IJESRT

INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY

Custom Web Scrapping For Content Aggregation From E-Commerce Websites

Bhushan J*[1], Nijagunarya[2]
*[1,2]Siddaganga Institute of Technology, Tumkur, Karnataka, India
vishal.vis20@gmail.com

Abstract

Today's internet user has a limited amount of time to mine the Internet manually for content such as videos, images, and documents that they want to view. In such case much of the user's time is wasted overhead: waiting for pages to load, clicking hyperlinks, and downloading the content for offline viewing. Therefore, many users would benefit from an application that could automatically crawl and download a large amount of content from the Internet. A lot of effort is put by customers and information seekers to collect useful information from E-commerce websites, information that is needed includes price of consumer products, related description, and other product related attributes. In the present situation where the websites change dynamically, there is a need for a system which is able to collect information for users irrespective of the changes to the web content. In this paper, we propose a novel approach for aggregation of information using Custom Web Scrapper. The results are discussed by procuring information from some popular E-commerce websites.

Keywords: Web scraping, E-commerce Websites, Information aggregation, HTML Parser.

Introduction

Web has come a long way from being a collection of documents to an organized, dynamically evolving entity which changes with the addition of new content every day. Any organization which seeks out information regarding products from large scale and small scale dealers obtains the same from retailers; the data so obtained is in an unorganized and raw format. It becomes difficult for the organization to organize the data obtained. There is an overhead involved with the traditional approach where useful data needs to be extracted, processed and analyzed. A need arises to address the issue of organization of data from unstructured information of the web. The data so extracted can be used by Organizations to increase the efficiency of their business process and address the in demand areas which form the focus. Users often need to browse only a portion of a Webpage. On commercial pages, for example, users probably want the price and product description and some details about product. Consequently, they are often required to search the page for the required information either by manually or by using the string search capability provided by the browser. If user's target pages are frequently modified, it is a heavy burden for the users to keep up with the latest information by repeating these Web browsing operations. To reduce this operational cost we have developed Custom Web Scraper [CWS].

Related Work

Although information is structured form inside database on the Web, such information is still flattened out for presentation, segmented into "pages" and aggregated into separate "sites". Anyone wishing to retain a piece of that information must bookmark the page and continuously repeat the effort of locating the piece of information within the page. To collect several items spread across multiple sites together, one must bookmark all the corresponding pages.

Search engines were invented to break down websites barriers letting users query the whole web rather than multiple sites separately. Some of the search engines available in market are listed below.

Chickenfoot

Chickenfoot[2] is an Firefox extension the script are written as set of java scripts which includes related function for specific web tasks. As the chickenfoot is embedded in browser itself it runs very slow because it is running with the browser which interprets javascript and Ajax calls. This makes impractical to scrap the amount of data present in million of threads. Chickenfoot interact with browser with the help of commands. Chickenfoot is primarily aimed at interaction with the browser but can also be used for scraping with the find() command. Here is an example script for scraping search results from a Google search:

```
go("www.google.com")
enter("chickenfoot")
click("Google Search")
for(m=find("link"); m.hasMatch; m=m.next) {
var link = m.element;
if(link.getAttribute("class") == "l") {
output(link.href);
}
}
```

This script searches Google for chickenfoot and returns the links that have a class of l, which from examining the Google HTML source is an attribute unique to the search result links.

Piggy Bank

Piggy bank is also Firefox extension. The end user writes the scrapping script along with regular expressions relevant to the WebPages [3]. Piggy bank scraps data when user navigates to the matching webpage with respect to user script. Disadvantage of this tool is only eleven scripts can be submitted at a point of time.

Sifter

Sifter [6] builds on top of Piggy Bank's infrastructure but tries to scrape semantic data automatically from any webpage. However the scraper has limited scope and only looks for the biggest group of links in a webpage. This is relevant to a commerce site like Amazon where the books are a series of links, but usually we will not want to extract the biggest group of links. For instance the biggest group of links in a web forum is generally navigation-related and not directly relevant to a given thread in isolation. Consequently, Sifter does not solve our scraping problem.

Scrubyt

Scrubyt is written in Ruby language. Scrubyt takes user input as key, searches for the key in webpage and extracts all the similar item from webpage. Here is the Scrubyt version of the Chickenfoot example to scrape Google search results:

```
google_data = Scrubyt::Extractor.define do
fetch "http://www.google.com/ncr"
fill_textfield "q", "ruby"
submit
link "Ruby Programming Language" do
url "href", :type => :attribute
end
end
```

This script searches Google for ruby and then uses the known title for the o_cial Ruby website to automatically build a model of the search results. It then extracts the links from this model.

Newprosoft

Web Content Extractor provides serious automation of the website scraping task. Usually, you only need to specify a basic extraction pattern (done in few clicks too) and run the extraction process. The program automatically scans the provided URLs and scrapes all the info that meets the specified template. Content extraction from urls, page links, and following the web index links, unable to adapt and address the dynamically changing nature of web. When the content in webpage is added or deleted the index of whole webpage changes.

Proposed Custom Web Scrapper (CWS)

In our proposed system, we crawl and parse all pages one by one and fetch the required data.

The custom web scraper gets the input from the database; the input contains the name of the retailer and urls of the related products.

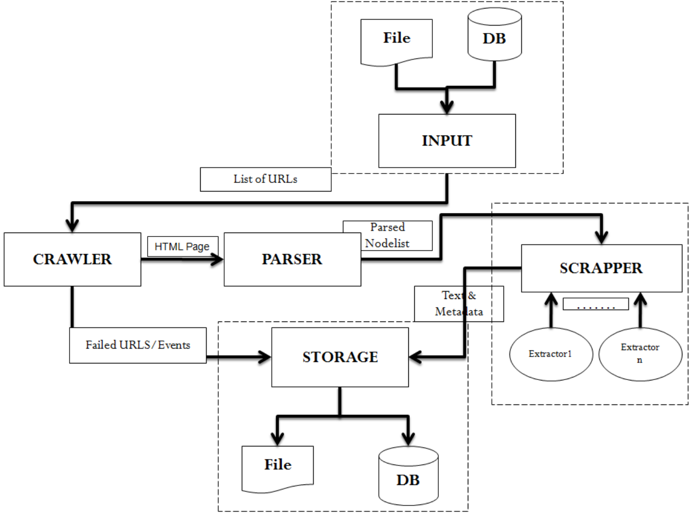


Fig 1: Architecture of Custom Web Scraper

Most of the scraping tools available in market parse the HTML page by positional indexing [7] (seed and test documents) and pattern generation [8] (seed documents only). Defined by the elements in the (X) HTML structure by partitioning the document into individual nodes, and identifying the text associated with each. The assumption made here is that each text chunk is within element boundaries and we need to identify the relative position of the element.

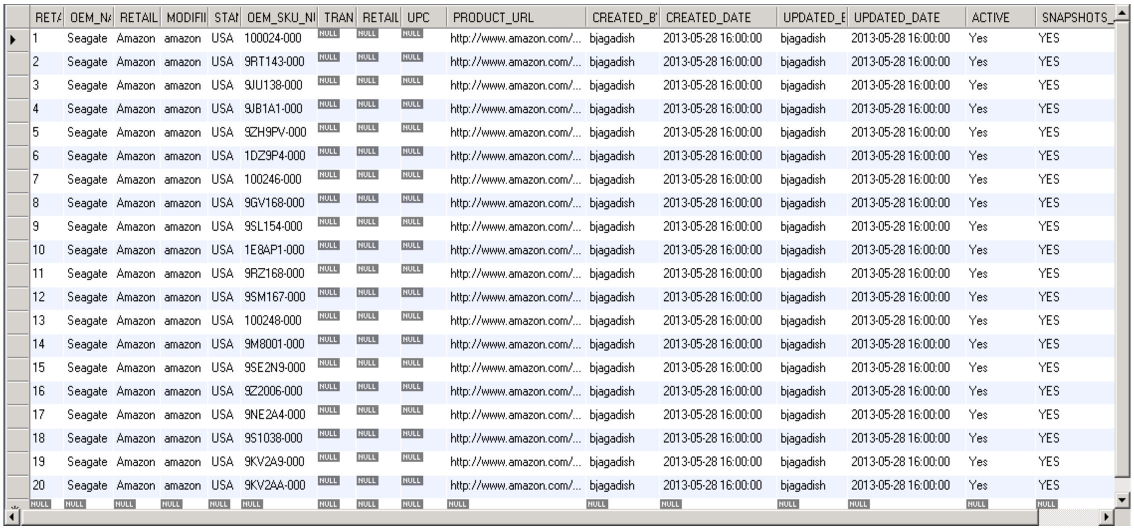
This tool is designed mainly considering commercial websites like Amazon, Flipkart bestbuy etc. The underlying assumption is that each commercial websites have same HTML layout for each products. As we has observed web pages are very volatile. The content of webpage is written and changed daily. Parsing by position (Indexing) is not a feasible idea. When the web developer modifies or adds some more data the indexing of whole page changes, the programmer need to find the new position of the required data, and pass the indexing to the crawler for scraping data. This is the painful job for the programmer to find the indexing every time when there is change in page. Keeping this in mind we undergo an assumption that all the pages from a particular site (Amazon) maintain same layout. Each element has predefined tags and class name.

Most of the product prize changes regularly with market demand and available resources for manufacturing the respective product and also retailers need to keep track of the stock – in such case it is literally impossible to manually keep track of those attributes. In all these cases our url scraping application serves its purpose by extracting data from listed url and store them in database for future use, we can use similar approach to update product listings in url whenever change is necessary.

The proposed system consists of 4 modules named INPUT, CRAWLER, SCRAPPER and OUTPUT.

INPUT Module:

Crawler gets URLs and related attributes (OEM, retailer, etc...) from Input module which is either from a file or database.



RET/	OEM_N/	RETAIL	MODIFII	STAI	OEM_SKU_NI	TRAN	RETAIL	UPC	PRODUCT_URL	CREATED_B'	CREATED_DATE	UPDATED_E	UPDATED_DATE	ACTIVE	SNAPSHOTS_
1	Seagate	Amazon	amazon	USA	100024-000	NULL	NULL	NULL	http://www.amazon.com/...	bjagadish	2013-05-28 16:00:00	bjagadish	2013-05-28 16:00:00	Yes	YES
2	Seagate	Amazon	amazon	USA	9RT143-000	NULL	NULL	NULL	http://www.amazon.com/...	bjagadish	2013-05-28 16:00:00	bjagadish	2013-05-28 16:00:00	Yes	YES
3	Seagate	Amazon	amazon	USA	9JU138-000	NULL	NULL	NULL	http://www.amazon.com/...	bjagadish	2013-05-28 16:00:00	bjagadish	2013-05-28 16:00:00	Yes	YES
4	Seagate	Amazon	amazon	USA	9JB1A1-000	NULL	NULL	NULL	http://www.amazon.com/...	bjagadish	2013-05-28 16:00:00	bjagadish	2013-05-28 16:00:00	Yes	YES
5	Seagate	Amazon	amazon	USA	9ZH9PV-000	NULL	NULL	NULL	http://www.amazon.com/...	bjagadish	2013-05-28 16:00:00	bjagadish	2013-05-28 16:00:00	Yes	YES
6	Seagate	Amazon	amazon	USA	1DZ9P4-000	NULL	NULL	NULL	http://www.amazon.com/...	bjagadish	2013-05-28 16:00:00	bjagadish	2013-05-28 16:00:00	Yes	YES
7	Seagate	Amazon	amazon	USA	100246-000	NULL	NULL	NULL	http://www.amazon.com/...	bjagadish	2013-05-28 16:00:00	bjagadish	2013-05-28 16:00:00	Yes	YES
8	Seagate	Amazon	amazon	USA	9GV168-000	NULL	NULL	NULL	http://www.amazon.com/...	bjagadish	2013-05-28 16:00:00	bjagadish	2013-05-28 16:00:00	Yes	YES
9	Seagate	Amazon	amazon	USA	9SL154-000	NULL	NULL	NULL	http://www.amazon.com/...	bjagadish	2013-05-28 16:00:00	bjagadish	2013-05-28 16:00:00	Yes	YES
10	Seagate	Amazon	amazon	USA	1E8AP1-000	NULL	NULL	NULL	http://www.amazon.com/...	bjagadish	2013-05-28 16:00:00	bjagadish	2013-05-28 16:00:00	Yes	YES
11	Seagate	Amazon	amazon	USA	9RZ168-000	NULL	NULL	NULL	http://www.amazon.com/...	bjagadish	2013-05-28 16:00:00	bjagadish	2013-05-28 16:00:00	Yes	YES
12	Seagate	Amazon	amazon	USA	9SM167-000	NULL	NULL	NULL	http://www.amazon.com/...	bjagadish	2013-05-28 16:00:00	bjagadish	2013-05-28 16:00:00	Yes	YES
13	Seagate	Amazon	amazon	USA	100248-000	NULL	NULL	NULL	http://www.amazon.com/...	bjagadish	2013-05-28 16:00:00	bjagadish	2013-05-28 16:00:00	Yes	YES
14	Seagate	Amazon	amazon	USA	9M8001-000	NULL	NULL	NULL	http://www.amazon.com/...	bjagadish	2013-05-28 16:00:00	bjagadish	2013-05-28 16:00:00	Yes	YES
15	Seagate	Amazon	amazon	USA	9SE2N9-000	NULL	NULL	NULL	http://www.amazon.com/...	bjagadish	2013-05-28 16:00:00	bjagadish	2013-05-28 16:00:00	Yes	YES
16	Seagate	Amazon	amazon	USA	9Z2006-000	NULL	NULL	NULL	http://www.amazon.com/...	bjagadish	2013-05-28 16:00:00	bjagadish	2013-05-28 16:00:00	Yes	YES
17	Seagate	Amazon	amazon	USA	9NE2A4-000	NULL	NULL	NULL	http://www.amazon.com/...	bjagadish	2013-05-28 16:00:00	bjagadish	2013-05-28 16:00:00	Yes	YES
18	Seagate	Amazon	amazon	USA	9S1038-000	NULL	NULL	NULL	http://www.amazon.com/...	bjagadish	2013-05-28 16:00:00	bjagadish	2013-05-28 16:00:00	Yes	YES
19	Seagate	Amazon	amazon	USA	9KV2A9-000	NULL	NULL	NULL	http://www.amazon.com/...	bjagadish	2013-05-28 16:00:00	bjagadish	2013-05-28 16:00:00	Yes	YES
20	Seagate	Amazon	amazon	USA	9KV2AA-000	NULL	NULL	NULL	http://www.amazon.com/...	bjagadish	2013-05-28 16:00:00	bjagadish	2013-05-28 16:00:00	Yes	YES
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Fig 2: Input fed to the CWS.

Original equipment manufacturer (OEM) is the manufacturer's name of the product, Retailer purchase large quantities of goods directly from manufacturer and sells to the consumer, some of the retailers are Amazon, Best Buy, Flipkart etc. Stock-Keeping Unit (SKU) is the unique ID for the product given by manufacturer. The URLs to be crawled are stored in INPUT. Here the INPUT is either a Database or a file.

CRAWLER Module:

The crawler gets list of URLs as input from the INPUT. User can specify the criteria to crawl according to OEM or RETAILER or All. According to the criteria crawler fetches the data from INPUT. When the CRAWLER ask for URLs the INPUT fetches the Database and place all the URLs in a List data structure, later passes this List to the CRAWLER. The CRAWLER parses [] all the URLs one by one, get the HTML page and place all the nodes in NodeList. This NodeList is passed to the SCRAPER component.

```
CRAWLER_ALGORITHM (USER_CRITERIA)
BEGIN
READ URL, INPUT_ATTRIBUTES from
DATA_STORE USING USER_CRITERIA
FOR EACH URL
        RETRY_COUNT <- 1
        READ_PASSED <- false
        WHILE RETRY_COUNT <= 5
                READ HTML Page
                IF NO_READ_ERROR
                THEN
                        READ_PASSED <- true
                        BREAK
                END IF
RETRY_COUNT <- RETRY_COUNT + 1
        END WHILE
        IF READ_PASSED
        THEN
                PARSE HTML AS Nodes s
                CALL SCRAPPER_COMPONENT WITH Nodes
                CALL OUTPUT_WRITER WITH OUTPUT_ATTRIBUTE, INPUT_ATTRIBUTE

        END IF
END FOR
END
```

SCRAPPER Module:

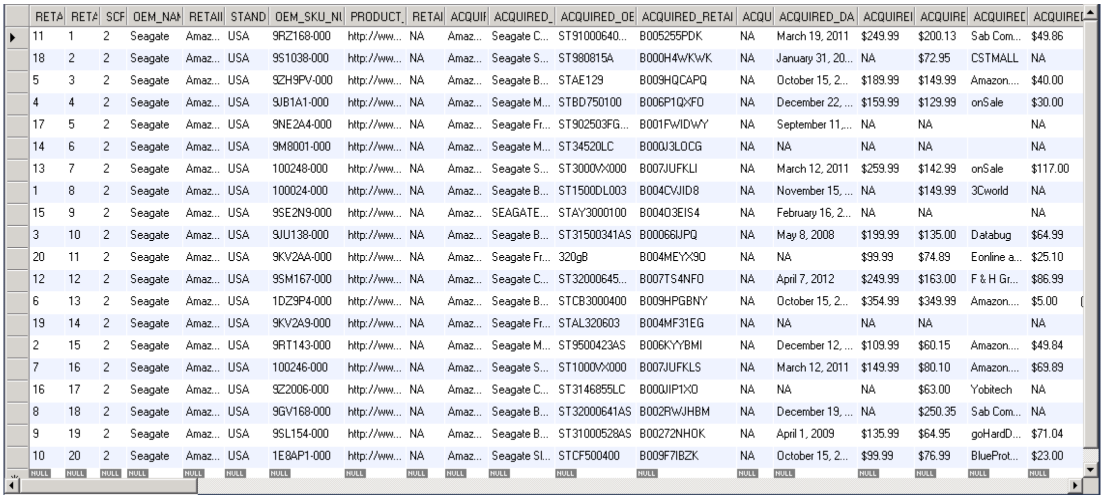
Here in **SCRAPPER Module** desired data is fetched and logs the triggered events in database. When required data is not fetched, an event is logged to the database such as CRAWLER fails to load the URL or fails to parse the URL.

Since each retailer will have different HTML layout, it is necessary to have different scrapper component for different retailers. SCRAPER Checks for which Retailer it is being called and plug-in the corresponding extractor. For Ex: If an Amazon URL is being Crawler SCRAPER initiates the corresponding extractor i.e. Amazon extractor which scrapes the related data from the Amazon URL.SCRAPER component writes the events to database which is triggered by the extractor, events such as failed to scrap the provided attributes, new layout etc. When the **SCRAPER** triggers any event it notifies the user by sending mail regarding cause for the event and input details.

```
SCRAPPER_COMPONENT (Nodes)
BEGIN
        FOR EACH Node in Nodes
                GET PRICE_DATA FROM Node
                STORE PRICE_DATA INTO OUTPUT_ATTRIBUTES
        END FOR
        RETURN OUTPUT_ATTRIBUTES
END
```

OUTPUT Module:

Once the SCRAPER completes the task it writes the scraped data to the database or File. The fig 3 shows the scraped content for the giving input as described in Input module.



RETA	RETA	SCF	OEM_NAN	RETAII	STAND	OEM_SKU_NU	PRODUCT_	RETAI	ACQUII	ACQUIRED_	ACQUIRED_OE	ACQUIRED_RETAI	ACQU	ACQUIRED_DA	ACQUIREI	ACQUIRE	ACQUIREC	ACQUIREC
11	1	2	Seagate	Amaz...	USA	9RZ168-000	http://ww...	NA	Amaz...	Seagate C...	ST91000640...	B005255PDK	NA	March 19, 2011	$249.99	$200.13	Sab Com...	$49.86
18	2	2	Seagate	Amaz...	USA	9S1038-000	http://ww...	NA	Amaz...	Seagate S...	ST980815A	B000H4WKWK	NA	January 31, 20...	NA	$72.95	CSTMALL	NA
5	3	2	Seagate	Amaz...	USA	9ZH9PV-000	http://ww...	NA	Amaz...	Seagate B...	STAE129	B009HQCAPQ	NA	October 15, 2...	$189.99	$149.99	Amazon....	$40.00
4	4	2	Seagate	Amaz...	USA	9JB1A1-000	http://ww...	NA	Amaz...	Seagate M...	STBD750100	B006P1QXFO	NA	December 22, ...	$159.99	$129.99	onSale	$30.00
17	5	2	Seagate	Amaz...	USA	9NE2A4-000	http://ww...	NA	Amaz...	Seagate Fr...	ST902503FG...	B001FWIDWY	NA	September 11,...	NA	NA		NA
14	6	2	Seagate	Amaz...	USA	9M8001-000	http://ww...	NA	Amaz...	Seagate M...	ST34520LC	B000J3LOCG	NA	NA	NA	NA		NA
13	7	2	Seagate	Amaz...	USA	100248-000	http://ww...	NA	Amaz...	Seagate S...	ST3000VX000	B007JUFKLI	NA	March 12, 2011	$259.99	$142.99	onSale	$117.00
1	8	2	Seagate	Amaz...	USA	100024-000	http://ww...	NA	Amaz...	Seagate B...	ST1500DL003	B004CVJID8	NA	November 15, ...	NA	$149.99	3Cworld	NA
15	9	2	Seagate	Amaz...	USA	9SE2N9-000	http://ww...	NA	Amaz...	SEAGATE...	STAY3000100	B004O3EIS4	NA	February 16, 2...	NA	NA		NA
3	10	2	Seagate	Amaz...	USA	9JU138-000	http://ww...	NA	Amaz...	Seagate B...	ST31500341AS	B00066IJPQ	NA	May 8, 2008	$199.99	$135.00	Databug	$64.99
20	11	2	Seagate	Amaz...	USA	9KV2AA-000	http://ww...	NA	Amaz...	Seagate Fr...	320gB	B004MEYX9O	NA	NA	$99.99	$74.89	Eonline a...	$25.10
12	12	2	Seagate	Amaz...	USA	9SM167-000	http://ww...	NA	Amaz...	Seagate C...	ST32000645...	B007TS4NFO	NA	April 7, 2012	$249.99	$163.00	F & H Gr...	$86.99
6	13	2	Seagate	Amaz...	USA	1DZ9P4-000	http://ww...	NA	Amaz...	Seagate B...	STCB3000400	B009HPGBNY	NA	October 15, 2...	$354.99	$349.99	Amazon....	$5.00
19	14	2	Seagate	Amaz...	USA	9KV2A9-000	http://ww...	NA	Amaz...	Seagate Fr...	STAL320603	B004MF31EG	NA	NA	NA	NA		NA
2	15	2	Seagate	Amaz...	USA	9RT143-000	http://ww...	NA	Amaz...	Seagate M...	ST9500423AS	B006KYYBMI	NA	December 12, ...	$109.99	$60.15	Amazon....	$49.84
7	16	2	Seagate	Amaz...	USA	100246-000	http://ww...	NA	Amaz...	Seagate S...	ST1000VX000	B007JUFKLS	NA	March 12, 2011	$149.99	$80.10	Amazon....	$69.89
16	17	2	Seagate	Amaz...	USA	9Z2006-000	http://ww...	NA	Amaz...	Seagate C...	ST3146855LC	B000JIP1XO	NA	NA	NA	$63.00	Yobitech	NA
8	18	2	Seagate	Amaz...	USA	9GV168-000	http://ww...	NA	Amaz...	Seagate B...	ST32000641AS	B002RWJHBM	NA	December 19, ...	NA	$250.35	Sab Com...	NA
9	19	2	Seagate	Amaz...	USA	9SL154-000	http://ww...	NA	Amaz...	Seagate B...	ST31000528AS	B00272NHOK	NA	April 1, 2009	$135.99	$64.95	goHardD...	$71.04
10	20	2	Seagate	Amaz...	USA	1E8AP1-000	http://ww...	NA	Amaz...	Seagate Sl...	STCF500400	B009F7IBZK	NA	October 15, 2...	$99.99	$76.99	BlueProt...	$23.00
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Fig 3: Output data extracted from CWS tool.

```
OUTPUT_WRITER (OUTPUT_ATTRIBUTES, INPUT_ATTRIBUTES)
BEGIN
        WRITE OUTPUT_ATTRIBUTES, INPUT_ATTRIBUTES INTO DATA_STORE
END
```

Results Produced

While developing the CWS we fine-tuned to work well on E-commerce sites. We ran the CWS against new prosoft tool and the results obtained are better, the fig. 3 shows the compared scrapping output which has extracted the content such as List Price, Price, SKU Number and ASIN.

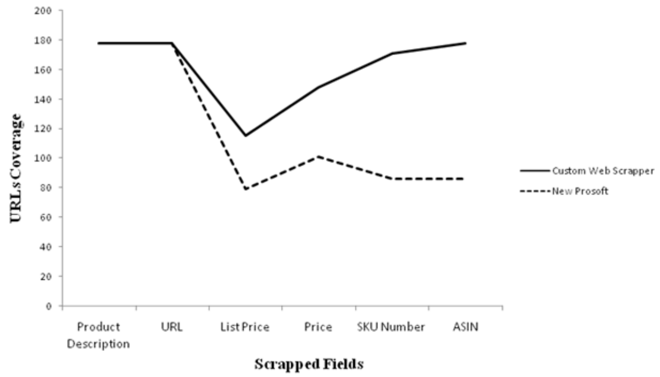


Fig 2: Scraped comparison between CWS and New Prosoft

With 178 url scrapped with both model, results shows that – using CUSTOM_WEB_SCRAPPER 115 List price are extracted successfully out of 178 resulting in 65% coverage where as at the same time NEW_PROSOFT is only able to extract 79 resulting in 44% coverage, similarly for price and SKU Number CUSTOM_WEB_SCRAPPER outperform NEW_PROSOFT with 83%-57% and 96%-48% coverage respectively.

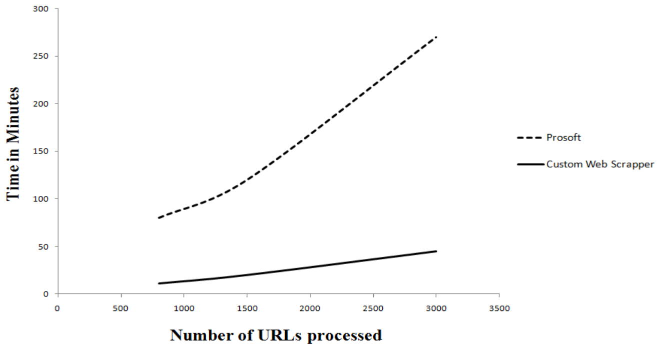


Fig 3: Time taken by the CWS against New Prosoft

The figure 3 clearly shows that the time taken by the Custom WebScraper to handle the number of URL requests is very less compared to that of NewProsoft. Thus custom WebScraper proves to be efficient in terms of time taken to handle multiple numbers of requests and scrape the accurate required data.

Future Work

Our Custom Web Scraper results are overwhelming and efforts will be made to improve the results obtained. Input module gets the url from database which had the copy of url content of remote database and we need to explicitly update url database – as future work we will make this update happening in real time. The whole project has been written in java and we would like to build this functionality into browser such that it helps people to compare product with rival retailers against the prize.

Conclusion

Custom Web Scraper has been tested with different retailer url's on e-commerce websites such as amazon, ebay, BestBuy and many more retailers. The result obtained won't be affected as long as the html tags are same for given retailer. Robustness of this result helps many organisations who are handling the product details of OEM for retailer websites.

References

[1] M. Bolin, M. Webber, P. Rha, T. Wilson, and R. Miller. Automation and customization of endered web pages. In UIST '05: Proceedings of the 18th Annual ACM symposium on User Interface Software and Technology, pages 163{172, New York, USA, 2005.

[2] Lars Graammel, Margaret-Anne Storey "An End User Perspective on Mashup Makers" University of Victoria Technical Report DCS-324-IR, September 2008

[3] D. Huynh, S. Mazzocchi and D. Karger "Piggy Bank: Experience the Semantic Web Inside Your Web Browser"

[4] R. B. Penman, T. Baldwin, D. Martinez "Web Scrapping Made Simple with SiteScrapper".

[5] Hammer, Joachim, Hector Garcia-Molina, Junghoo Cho, Rohan Aranha, and Arturo Crespo. "Extracting Semistructured Information from the Web." (1997).

[6] D. Huynh, R. Miller, and D. Karger. Enabling web browsers to augment web sites' filtering and sorting functionalities. In UIST '06: Proceedings of the 19th Annual ACM symposium on User Interface Software and Technology, pages 125-134, New York, USA, 2006.

[7] Pan, Alberto, et al. "Semi-Automatic Wrapper Generation for Commercial Web Sources" Engineering Information Systems in the Internet Context 231 (2002): 265-283.

[8] Fernández Villamor, José Ignacio, Jacobo Blasco Garcia, Carlos Angel Iglesias Fernandez, and Mercedes Garijo Ayestaran. "A Semantic Scraping Model for Web Resources-Applying Linked Data to Web Page Screen Scraping." (2011): 451-456.