ABSTRACT
In this 21st Century, globalization has become a common knack for all. In most used ways, globalization is made by creating websites. As a result, information volume of internet has increased exponentially [1]. Therefore, it is increasingly becoming difficult to retrieve specific information from the internet. Search engine plays a big role to retrieve information from internet [2][3]. Effective search result for a user given search string always increase search engine efficiency. Most of the search engines are available in the market that produced search results in two parts, i.e. one is highlighted part and the other one is main part. Most of the cases in the main part of their search results produced by appending title tag, URL and meta-tag information or first few words from the Web-page content. While searching of any information for few commonly used products like book, mobile handset, medicine, jewelry, etc., Web-searcher wants to know some basic attributes based on the product. Now, if we display these products basic attributes information in the search result instead of displaying meta-tag or first few words from the Web-page content, then no need to visit all the URLs displayed in the search result. As a result all the Web-searcher saves their time and Web-page download cost. In this paper, we have proposed a design and development mechanism of Ontology based domain specific Web-search tool for commonly used products using Resource Description Framework (RDF).

Categories and Subject Descriptors
H.3.3 [Information Search and Retrieval]: Information filtering, Retrieval models.

General Terms

Keywords
Search engine, Ontology, Ontology Based Search, Domain Specific Crawler and Resource Description Framework (RDF)

1. INTRODUCTION
Search engine is an information retrieval system from WWW (World Wide Web). Generally Web-search engines hold title tag with URL and meta-tag information or first few words from the Web-page content in their main search result (refer figure 1). Without visiting search result links Web-searcher cannot understand the Web-page content will fulfill their expectation or not. Based on this background, we have proposed an Ontology based domain specific Web-search tool for commonly used products like book, mobile handset, medicine, jewelry, etc., where most of the cases Web-searcher intention to know various types of attributes for each product.

In our approach, we have treated each product as a domain. Say, Book domain mainly wants to know ‘Book Title’, ‘Author’, ‘Price’, ‘Edition’, etc. information. On the other hand Mobile handset domain wants to know ‘Company’, ‘Model Number’, ‘Price’, ‘Color’, ‘Type’, etc. information. So fulfill Web-searcher various types of requirements we produce a RDF (Resource Description Framework) for each domain by considering Web-searcher basic intentional attributes and Web-page title from the Web-page content title tag as RDF properties [4]. Then set RDF properties value for each domain specific Web-pages, which are already crawled by domain specific crawler. For keeping track each RDF content, we store them with respect to their Web-page ID. While displaying the search result, we just append the RDF properties information along with the Web-page URL. In this approach, Web-searchers have received basic information about the product in the search result, hence no need to visit each and every result URLs. As a result this search tool saves Web-searcher time as well as Web-page download cost. On the other hand, while producing search result, no need to check Web-page content for displaying meta-tag information in the search result page at search time, hence it produces faster result.

Our paper is not intended to provide a complete survey of techniques. According to our knowledge, we have applied these techniques on few examples. Now a day’s research on search engine has been carried out in universities and open laboratories, many dot-com companies. Unfortunately, many of these techniques are used by dot-coms, and especially the resulting performance, are kept private behind company walls, or are disclosed in patents that can be comprehend and appreciate by the lawyers. Therefore, we believe that the overview of problems and techniques that we presented here can be useful.

This paper discusses survey of the problem area in section 2. Section 3 tells about the existing work. Section 4 discusses about the resource description framework. Section 5 depicts the
proposed approach. Section 6 shows some experimental analyses. Finally, section 7 concludes the paper.

2. THE PROBLEM AREA

The exponential growth rate of Web-data leads to many complications in retrieval of relevant information. In addition to this, the navigation of many links in an attempt to find desired information cause wastage of user time and makes the user annoyed. Therefore, Web-searchers are introduced domain specific Web-search engines. But most of the cases search engines produced their main search results by appending title tag, URL and meta-tag information or first few words from the Web-page content (refer figure 1). Now, few commonly used products such as book, mobile handset, medicine, jewelry, etc., where most of the cases Web-searchers main intention to know some basic information about each domains, which is shown in table 1.

<table>
<thead>
<tr>
<th>Products</th>
<th>Basic Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Book</td>
<td>Book Title, Author, Price, Publisher, Edition</td>
</tr>
<tr>
<td>Mobile handset</td>
<td>Company, Model Number, Price, Color, Type (Basic handset, Multimedia set, Touch Screen, etc.)</td>
</tr>
<tr>
<td>Medicine</td>
<td>Name, Price, Purpose, Type (Tablet, Capsule, Syrup, etc.), Major side effect</td>
</tr>
<tr>
<td>Jewelry</td>
<td>Model Number, Price, Design code, Jewelry type (Ring, Chain, etc.)</td>
</tr>
</tbody>
</table>

In general, our main goal is to identify the basic information from the search result for a Web-searcher. The example illustrates the problem in the search result, which can overcome by using our search model.

Example 1. John is looking for a mobile handset. He prefers a Samsung I5500 Galaxy handset, but preferably costing not more than $200. He planned to search with search string “Samsung I5500 Galaxy”. Unfortunately he has received lots of search results. He needs to open each and every search result links to get the basic information about the handset, which is not only time-consuming, but also it deals with Web-page download cost. In figure 1, we have presented few screenshot of search results obtained from worlds most commonly used search engines.

3. EXISTING WORK

In this section, we describe how a domain specific crawler crawls domain specific Web-pages from WWW and the utility of Ontology to create a domain specific Web-search engine [13]. We also describe a mechanism for finding domain specific Web-pages and store them in a domain specific Web-page repository.

Definition 1. Ontology - It is a set of domain related key information, which is kept in an organized way based on their importance.

Definition 2. Seed URL - It is a set of base URL from where crawler starts crawling the Web pages from Internet.

Definition 3. Weight Table - This table has two columns; first column denotes Ontology terms and second column denotes weight value of that Ontology term. Ontology term weight value lies between ‘0’ and ‘1’.

Definition 4. Syntable - This table has two columns; first column denotes Ontology terms and second column denotes synonym of that ontology term. For a particular ontology term, if more than one synonym exists, those are kept using comma (,) separator.

Definition 5. Relevance Value - It is a numeric value for each Web-page, which is generated on the basis of the term Weight value, term Synonyms, number of occurrence of Ontology terms which are existing in that Web-page.

3.1 Ontology based Domain Specific Search

The name, ‘Domain Specific Search’ means searching a topic for a particular domain. To search in a particular domain, we use Ontology for that domain. Each and every domain uses a different Ontology to express their domain interest [5]. When we are planning to search any topic that time we need to select that domain related Ontology for getting the domain related Web-pages. A domain specific crawler uses Ontology to crawl down the domain specific Web-pages from internet. A problem with
standard keyword based search queries is that it is difficult to express advanced search queries. It is possible to express richer and more accurate queries by using Ontology. The system uses Ontologies that describes the area of all products, in which the search will be performed and the user enters different parameters to perform the search [6][7].

3.2 Domain Specific Web-page Crawling
Ontology based domain specific crawling means, a crawler that find domain specific Web-pages using that domain ontology [8][9]. The term Ontology is a data model that represents a set of concepts within a domain and the relationships between those concepts. It is used to reason about the objects within that domain. Ontologies are used in artificial intelligence, the semantic Web, software engineering, biomedical informatics, library science, and information architecture as a form of knowledge representation about the world or some part of it.

For finding domains based on the Web-page content, we have to parsed the Web-page content and extract all the Ontology terms as well as syntable terms [10]. Then we have calculated the relevance value of that Web-page. Now if this relevance value is greater than the predefined relevance limit of that domain, then we consider that Web-page belongs to our domain otherwise discard the Web-page, i.e., the Web-page not belongs to our domain. We have calculated the relevance value using below formula:

\[
\text{Relevance Value (} \sigma \text{)} = \Sigma W_{ON\text{T}}N_{OTC} + \Sigma W_{SY\text{T}}N_{STC}
\]

Where, \( W_{ON\text{T}} \) denotes Ontology term weight value, \( N_{OTC} \) denotes Ontology term occurrence count in the Web-page content, \( W_{SY\text{T}} \) denotes synonym of Ontology term weight value and \( N_{STC} \) denotes synterm term occurrence count in the Web-page content.

3.3 Domain Specific Web-page Repository Creation
Domain specific crawler crawls down the domain specific Web-pages. All the crawled Web-pages are stored into a repository. We have used single domain specific crawler for each domain.

Figure 2. Domain Specific Web-page Repository
In our approach, we are working with four products, i.e. four domains. Therefore, at least four different Web-search crawlers are required to crawl down domain specific Web-pages [8]. We have stored Web-page identifier (P_ID), Web-page URL, Web-page content and calculated relevance value for each downloaded domain specific Web-pages. In figure 2, we have shown a mechanism of storing domain specific Web-pages.

4. RESOURCE DESCRIPTION FRAMEWORK
RDF is a general method to convert a document into data. For each product we have treated as a new domain and used different RDF format [11][12]. Product basic information stores in their corresponding RDF properties fields. In figure 3, we have illustrated a sample RDF content of Mobile handset domain. We have used “http://www.w3.org/1999/02/22-rdf-syntax-ns#” and “http://www.w3.org/2001/vcard-rdf-3.0#” provided syntax for generating RDF content. Web-page ID stored in “rdf:nodeID” field. The entire domain related basic information and domain name are kept in various “vcard:” tags. In “rdf:Description” tag holds “rdf:about” information, where we store Web-page URL. Finally, each Web-page content holds title tag information that information stores in “vcard:SearchStr” tag, which is used for retrieving search result based on the user given search string (refer section 5.4).

Figure 3. Sample RDF Content of Mobile Handset Domain.

5. PROPOSED APPROACH
In our approach, we have generated a new search model, which stores RDF content for each domain specific Web-pages exists in domain specific Web-page repository and offers more complete and comprehensive output.

5.1 RDF Generation Process
Domain specific crawler crawls down the domain specific Web-pages using Ontology, Syntable and Weight table. For each domain, we have stored domain specific Web-pages into their respective storage. Now, we have created some driver cursor to fetch Web-page content one by one and pass it for processing.

Figure 4. RDF Generation Process
We have different set of properties information for each domain (refer table 1), which are extracted from Web-page content and following a predefined format we have generated RDF content.
(refer figure 3). We are using two fields such as RDF available flag and error message for tracking RDF generation process. While extracting RDF properties information from Web-page content, some times we have received properties data and some times not. We have updated RDF available flag as “N” and populate error message while any error encountered or not a single property information is available in the Web-page content. For all successfully generated RDF content, we have updated RDF available flag as “Y”. Finally, we have stored the RDF content with respect to their Web-page ID. In figure 4 we have shown a block diagram of a RDF generation process.

In figure 5 and 6 illustrates how Web-page content has stored in the domain specific Web-page repository and code snippet used for generating RDF content.

5.2 Algorithm

In this subsection, this algorithm describes a mechanism of RDF contents generation from domain specific Web-page contents and storing them with respect to their Web-page ID. This process has been done in offline, i.e., while building search engine resources and it is not affecting searching time. There are lots of difficulties to identify the RDF properties information. Fetching RDF properties is a combination of both algorithmic and manual approach. Manual approach used for those cases where algorithmic approach fails and generates error. For fetching RDF property information we have used a set of similar attributes (A) for each property.

\[ A = \{ a_1, a_2, a_3, \ldots, a_n \} \]

Now, \( a_1, a_2, a_3, \ldots a_n \) all are similar attributes and prioritized based on their importance. This priority defines based on the nature of the each attribute. Let consider, ‘Price’ is a RDF property in mobile handset domain to give clarity of our attribute identification using prioritized attribute approach. There are many Web-pages available in the mobile handset domain resources, which contents price attribute as well as similar type of attributes such as ‘best price’, ‘cost’, ‘offer price’, etc. in a single Web-page content. Now, the question is which value we need to set in our RDF properties tag. Simply, we are setting available high priority attribute value in the RDF property tag. Below algorithm states how to generate our RDF content.

```
GenerateAndStoreRDFContent()
begin
  do for ever
    wait (Web-page content);
    while (empty (Web-page content))
    begin
      extract Web-page Content;
      find RDF properties information;
      generate RDF Content;
      set RDF available flag;
      store RDF Content with respect to their Web-page ID;
    end;
    signal (hungry);
  end.
end.
```

5.3 User Interface

Figure 7 shows a part of user interface of the search engine. A search string is typed in the input search string box, enter the relevance range and select the domain. In the user interface mandatory fields are denoted by star (*) sign. Relevance range default value set as [maximum relevance value, minimum relevance value], which is an editable field and according to the requirement, user can customize the relevance range values. We are also providing flexibility to the users to get only RDF resources and limited search result using ‘Only RDF Resources’ check box and ‘Number of Search Result’ field. If user wants to get only RDF available resources, then we have just checked the only RDF resources check box. Suppose, the user has given search string and relevance range value, this prototype selects 100 Web-pages based on their relevance range value. Now, user wants to restrict the number of search results (say, 20 Web-pages). Lesser time will be taken for displaying 20 Web-pages instead of displaying 100 Web-pages. Also these 20 Web-pages hold higher relevance values because output of fetched query contains Web-pages in sorted order in terms of their relevance value. In the user interface, maximum relevance value and minimum relevance value are set dynamically according to the user selected domain. While refreshing the database, maximum relevance value is taken using ceiling function for the largest Web-page’s relevance value and minimum relevance value is taken using floor function for the smallest Web-page’s relevance value. Here, four domains such as “Book”, “Mobile Handset”, “Medicine” and “Jewelry” have been considered. For domain selection we have used radio button because at a time only one domain can be selected. Initially search string parse operation has been done after clicking on the ‘Search’ button of the user interface (Web-page). Then the searching operation would be done simply according to the parsed
query on the Web-pages based on user given relevance range and selected domain.

5.4 Search Result Generation Mechanism
In this section we explain how to select Web-pages from our Web-page repository for user given inputs. Initially we have fetched the RDF contents based on the domain and relevance range. Then we have parsed the user provided search string and generate tokens. Now, we will match these tokens with the “vcard:SearchStr” tag in RDF content (refer figure 3). To produce our search result, we have considered those Web-pages where any one of the token, which generated from the input search string, match with “vcard:SearchStr” tag. In the search result page, we have displayed “vcard:SearchStr”, “rdf:about” and properties information for all the considered Web-pages where RDF available. For example, we have shown an entry in the search result page for figure 3 given RDF content.

Samsung I5500 Galaxy 5 price in India, Samsung Galaxy 5 price and specifications http://www.priceofmobiles.com/categories/desc_samsungi5500galaxy5.php
MobileHandset, Samsung, I5500, INR 8413, Touch Screen

On the other hand, in the search result we have displayed only title tag, Web-page URL and meta-tag information for those Web-pages where RDF not available.

6. EXPERIMENTAL ANALYSIS
Here we will explain our test setting and test results. We will also discuss few comparisons between existing and our prototype produced search result.

6.1 Test Setting
We have used seed URLs, weight table, syntable to create each product related domain specific Web-page repository. Seed URLs are used for starting domain specific crawler. Weight table and syntable are used for calculating relevance value of a downloaded Web-page.

6.1.1 Seed URL
Seed URL is a set of URLs from where crawler starts crawling, i.e., downloading the Web pages from WWW. For the mobile handset crawler to start crawling, we have provided some seed URL as shown in table 2, depending on mobile handset Ontology. Similarly, all other domain specific crawlers seed URLs are delivered depending on their Ontology.

<table>
<thead>
<tr>
<th>Seed URL</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://www.nokia.co.in/">http://www.nokia.co.in/</a></td>
</tr>
<tr>
<td><a href="http://www.samsung.com/GalaxySII">www.samsung.com/GalaxySII</a></td>
</tr>
<tr>
<td><a href="http://www.lg.com/">www.lg.com/</a></td>
</tr>
<tr>
<td><a href="http://www.micromaxinfo.com/">www.micromaxinfo.com/</a></td>
</tr>
</tbody>
</table>

6.1.2 Weight Table
Each Ontology term has an importance of that domain. We assign weight for each Ontology term based on its importance. The strategy of assigning weights is that, the more significant term will have more weight and the terms weight value lies between ‘0’ and ‘1’. Few weight values of mobile handset domain Ontology terms are shown in table 3.

<table>
<thead>
<tr>
<th>Ontology Term</th>
<th>Weight Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mobile</td>
<td>1.0</td>
</tr>
<tr>
<td>Handset</td>
<td>0.8</td>
</tr>
<tr>
<td>Model</td>
<td>0.4</td>
</tr>
<tr>
<td>Touch screen</td>
<td>0.2</td>
</tr>
</tbody>
</table>

6.1.3 Syntable
In order to get appropriate result for a domain specific Web-page we use syntable, which contains two fields. One field is for Ontology term and another field for synterm. In this table we store all Ontology terms and their synonyms. Some of them for mobile handset domain are shown in table 4.

<table>
<thead>
<tr>
<th>Ontology Term</th>
<th>Synterm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mobile</td>
<td>Cell phone, phone, cell</td>
</tr>
<tr>
<td>Memory</td>
<td>Space</td>
</tr>
<tr>
<td>Company</td>
<td>Organization</td>
</tr>
<tr>
<td>Network</td>
<td>Connection</td>
</tr>
</tbody>
</table>

6.2 Test Result
In this section we have generated some test results based on user given inputs. We have verified fruitfulness of our search result for a user given search string. In the successive subsections we have provided few snapshots such as RDF storage, search result, etc.

6.2.1 RDF Storage Snapshot
In figure 8 we have shown how RDF content of a domain specific Web-page has stored in our repository.

6.2.2 Search Result Display Snapshot
In figure 9 we have shown a snapshot of our search result for the search string “Samsung I5500 Galaxy”. From the user interface we have delivered “Samsung I5500 Galaxy” as search string, checked only RDF resources, selected mobile handset domain,
used default relevance range value, i.e., [386, 5] (refer figure 7) and restricted on 10 search results.

**Samsung 15500 Galaxy 5 - Full phone specifications**
Mobilenet, Samsung, 15500 Galaxy, EUR 130, Touch Screen

**Samsung Galaxy 5 15500 Preview**
Mobilenet, Samsung, 15500 Galaxy, USD 242.50, Touch Screen

**Samsung 15500 Galaxy 5 price in India, Samsung Galaxy 5 price and specifications**
http://www.priceindia.co.in/category/loc_samsung_galaxy5.php
Mobilenet, Samsung, 15500 Galaxy, INR 3413, Touch Screen

**Samsung Galaxy 5 Price, Samsung 15500 Galaxy 5 Price | MobilePhone.co.in**
http://www.mobilephone.co.in/2010/samsung-galaxy-5-price.html
Mobilenet, Samsung, 15500 Galaxy, INR 9090, Touch Screen

**Amazon.com: Samsung I5500 Corby Galaxy 5 Android Smartphone with Wi-Fi, Bluetooth, GPS, Touch Screen - No Warranty - Black: Cell Phones & Accessories**
Mobilenet, Samsung, 15500 Galaxy, USD 129.99, Touch Screen

**Samsung Galaxy 5 (I5500) First Take - Phones - CNET Asia**
Mobilenet, Samsung, 15500 Galaxy, USD 218, Touch Screen

![Figure 9. A Sample Search Result Snapshot](image)

### 6.3 Comparison between Existing and Our Search Result

In this subsection we have explained a comparison between existing and our search results by doing a search with a common search string. According to example 1 (described in section 2), John has performed a search operation using “Samsung 15500 Galaxy” as a search string. He has received a lot of search results from various search engines, which are currently available in the market (refer figure 1). But he needs to visit all the search results to get the basic information about the mobile handset. As a result he has wasted his valuable time as well as Web-page download cost. On the other hand, according to our prototype he can get the basic information about the mobile handset just seeing the search result (refer figure 9). As a result he can save his valuable time as well as Web-page download cost.

### 7. CONCLUSION

In this paper, we have proposed a prototype of a domain specific Web-search engine, which supports multiple domains. This prototype mainly designed for few commonly used products such as book, mobile handset, medicine, jewelry, etc. This prototype is highly scalable. We can expand supporting domains by introducing new products Ontology, RDF format and other details such as weight table, syntable, etc. We can also increase basic information attributes in the search result, just adding new properties in the RDF format. According to our search result compressions, we can claim that our prototype offers more complete and comprehensive output. Web-searcher can get the basic information about the product from search result page instead of visiting search result links. This prototype saves Web-searcher search time as well as Web-page download cost. On the other hand, while producing search result, no need to check Web-page content for displaying meta-tag information in the search result page at search time, hence it produces faster result.

### 8. REFERENCES


