International Journal of Computer Science Engineering and Information Technology Research (IJCSEITR)
ISSN(P): 2249-6831; ISSN(E): 2249-7943
Vol. 6, Issue 1, Feb 2016, 69-78
© TJPRC Pvt. Ltd.

FOSSICK: AN IMPLEMENTATION OF FEDERATED SEARCH ENGINE

DEVJI CHHANGA[1] & XITIJ SHUKLA[2]

[1]*Department of Computer Science, KSKV Kachchh University, Gujarat, India*

[2]*Department of Agricultural Statistics, B. A. College of Agriculture, Anand Agricultural University, Gujarat, India*

ABSTRACT

Since the world wide web has become primary destination of information quest, a number of general purpose search engines exist to query the web and cater to the information needs of individuals. However, search results may not include results from sources which can potentially be very relevant to the user. An individual engages to various organizations in one or the other way. Majority of content from their web applications stored online is not indexed by the search engines. Queries do not yield results from user's personal email account or her company email or even her social media posts even though each of these applications have some built-in search tool. There is no fusion of information even as web applications like email, social media, e-commerce sites, government portals etc. can be searched in isolation. A federated search engine aggregates results from multiple information retrieval systems simultaneously and presents unified view of information. Standard output formats, for search engines, such as OpenSearch enable a federated search engine to combine results not only from general purpose search engines but from any web application conforming to such a format. This papers presents an on-line and non indexing implementation of federated search engine, FOSSICK, that aggregates OpenSearch based results from various user selected search engines.

KEYWORDS: Federated Search Engine, OpenSearch, Information Retrieval

Original Article

Received: Jan 12, 2016; **Accepted:** Jan 22, 2016; **Published:** Jan 25, 2016; **Paper Id.:** IJCSEITRFEB20168

1. INTRODUCTION

The field of Information retrieval has grown rapidly, since its introduction [1], to include wide range of applications from library science to web search. Even as 3.17 billion people worldwide uses Internet, 60% of world population is yet to get Internet access [2] which is an indicator of growth for web based applications, services and content. The needs and variety of information seekers is growing with the growth of web based applications.

A number of search engines exist to search from publicly available web content. However, as estimated by Steve Lawrence et al., no single search engine indexes more than 16% of the web [3]. Majority of the web's content called the “deep web” [4] remains hidden from us as only 4% of the web is indexed by general purpose search engines [5]. Thus, general purpose search engines alone may not always suffice to all types of information needs. From among these search engines a user may have to use multiple search engines for a query because results of one search engine differ considerably from the other [6]. Individuals engage to organizations in one or the other form viz. a customer, a consumer, an employee or a stock holder. Information access from sources across these organizations presents a novel problem to information retrieval research. For an enterprise that houses multiple application servers, integration of a search tool in organization' information management strategy results in improved decision making [7].

Expanding the web search beyond typical boundaries of surface web applications presents the challenge of integrating systems with myriad variety. An email server, for example, has its own search engine presenting results as a web page while a technical reports server in an organization may rely on a standalone search tool. However, introduction of standard output formats for search engines such as OpenSearch and federated search engines can overcome this hurdle. A federated search engine can aggregate results from multiple information retrieval systems simultaneously [8]. It creates a seamless search experience across different systems, domains and organizations. Combining top ranked results from each of the source search engines gives higher precision and still maintains the recall as search engine results are considerably different from one another [6].

We present an on-line and non indexing implementation of federated search engine, FOSSICK, which combines search results from multiple loosely-coupled search engines: search tools of web applications, general purpose web search engines or deep web search engines. Architecture section gives a bird's eye view of FOSSICK. Actual implementation details, including tools and libraries used, are discussed in implementation section followed by results section discussing performance analysis on CPU, memory and network as well as analysis of quality results.

2. ARCHITECHTURE

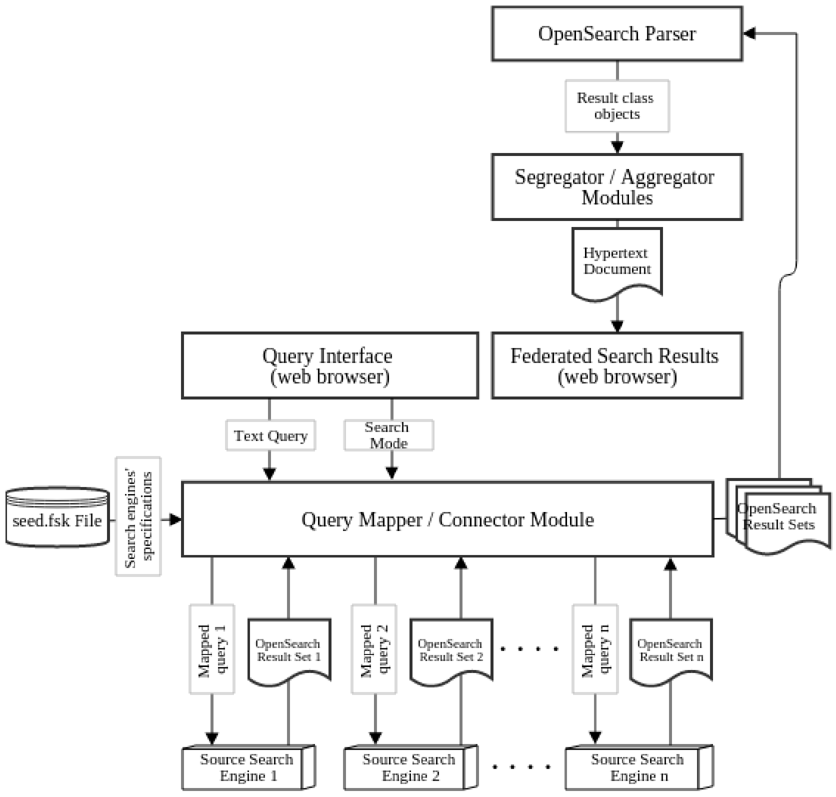


Figure 1: Architecture

Any search engine that supports OpenSearch format can be configured as a source search engine. OpenSearch is chosed because all major search engines support this format. Search engines to be used as source search engines are configured in an XML file: *seed.fsk*. This configuration can be changed at any instance which makes it infeasible to include indexing in FOSSICK.

For each run, user enters textual query in a web based user interface and selects one of the two running modes: (i) aggregate mode or (ii) segregate mode. In the first mode results are aggregated by their relevance. It is used when high precision is desired. The second mode segregates results by their source, ignoring relevance. This mode is used when high recall is desired.

Text queries are translated (mapped) before being sent to a source search engine by *Query Mapper / Connector* module. It generated mapped queries using configuration provided in *seed.fsk*. Response from each source search engine is received as an OpenSearch RSS document [9]. These documents are translated into processable custom *Result* class objects by *OpenSearch Parser* module. These objects are than aggregated or segregated as per the selected running mode and a hypertext document is generated for presentation to the user.

3. IMPLEMENTATION

FOSSICK is implemented as a web based application using: PHP version 5.5.9 on Apache web server version 2.4.7. Built-in SimpleXML library of PHP is used to store and process XML.

3.1 Configuring Search Engines in *seed.fsk* File

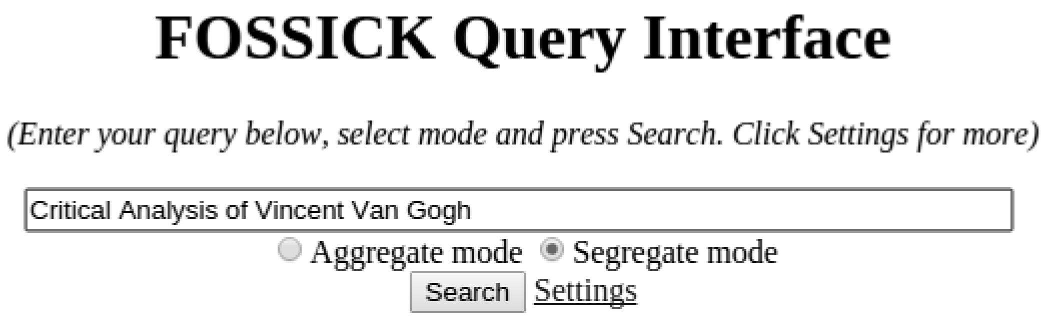
File *seed.fsk* is edited to add/remove or update source search engines. OpenSearch support is the only prerequisite for a source search engine to be compatible. For each source search engine following parameters are configured: (i) Search engine title, (ii) Connection URL, (iii) Parameters as name value pairs.

Following is an example *seed.fsk* file with two source search engines configured: (i) Bing, (ii) Archive.org:

```
<?xml version="1.0" encoding="UTF-8"?>
 <settings>
        <datasource>
                <title>Bing</title>
                <url>http://www.bing.com/search</url>
                <param>
                        <paramtitle>q</paramtitle>
                        <paramvalue>query</paramvalue>
                </param>
                <param>
                        <paramtitle>format</paramtitle>
                        <paramvalue>RSS</paramvalue>
                </param>
        </datasource>
        <datasource>
```

```
                <title>Internet Archive</title>

                <url>http://archive.org/services/collection-rss.php</url>

                <param>

                        <paramtitle>query=subject:</paramtitle>

                        <paramvalue>query</paramvalue>

                </param>

        </datasource>

 </settings>
```

3.2 Input Using Query Interface



FOSSICK Query Interface

(Enter your query below, select mode and press Search. Click Settings for more)

Critical Analysis of Vincent Van Gogh

Aggregate mode [✓ Segregate mode]

Figure 2: Query Interface: a Text Query "Critical Analysis of Vincent Van Gogh" is Input

3.3 Mapping Text Queries to Search Engine Acceptable Format

Query Mapper module maps queries using settings stored in *seed.fsk* file. Following code reads the configuration and generates mapped queries.

```
function qMap($seedAddress, $query) {

        $dataSources = simplexml_load_file('seed.fsk');

        $dataSourceArray = array ();

        $i = 0;

        foreach  ($dataSources as $datasource) {

                $url = $datasource->url;

                $getString = $url."?";

                $paramString = "";

                foreach  ($datasource->param as $param) {

                        $pTitle = $param->paramtitle;

                        $pValue = $param->paramvalue;
                        if($pValue == "query") {
                                $pValue = $query;
                        }
                        if ($paramString !== ""){
                                $paramString = $paramString."&";
                        }
                        $paramString = $paramString.$pTitle."=".$pValue;
                }
                $getString = $getString.$paramString;
                $paramString = "";
                $dataSourceArray[$i] = $getString;
                $i = $i + 1;
        }
        return $dataSourceArray;
}
```

3.4 Connecting to a Source Search Engine

Connector submodule connects to a source search engine and obtains result set in OpenSearch format.

```
        foreach  ($dataSources as $datasource) {
                $rss = simplexml_load_file($datasource);
        }
```

3.5 Parsing OpenSearch Results For Processing

Parser module parses XML objects containing OpenSearch results and converts them into an array of custom *Result* class objects. Shown below is the parsing code:

```
                $channel = $rss->channel[0];
                $source = $channel->title;
                foreach  ($channel->item as $item) {
                        $currentResult = new Result();
                        $currentResult->source = $source;
                        $currentResult->link = $item->link;
                        $currentResult->title = $item->title;
                        $currentResult->desc = $item->description;
                        array_push($results, $currentResult);
                }
```

3.6 Segregating the Results

Segregate is the default running mode which emphasizes on recall over precision. *Segregator* module groups custom *Result class* objects by their source and converts them into hypertext.

```
function segregate($resultSet) {
    $temp = "";
    foreach ($resultSet as $res) {
        if ($res->source != $temp) {
            echo "<hr/><b>Results from $res->source :</b><br/>";
        }
        $temp = $res->source;
        echo "<a href='$res->link'><b>$res->title</b></a><br/>$res->desc";
        echo "<br/><br/>";
    }
}
```

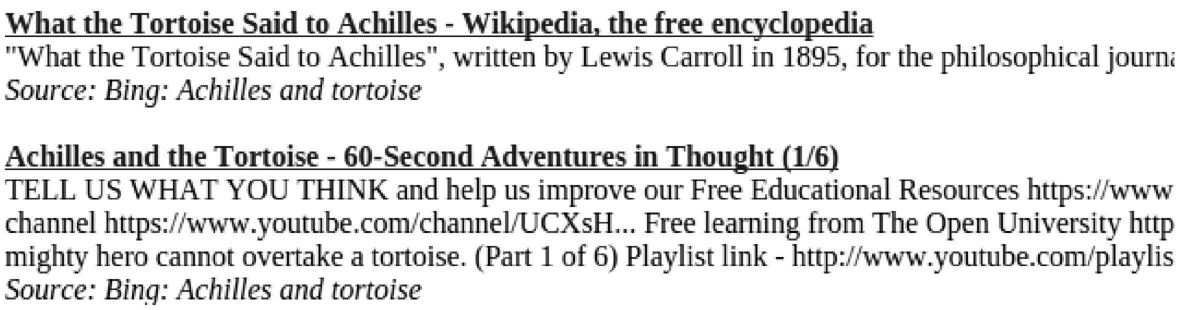
3.7 Aggregating the Results

Where precision is emphasized over recall, user selects aggregate running mode. The document should have at least one keyword from the user query in it's title. Results not matching this criteria are discarded. Remaining *Result class* objects are translated into hypertext.

```
function aggregate($resultSet, $query) {
    foreach ($resultSet as $res) {
        $resultTitle = $res->title;
        $keywords = preg_split("/[\s,]+/", $query);
        foreach($keywords as $keyword) {
            if(strpos($resultTitle, $keyword) !== false) {
                echo "<a href='$res->link'><b>$res->title</b></a><br/>$res->desc<br/>";
                echo "<i>Source: $res->source</i>";
                echo "<br/><br/>";
```

```
                        continue;
                    }
                }
            }
        }
```

3.8 Federated Results Presented to User



What the Tortoise Said to Achilles - Wikipedia, the free encyclopedia
"What the Tortoise Said to Achilles", written by Lewis Carroll in 1895, for the philosophical journa
Source: Bing: Achilles and tortoise

Achilles and the Tortoise - 60-Second Adventures in Thought (1/6)
TELL US WHAT YOU THINK and help us improve our Free Educational Resources https://www
channel https://www.youtube.com/channel/UCXsH... Free learning from The Open University http
mighty hero cannot overtake a tortoise. (Part 1 of 6) Playlist link - http://www.youtube.com/playlis
Source: Bing: Achilles and tortoise

Figure 3: Federated Results UI

Each result in federated results list contains a document title with hyper-link to that document and an informative summary of the document which helps the user to decide its relevance [10]. Underlying source search engine generates this summary.

4. RESULTS

Performance analysis is done on a workstation with Intel Core 2 Duo CPU P8400 @ 2.26GHz having 4 gigabytes of memory running Ubuntu 14.04 LTS 64-bit. Statistic data gathering is achieved using Collectl utilities [11]. Plotting Colplot library version 4.7.1 [12] is used for plotting graphs using gnuplot-x11 library version 4.6.

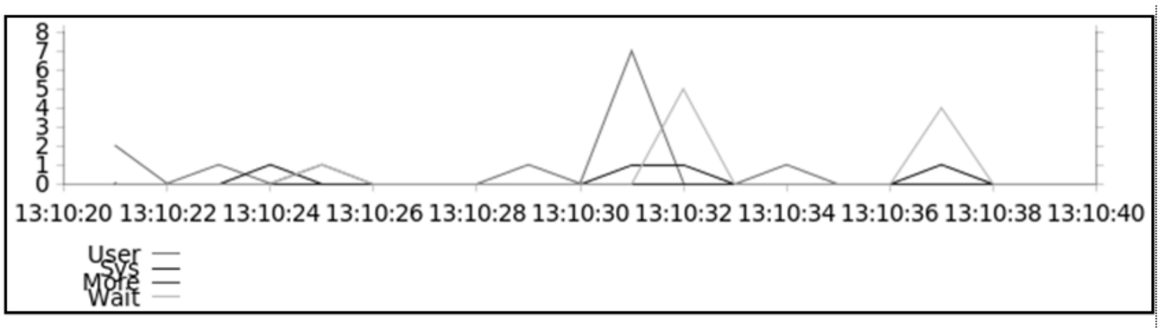


Figure 4: Idle CPU Usage

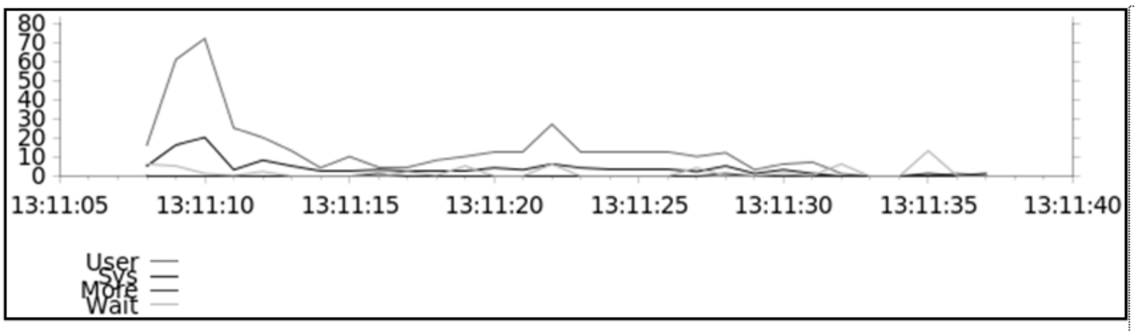


Figure 5: CPU Usage (Single Query with Two Search Engines Configured)

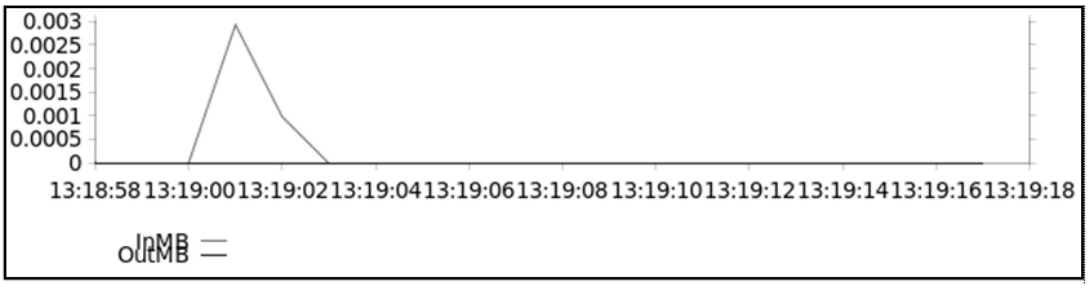


Figure 6: Idle Network Usage

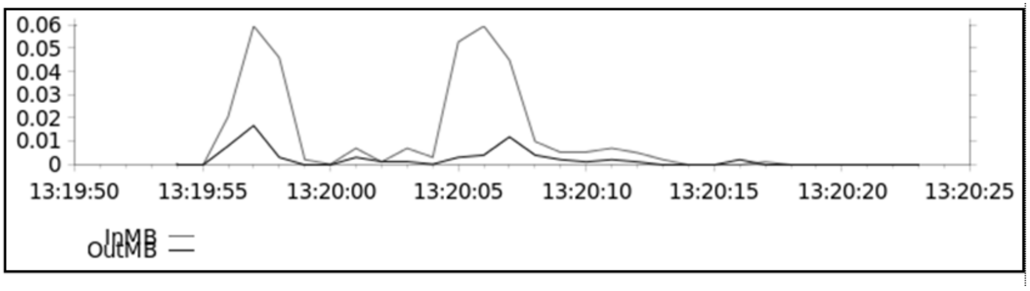


Figure 7: Network Usage (Single Query with Two Search Engines Configured)

5. CONCLUSIONS

FOSSICK provides aggregated search results from a combination of different general purpose search engines and search tools of various web applications. In general, federated search engines may also allow the user to avoid being target of framed results by proprietary search engines [13]. It improves efficiency in terms of time required to find relevant results. It can also help overcome discrimination by search engines based on IP addresses or country [14].

Colophon: Fossick is an English word of Australian origin which literally means to search for gold nuggets.

6. REFERENCES

1. *Bush, Vannevar (Jul 1945). As We May Think. The Atlantic Monthly, 176(1): 101-8*
2. *ITU, The World in 2015 (2015). Retrieved from: https://www.itu.int/en/ITU-D/Statistics/Documents/facts/ICTFactsFigures2015.pdf*
3. *Lawrence, Steve & C. Lee, Giles et al. (Jul 1999). Accessibility of information on the web. Nature, 400, 107-109*
4. *Bergman, Michael K. (2001). White Paper: The Deep Web: Surfacing Hidden Value. The Journal of Electronic Publishing, 7 (1).*

1. *5. Spink, Amanda & Jensen J., Bernard (2006). A s tudy of results overlap and uniqueness among major Web search engines. Information Processing and Management, 42 (5), 1379-1391*

5. *White, Martin (2015). Critical success factors for enterprise search, Business Information Review, 32 (2), 110-118*

6. *Kumar, Shailendra (2008). Federated Search: New Option for Libraries in the Digital Era, Proceedings of 6th International CALIBER-2008, 267-85*

7. *Clinton, D. OpenSeacrh 1.1 Specification (draft 4). Retrieved from: http://www.opensearch.org*

8. *IETF, Network Working Group (2001). The Application/RSS+XML Media Type. Retreived from: http://tools.ietf.org/id/draft-nottingham-rss-media-type-00.txt*

9. *Christopher D. Manning, Prabhakar Raghvan, Hinrich Schütze (2008). Introduction to Information Retrieval. Cambridge University Press*

10. *Collectl (2016). Retrieved from: http://collectl.sourceforge.net*

11. *Collectl Utilities (2016). Retrieved from: http://collectl-utils.sourceforge.net/colplot.html*

12. *Lao, Marina (2013). Neutral Search As A Basis for Antitrust Action?. Harvard Journal of Law & Technology Occasional Paper Series: 1–12.*

13. *Grimmlemann, James (2011). Some Skepicism about Search Neutrality. The Next Digital Decade: Essays on The Future of The Internet, 435*