ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Special Issue 2, January 2018

A Review Study of Framework Designing for Smart Crawler

GurdasSingh[1],Brijesh Bakariya[2]

Research Scholar, IKGPTU, Jalandhar, INDIA[1]
ggurdas@gmail.com
Assistant Professor, Computer Science, IKGPTU Campus, Hoshiarpur, INDIA[2]

Abstract: To leverage the large volume information buried in deep web, previous work has proposed a number of techniques and tools, including deep web understanding and integration, hidden web crawlers, and deep web samplers. Also the improvement issue, request assurance has similarly been shown as bolster learning issue. In these models, a crawler and an objective data source are considered as an administrators and nature independently. By then its assurance framework will be quickly adjusted by learning past addressing results and makes note of at most two-phase long two-stage remunerate.This paperalso displays an investigation of existing smart crawling techniques and a proposed solution over the shortcoming of existing techniques.

Keywords: Crawling, Query analysis, review, semantic approach and smart crawler

I. Introduction

Web crawling is an essential bit of a web crawler. Despite an unrivaled crawler, a legitimate searching methodology is fundamental to propel the crawl adequacy. Since the amount of URLs additions exponentially with the crawling significance, the seed assurance is a basic component to achieve a high searching adequacy [1]. An ordinary crawler joins a seed scheduler or submitter, an association extractor, a URL shipper, duplicate URL eliminator, a URL sorted out, a URL wild, a fetcher, and a movement of URL channels [2]. For a normal vertical (focused) crawler, for instance, the hidden crawler, the channels are especially adjusted.

A. The Crawler/Spider Module

To find the information on the web where endless site pages exist, an ordinary web look device uses remarkable programming called crawly crawlies or crawlers to collect courses of action of the words found on the destinations. A web crawler is a program that actually explores the web by downloading records and taking after associations from page to page [3]. Web searchers use web crawler to gather data for ordering the pages; Crawlers are the robotized ventures that take after the associations found on the webpage pages. The program i.e. Web Explorer, sends HTTP requests (hypertext trade tradition), the most generally perceived tradition on the web which is used to recuperate the webpage pages and to download and uncover to them on the customer's PC structure. A web searcher doles out a few URLs as the beginning URL's for every web crawler. The work stream of a web crawler can be depicted as; a web crawler starts working with picking the early on URLs. The web crawler issues a requesting (i.e. GET claim) to recoup the site page. It then focuses most of the URLs (current URL's) in the site pages. The web crawler adds them to the lined URLs [4].

B. The Repository/Database Module

The repository or database has an unlimited amassing of data things. Each site page recouped by the crawler is pressed and after that set away in the storage facility with an intriguing ID associated with the URL and a note is taken of the length of each page [5].

C. The Indexer/Link Analysis Module

The information is accessible in the database in sweeping aggregate so the information of site pages is to be secured in the most critical demand. It note worthily affects web look as indexer takes a gathering of data or reports and makes a searchable record. There could be different records in light of the substance of the pages so that the crawler can record the information required by the customer. The indexer parses out every one of the associations in each site page and stores fundamental information about them in a hooks record. Stay record contains enough information to make sense of where each association shows from and, and the substance of the association [6]. Each web searcher makes its own particular custom word reference or vocabulary, which needs to join

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Special Issue 2, January 2018

each new "term" found after a crawl. The vocabulary stores every word with the objective that it is aware of the words entered by the customer in the question. Join examination recognizes the predominance and information how substance stay can by and large update the presentation of pages showed by links.

D. The Retrieval/Ranking Module

The recovery means to find the records related with the request term. It determines the scores for the reports using a positioning figuring. This module is the inside fragment of any web searcher. Positioning calculations are planned for capable execution progression to bring the relevant pages. The recuperation module tackles the yield records of the indexer. It recognizes the customer request from the question interface, executes the some bit of the request over its a bit of rundown and returns sorted results back to the request interface. The indexer performs different limits. It examines the chronicle, un-pack the reports, and parses them. The hits record the amount of snaps on the website pages that is gotten through web crawler using planning technique with customer address. The indexer courses the hits into a game plan of barrels that makes a mostly sorted forward document which records the URL's to the customer requested according to the outstanding doc Id distributed to each web URL. Page positioning procedures are associated, which plan the reports out and out of their hugeness, essentialness and rank score orchestrate the site page [8]. A couple of calculations depend just on the association structure of the records i.e. their rank scores, however a couple of calculations look for the substance or data in the records. Page rank estimation allots numerical weight to hyperlinked reports recorded by a web engine. [7] proposed a secure hash message authentication code. A secure hash message authentication code to avoid certificate revocation list checking is proposed for vehicular ad hoc networks (VANETs). The group signature scheme is widely used in VANETs for secure communication, the existing systems based on group signature scheme provides verification delay in certificate revocation list checking. In order to overcome this delay this paper uses a Hash message authentication code (HMAC). It is used to avoid time consuming CRL checking and it also ensures the integrity of messages. The Hash message authentication code and digital signature algorithm are used to make it more secure . In this scheme the group private keys are distributed by the roadside units (RSUs) and it also manages the vehicles in a localized manner. Finally, cooperative message authentication is used among entities, in which each vehicle only needs to verify a small number of messages, thus greatly alleviating the authentication burden.

E. The User Query Interface

The customer enters a request related to the information required by the customer to the graphical customer interface gave by the web list. Most web interfaces are amazingly essential; applications may use structures to make the customer display a query.

II. BACKGROUND

With the Mercator web crawler, Heydon and Najork exhibited a "diagram plan" for web crawlers [8, 9]. Mercator was composed in Java, exceptionally versatile, and effectively extensible. The primary rendition [8] was non-dispersed; a later conveyed form [9] divided the URL space over the crawlers as per host name, and maintained a strategic distance from the potential bottleneck of an incorporated URL server. The second Mercator paper gave insights of a 17-day, four-machine crawler that secured 891 million pages. Mercator was utilized as a part of various web mining ventures [10, 11, 12], and in 2001 supplanted the original AltaVista crawler.

Shkapenyuk and Suel'sPolybot web crawler [13] speaks to another "diagram plan." Polybot is a dispersed framework, comprising of a crawl chief process, different downloader forms, and a DNS resolver handle. The paper depicts adaptable information structures for the URL outskirts and the "seen-URL" set used to abstain from crawling the same URL different times; it likewise talks about strategies for guaranteeing respectfulness without backing off the crawl. Polybot could download 120 million pages more than 18 days utilizing four machines.

The IBM Web Fountain crawler [14] spoke to another mechanical quality plan. The Web Fountain crawler was completely circulated. The three noteworthy segments were multi-strung crawling procedures ("Ants"), copy location forms in charge of recognizing downloaded pages with close copy content, and a focal controller handle in charge of allotting work to the Ants and for checking the general condition of the framework. Web Fountain highlighted an exceptionally adaptable crawl booking instrument that permitted URLs to be organized, kept up a good manners arrangement, and even permitted the strategy to be changed on the fly. It was outlined starting from the earliest stage to bolster incremental crawling, i.e., the procedure of re-searching pages routinely in light of their verifiable change rate. The Web Fountain crawler was composed in C++ and utilized MPI (the Message Passing Interface) to encourage correspondence between the different procedures. It was allegedly sent on a bunch of 48 searching machines [15].

Ubi-Crawler [16] is another versatile conveyed web crawler. It utilizes reliable hashing to segment URLs as indicated by

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Special Issue 2, January 2018

their host segment crosswise over searching machines, prompting effortless execution debasement in case of the disappointment of a crawling machine. Ubi-Crawler could download around 10 million pages for every day utilizing five crawling machines. Ubi-Crawler has been utilized for investigations of properties of the African web [17] and to gather a few reference accumulations of site pages [18].

For all these approaches, the ability to crawl deep web is a key challenge. Olston and Najork systematically present that crawling deep web has three steps: locating deep web content sources, selecting relevant sources and extracting underlying content [19]. Following their statement, we discuss the two steps closely related to this work as below.

A. Locating Deep Web Content Sources

A recent study shows that the harvest rate of deep web is low — only 647,000distinctwebformswerefoundbysampling25 millionpagesfromtheGoogleindex(about2.5%)[27], [33]. Generic crawlers are mainly developed for characterizing deep web and directory construction of deep web resources, which do not limit search on a specific topic, but attempt to fetch all searchable forms [10], [11], [12], [13], [14]. The Database Crawler in the MetaQuerier [10] is designed for automatically discovering query interfaces. Database Crawler first finds root pages by an IP-based sampling, and then performs shallow crawling to crawl pages within a web server starting from a given root page. The IPbased sampling ignores the fact that one IP address may have several virtual hosts [11], thus missing many websites. To overcome the drawback of IPbased sampling in the Database Crawler, Denis et al. propose a stratified random sampling of hosts to characterize national deep web [13], using the Hostgraph provided by the Russian search engine Yandex. I-Crawler [14] combines pre-query and post-query approaches for classification of searchable forms.

B. Selecting Relevant Sources

Existing hidden web directories [34], [8], usually have low coverage for relevant online databases [23], which limits their ability in satisfying data access needs [35]. Focused crawler is developed to visit links to pages of interest andavoidlinkstooff-topicregions[17],[36],[15],[16]. Soumen et al. describe a best-first focused crawler, which uses a page classifier to guide the search [17]. The classifier learns to classify pages as topic-relevant or not and gives priority to links in topic relevant pages. However, a focused best-first crawler harvests only 94 movie search forms after crawling 100,000 movie related pages [16]. An improvement to the best-first crawler is proposed in [36], where instead of following all links in relevant pages; the crawler used an additional classifier, the apprentice, to select the most promising links in a relevant page. The baseline classifier gives its choice as feedback so that the apprentice can learn the features of good links and prioritize links in the frontier. The FFC [15] and ACHE [16] are focused crawlers used for searching interested deep web interfaces.

The FFC contains three classifiers: a page classifier that scores the relevance of retrieved pages with a specific topic, a link classifier that prioritizes the links that may lead to pages with searchable forms, and a form classifier that filters out non-searchable forms. ACHE improves FFC with an adaptive link learner and automatic feature selection. SourceRank [20], [21] assesses the relevance of deep web sources during retrieval. Based on an agreement graph,SourceRankcalculatesthestationaryvisitprobability of a random walk to rank results. Different from the crawling techniques and tools mentioned above, SmartCrawler is a domain-specific crawler for locating relevant deep web content sources. SmartCrawler targets at deep web interfaces and employs a two-stage design, which not only classifies sites in the first stage to filter out irrelevant websites, but also categorizes searchable forms in the second stage. Instead of simply classifying links as relevant or not, SmartCrawler first ranks sites and then prioritizes links within a site with another ranker.

III. Review Analysis of Recent Studies

*E. Mustafa et. al., A Model-Based Approach for Crawling Rich Internet Applications [37]*New Web innovations, similar to AJAX, result in more responsive and intuitive Web applications, infrequently called Rich Internet Applications (RIAs). Crawling methods produced for conventional Web applications are not adequate for crawling RIAs. The powerlessness to crawl RIAs is an issue that should be tended to for in any event making RIAs searchable and testable. This work showed another system, called "demonstrate based crawling", that can be utilized as a premise to plan effective crawling procedures for RIAs. This work showed object based crawling with a specimen system, called the "hypercube procedure". The exhibitions of our model-based crawling methodologies are looked at against existing standard crawling procedures, including broadness to begin with, profundity in the first place, and a covetous system. Trial comes about demonstrate that our model-based crawling methodology is altogether more productive than these standard procedures.

The outcomes have acquired demonstrate that more effective crawling methodologies are conceivable utilizing a model-based crawling technique. In spite of the fact that the hypercube procedure is a decent case to show how the model-based crawling functions and has a decent execution contrasted with the current systems utilized so far in the related works, the test concentrate made it clear to us that the hypercube presumptions are excessively strict for most genuine RIAs; they are abused again and again. The model-based crawling methodologies outlined after the hypercube demonstrate, the menu and the likelihood, have more reasonable suspicions. Consequently, they are a great deal more productive for genuine application than whatever other known crawling technique. Nonetheless, there are as yet numerous opportunities for development. This work closed the research with a discourse of some future research directions.

*Y. He et. al., Crawling deep web entity pages [38]*Deep-web crawl is concerned with the issue of surfacing hidden content behind pursuit interfaces on the Web. While look interfaces from some profound sites uncover printed content (e.g., Wikipedia, Twitter, PubMed, and so on), a noteworthy segment of profound sites, including all shopping locales, clergyman organized elements rather than content reports. Crawling of such element arranged substance can be valuable for an assortment of purposes. Ithascontrived a model framework that represents considerable authority in crawling substance situated profound sites. The main attention on elements permits a few imperative advancements that set our framework apart from existing work. This paper depicted the imperative segments in our framework, each handling a sub-issue including query generation, exhaust page separating and URL deduplication. The objective of this paper is to share meets and discoveries in building the entity-oriented prototype system.

This work build up a model framework that spotlights on crawling element situated profound sites. Concentrating on element arranged destinations permit to improve our crawl framework by utilizing certain attributes of these substance locales. Three such improved parts are depicted in detail in this paper, to be specific, querygeneration, and purge page sifting and URL deduplication.

*Ch. Sheng et. al., Optimal algorithms for crawling a hidden database in the web [39]*A hidden database alludes to a dataset that an association makes available on the web by permitting clients to issue queries through a search interface. At the end of the day, information obtaining from such a source is not by taking after static hyper-joins. Rather, information is gotten by questioning the interface, and perusing the outcome page progressively created. This, with different certainties, for example, the interface may answer a question just mostly, has kept hidden databases from being crept adequately by existing web indexes. This paper cures the issue by offering calculations to concentrate all the tuples from a hidden database. The calculations are provably productive, to be specific, they fulfill the assignment by performing just few questions, even in the most pessimistic scenario. It likewise set up hypothetical outcomes demonstrating that these calculations are asymptotically ideal – i.e., it is difficult to enhance their proficiency by more than a steady component. The deduction of our upper and lower bound outcomes uncovers noteworthy knowledge into the qualities of the fundamental issue. Broad trials affirm the proposed strategies work extremely well on all the genuine datasets inspected.

This paper also assaulted an issue that lies at the heart of the issue, in particular, how to crawl a hidden database completely with the littlest cost. It created calculations for taking care of the issue when the hidden dataset has just numeric characteristics, just all out properties, or both. Every one of calculations are asymptotically ideal, i.e., none of them can be enhanced by more than consistent circumstances in the most pessimistic scenario. The hypothetical examination has additionally uncovered the elements that decide the hardness of the issue, and also how much impact each of those variables has on the hardness.

*N. Dalviet. al., Sampling hidden objects using nearest-neighbor oracles [40]*Given an unknown of articles inserted in the Euclidean plane and a closest neighbor prophet, how to gauge the set size and different properties of the items? This paper proposed a proficient strategy that uses the Voronoi dividing of the space by the items and a closest neighbor prophet. The technique can be utilized as a part of the hidden web/databases setting where the objective is to assess the quantity of specific objects of intrigue. Here, itis expected that each protest has a geographic area and the closest neighbor prophet can be acknowledged by applications, for example, maps, nearby, or store-locator APIs. It showed the execution of our technique on a few real-world datasets.

This paper also considered the issue of assessing totals over hidden information questions on an arrangement utilizing a top-k closest neighbor prophet. This obliged us to create systems for testing consistently from the arrangement of items. The key specialized commitment of this paper lies in a novel calculation EdgeChase to process the region of a Voronoi cell of a protest utilizing the closest neighbor prophet. That the quantity of prophet calls made by EdgeChase is direct in the quantity of edges of the Voronoi cell makes this system proficient. Utilizing this device a total can be evaluated by testing an irregular point, finding the closest protest and separating the estimation of the capacity at that question by the range of the Voronoi cell of a similar object.

*T.Kabischet. al., Deep web integration with visqi [41]*This paper showedVisQI (VISual Query interface Integration framework), a Deep Web mix framework. VisQI is prepared to do (1) changing Web question interfaces into progressively organized portrayals, (2) of grouping them into application spaces and (3) of coordinating the components of various interfaces. Consequently VisQI contains answers for the significant difficulties in building Deep Web mix frameworks. The framework joins a full-edged assessment framework that naturally analyzes created information structures against a highest quality level. VisQI has a structure like architecture with the end goal that different designers can reuse its segments effortlessly. It might show the whole scope of functionalities of VisQI utilizing URL's of genuine Deep Web sources. The guest will be urged to take an interest in an intuitive mode.

For example, the guest can supply Deep Web wellsprings of his/her decision. It might dissect these sources with VisQI. To start with, the consequence of the extraction will be examined together with the guest. At that point it will show the order part. It will request that the framework characterize the interface picked by the client. At that point the coordinating is illustrated. The new interface is coordinated against existing interfaces. It will control the guest through the way toward confirming the coordinating applicants. To showthe use of the group mode, the guest may choose various interfaces, new or existing ones.

IV. Conclusion

The Semantic Web gives an approach to encode data and information on site pages in a frame that is simpler for PCs to comprehend and prepare. While site classifier checks the topical significance the semantic classifiers fabricate the search question answers (joins). Semantic classifier checks if enough semantic data is available in the url and arranges it where semantic data is accessible or not. Client is scanning for semantic data and the internet searcher gives that data. So client can now rate instead of RDF is really applicable to his search and he rates in view of that. Because of along these lines next time when whatever other client seeks, he can see that RDF in top on the off chance that it is appraised high or it will be at base if different clients has evaluated low.

V. Acknowledgement

We acknowledge the IKGPTU, Kapurthala,Punjab for given us the opportunity for this research work.

References

[1] E. Adar, J. Teevan, S. T. Dumais, and J. L. Elsas, “The web changes everything: Understanding the dynamics of web content,”Proc. of the 2nd International Conference on Web Search and Data Mining, 2009.
[2] A. Agarwal, H. S. Koppula, K. P. Leela, K. P. Chitrapura, S. Garg, P. K. GM, C. Haty, A. Roy, and A. Sasturkar, “URL normalization for de-duplication of web pages,” Proc. of the 18th Conference on Information and Knowledge Management, 2009.
[3] C. C. Aggarwal, F. Al-Garawi, and P. S. Yu, “Intelligent crawling on the world wide web with arbitrary predicates,” Proc. of the 10th International World Wide Web Conference, 2001.
[4] D. Ahlers and S. Boll, “Adaptive geospatially focused crawling,” Proc. of the 18th Conference on Information and Knowledge Management, 2009.
[5] R. Baeza-Yates and C. Castillo, “Crawling the infinite web,” Journal of Web Architecture, vol. 6, no. 1, pp. 49–72, 2007.
[6] R. Baeza-Yates, C. Castillo, M. Marin, and A. Rodriguez, “Crawling a country: Better strategies than breadth-first for web page ordering,” Proc. of the 14th International World Wide Web Conference, 2005.
[7] Christo Ananth, M.Danya Priyadharshini, “A Secure Hash Message Authentication Code to avoid Certificate Revocation list Checking in Vehicular Adhoc networks”, International Journal of Applied Engineering Research (IJAER), Volume 10, Special Issue 2, 2015,(1250-1254)
[8] A. Heydon and M. Najork, “Mercator: A scalable, extensible web crawler,” World Wide Web, vol. 2, no. 4, pp. 219–229, 1999.
[9] M. Najork and A. Heydon, “High-performance web crawling,” Technical report, Compaq SRC Research Report 173, 2001.
[10] A. Broder, M. Najork, and J. Wiener, “Efficient URL caching for World Wide Web crawling,” Proc. of the 12th International World Wide Web Conference, 2003.

[11] D. Fetterly, M. Manasse, M. Najork, and J. L. Wiener, "A large-scale study of the evolution of web pages," Proc. of the 12th International World Wide Web Conference, 2003.

[12] M. Henzinger, A. Heydon, M. Mitzenmacher, and M. Najork, "Measuring index quality using random walks on the web," Proc. of the 8th International World Wide Web Conference, 1999.

[13] V. Shkapenyuk and T. Suel, "Design and implementation of a highperformance distributed web crawler," Proc. of the 18th International Conference on Data Architecture, 2002.

[14] J. Edwards, K. S. McCurley, and J. A. Tomlin, "An adaptive model for optimizing performance of an incremental web crawler," Proc. of the 10th International World Wide Web Conference, 2001.

[15] D. Gruhl, L. Chavet, D. Gibson, J. Meyer, P. Pattanayak, A. Tomkins, and J. Zien, "How to build a WebFountain: An architecture for very large-scale text analytics," IBM Systems Journal, vol. 43, no. 1, pp. 64–77, 2004.

[16] P. Boldi, B. Codenotti, M. Santini, and S. Vigna, "UbiCrawler: A scalable fully distributed web crawler," Software — Practice & Experience, vol. 34, no. 8, pp. 711–726, 2004.

[17] P. Boldi, B. Codenotti, M. Santini, and S. Vigna, "Structural properties of the African web," Proc. of the 11th International World Wide Web Conference, 2002.

[18] Yahoo! Research Barcelona, "Datasets for web spam detection," http://www.yr-bcn.es/webspam/datasets

[19] Infomine. UC Riverside library. http://lib-www.ucr.edu/, 2014.

[20] Clusty's searchable database dirctory. http://www.clusty. com/, 2009.

[21] Kevin Chen-Chuan Chang, Bin He, and Zhen Zhang. "Toward large scale integration: Building a metaquerier over databases on the web," In CIDR, pages 44–55, 2005.

[22] Denis Shestakov. "Databases on the web: national web domain survey,"Proc. of the 15th Symposium on International DatabaseArchitecture&Applications,pages179–184.ACM,2011.

[23] Denis Shestakov and TapioSalakoski. "Host-IP clustering technique for deep web characterization,"Proc. of the 12th International Asia-Pacific Web Conference (APWEB), pages 378–380. IEEE, 2010.

[24] Denis Shestakov and TapioSalakoski. "On estimating the scale of national deep web," In Database and Expert Systems Applications, pages 780–789. Springer, 2007.

[25] Shestakov Denis. "On building a search interface discovery system,"Proc. of the 2nd international conference on Resource discovery, pages 81–93, Lyon France, 2010. Springer.

[26] Luciano Barbosa and Juliana Freire. "Searching for hidden-web databases," In WebDB, pages 1–6, 2005.

[27] Luciano Barbosa and Juliana Freire. "An adaptive crawler for locating hidden-web entry points,"Proc. of the 16th international conference on World Wide Web, pages 441–450. ACM, 2007.

[28] SoumenChakrabarti, Martin Van den Berg, and Byron Dom. "Focused crawling: a new approach to topic-specific web resource discovery," Computer Networks, 31(11):1623–1640, 1999.

[29] Olston Christopher and Najork Marc. "Web crawling. Foundations and Trends in Information Retrieval," 4(3):175–246, 2010.

[30] Balakrishnan Raju and KambhampatiSubbarao. "Sourcerank: Relevance and trustassessment for deep web sourcesbased on inter-source agreement,"Proc. of the 20th international conference on World Wide Web, pages 227–236, 2011.

[31] Balakrishnan Raju, KambhampatiSubbarao, and JhaManishkumar. "Assessing relevance and trust of the deep web sources and results based on inter-source agreement," ACM Transactions on the Web, 7(2):Article 11, 1–32, 2013.

[32] Eduard C. Dragut, WeiyiMeng, and Clement Yu. "Deep Web Query Interface Understanding and Integration," Synthesis Lectures on Data Management. Morgan & Claypool Publishers, 2012.

[33] Jayant Madhavan, Shawn R. Jeffery, Shirley Cohen, Xin Dong, David Ko, Cong Yu, and Alon Halevy. "Web-scale data integration: You can only afford to pay as you go,"Proc. of CIDR, pages 342–350, 2007.

[34] Brightplanet's searchable database dirctory. http://www. completeplanet.com/, 2001.

[35] MohamamdrezaKhelghati, DjoerdHiemstra, and Maurice Van Keulen. "Deep web entity monitoring,"Proc. of the 22nd international conference on World Wide Web companion, pages 377–382. International World Wide Web Conferences Steering Committee 2013.

[36] SoumenChakrabarti, KunalPunera, and MallelaSubramanyam. "Accelerated focused crawling through onlinerelevancefeedback,"Proc.

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Special Issue 2, January 2018

Ofthe11th internationalconference on World Wide Web, pages 148–159, 2002.

[37] Mustafa EmmreDincturk, Guy vincent Jourdan, Gregor V. Bochmann, and IosifViorelOnut. “A model-based approach for crawling rich internet applications,” ACM Transactions on the Web, 8(3):Article 19, 1–39, 2014.

[38] Yeye He, Dong Xin, VenkateshGanti, SriramRajaraman, and Nirav Shah. “Crawling deep web entity pages,” In Proc. of the sixth ACM international conference on Web search and data mining, pages 355–364. ACM, 2013.

[39] Cheng Sheng, Nan Zhang, Yufei Tao, and Xin Jin. “Optimal algorithms for crawling a hidden database in the web,”Proc. of the VLDB Endowment, 5(11):1112–1123, 2012.

[40] Nilesh Dalvi, Ravi Kumar, Ashwin Machanavajjhala, and VibhorRastogi. “Sampling hidden objects using nearest-neighbor oracles,” In Proc. of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 1325– 1333. ACM, 2011.

[41] Thomas Kabisch, Eduard C. Dragut, Clement Yu, and Ulf Leser. “Deep web integration with visqi,”Proc. of the VLDB Endowment, 3(1-2):1613–1616, 2010.