is the result of inspiration, support, guidance and co-operation that is extended to us at their best by our teachers, especially our guides. We are grateful to Prof. Reena Pagare, for her valuable guidance throughout the preparation of this paper and most of all for believing in us. Sometimes a little belief is all you need to go the extra mile. We would like to thank Mr. Manoj Wartikar and Sonali Khandelwal, Persistent Systems Ltd. for being there for us whenever we needed help. We would like to thank them for his utmost co-operation and valuable guidance. Lastly we would like to thank all of them who have directly or indirectly guided us for this work.

REFERENCES

[1] Guojun Yu,Xiaoyao Xie, Zhijie Liu, "The Design and Realization of Open-Source Search Engine Based on NUTCH", International Conference on Anti-Counterfeiting Security and Identification in Communication , Page(s): 176 - 180 , 2010

[2] Rohit Khare, Doug Cutting, Kragen Sitaker, Adam Rifkin, "NUTCH: A Flexible and Scalable Open-Source Web Search Engine", 2005.

[3] Tom White, January 10, 2006, "Introduction to NUTCH", http://www.java.net/pub/a/today/2006/01/10/introduction-to-NUTCH-2.html

[4] Sebastian Nagel, November 1, 2012," NutchTutorial", http://wiki.apache.org/nutch/NutchTutorial

[5] Alessio Signorini, "A Survey of Ranking Algorithms", University of Iowa, pages 1-19, September 11, 2005.

[6] john-battelle , September 10, 2003, "An Open Source Search Engine", http://searchenginewatch.com/author/1765/john-battelle.

[7] Rebecca S. Wills, "Google's PageRank: The Math Behind the Search Engine", North Carolina State University, Raleigh, NC 27695, rmwills@ncsu.edu, May 1, 2006.

[8] Andrzej Białecki , "Nutch as a Web mining platform: the present and future", https://wiki.apache.org/ nutch/Presentations

[9] Ashutosh Kumar Singh, Ravi Kumar P, "A Comparative Study of Page Ranking Algorithms for Information Retrieval", International Journal of Electrical and Computer Engineering 4:7 2009