MJCIS Vol. 15 No.1 Jun 2019

Mansoura Journal of Computers and Information Sciences

Web Crawler Architecture over Cloud Computing compared with Grid Computing

M. E. ElAraby
CS Dept. Faculty of Computers and information, Mansoura, Beni-Suef University
mohamed.elaraby@fcis.bsu.edu.eg

Sherihan M.
CS Dept. Faculty of Computers and information, Mansoura University
sherihan@mans.edu.eg

Hossam M. Moftah
CS Dept. Faculty of Computers and information, Beni-Suef University
hossam.moftah@fcis.bsu.edu.eg

M. Z. Rashad
CS Dept. Faculty of Computers and information, Mansoura University
mzrashad@mans.edu.eg

ABSTRACT

Web Crawler is considered as the core module of web search engines. It should be designed to cover high percent of Internet and adapt on scaling and in a distributed architecture. The crawler architecture has an effect on the quantity of fetched web pages in a determined time. Cloud computing is a type of computing paradigm that is characterized by a set of powerful points such as excitability, scalability, dynamism, and resource provisioning on demand, where these features are adding value in the crawler architecture. In this article, we propose an architecture for the web crawler that is designed over the cloud computing. The web crawler needs highly intensive computation, storage, and bandwidth. These resources can be provisioned by the cloud computing on demand with superior flexibility in changing as in the proposed architecture. We implemented and experimented the proposed architecture over cloud computing and evaluated the results of running. We also proposed another architecture based on grid computing to compare the results of the experiments over cloud computing with results over grid computing to evaluate the cloud-based architecture. Cloud computing has a higher performance than the grid computing. The proposed crawler over cloud computing exploited the features of cloud computing such as scalability, reliability, and flexibility through a well-defined service based architecture. Moreover, the results highlighted the enhancement in performance of the cloud-based architecture against the grid-based and monolithic.

Keywords

Web Crawler; Grid Computing; Cloud Computing; Architecture; Grid-based Crawler; Cloud-based Crawler.

1. INTRODUCTION

The World Wide Web and technologies are continuously growing, so this grown is useful for information search and retrieval on the Web. The data on the web are from several different sources [1]. The web is viewed as an information universe. It is considered as public source. Each resource of the web is associated with a URL. The size of the indexed documents from the internet contains more than 4.7 Billion indexed web documents in 25 September, 2016 [2].

Search engine typical design mainly consists of three stages are the web crawler that creates a collection of web pages, indexer that indexes the collection and searched. This search engine design is a sequence design. There are many schemes of web search engines that considered the crawler as the initial step in the search engine. The cascade model of the search engine, which operations are running in firm order crawling first, then indexing, and then searching [3]. There is a huge quantity of information that is available on the internet, so the web crawler architecture should be developed efficiently to download a large fraction of the Web [4]. Web crawler faces a group of hitches to fetch all documents from the internet which may not apply due to the characteristics of the web such as dynamism and continuous modifications. The web crawler needs stretch, dynamic and cooperative computational storage space. Moreover, it needs a high bandwidth of network to collect the most number of web pages [5].

The computing paradigm has been upgraded from the parallel computing toward distributed computing, grid computing, and then early to cloud computing. The distributed computing provides solutions for large scale problems. The distributed computing provides great pledge for using computer resources effectively [6].

Grid computing term appeared in the middle of 1990s, it means a suggested distributing computing structure for higher engineering and sciences [7, 8]. Overall, Grid Computing considered as a kind of distributed computing paradigm that relies on standalone computers connected to a network by Ethernet network interface.

Grid computing system contains one Master Node, a number of Executor Nodes and Storage Nodes. Master Node responsibility of connecting to other elements in the grid and using a system for Load Management to distribute activities over the executors. The storage Nodes are in concern with the storing of inputs and outputs of the data necessary for tasks [9]. Grid Computing is a software and hardware infrastructure

which provides pervasive, consistent, dependable, and low-cost admission to high-end computation resources [10].

The Cloud Computing is a fresh computing paradigm in Information Technology. The Cloud Computing permits suitable, on-demand and rapid network access a pooled collection of resources. These resources are configurable and can be rapidly provisioned. They are dynamically scalable and visualized platform as services, and provided in a geographically distributed [11]. Moreover, other features of Cloud Computing include fitness, elasticity, boundless capacity, availability, and boundless capacity are important. Many significant challenges are offered in Cloud Computing. Numerous researches concentrated on the technical problems which rose in providing and forming the Clouds and the effects on business and consumers [12].

Cloud Computing paradigm has a pioneer responsibility in the potential of World Wide Web services. There are numerous technology challenges in the Cloud community for fitting its vision to actuality. Moreover, the cloud computing has issues related to the management to introduce scalable and elastic (stretchy) service platforms built on demand and evolving Cloud combine technologies and architecture [13]. The issues that face migration of information systems to Cloud Computing are available, customization, integration and interoperability.

There are various system architectures that are used to develop software. The software architecture design is the system structure that involves the modules of software, the properties of visibility for those modules, and the associations between them [14]. Architecture of the software refers to the overall structure of the system software, managing complication and rising reuse, through the breakdown of a system into its high level subcomponents and their interconnections [15].

Software architecture differs based on the nature of computing paradigm that is used to deploy it where each computing paradigm has its nature and features that are different. If the software is deployed into a Cloud computing, then the software architecture should be described differently to utilize the Cloud features. Cloud features that should be utilized are reliable, availability, scalability, total cost of ownership and ease of deployment [16]. The software architecture over Cloud is the components and sub-components including security, layers, and the whole organization of the system on cloud computing [17].

If the same software is deployed into a grid computing, then the software architecture should be described differently to adapt the grid computing nature. Grid computing characteristics are that the distributed ownership of resources, each resource have its access policy, mechanism and cost. These characteristics in addition to architectures that are based on modular and component to enable portability ease of development, extensibility, and interoperability of independently developed components [18].

The motivation of this paper is to exploit the flexibility, scalability, geographical distribution and other capabilities of cloud computing to design a web crawling architecture outperforms any web crawler architecture on other computing paradigms. The paper contribution is the proposed web crawler architecture. The proposed web crawler is designed built on the micro-service and service oriented that can be deployed on cloud computing environment and utilized the capabilities of the cloud computing such as scalability and flexibility. Also, the power and fitness of the proposed web crawler gained from its architecture and cloud computing as an environment for running that has no limitation in its resources and scalable.

The paper is organized in this way. Section 2 is related work which introduces the related researches in the web crawling architecture. Section 3 is Proposed Web Crawler Architectures that is compose of two sub-sections one of them presents the proposed architecture using grid computing and the other presents the proposed architecture using cloud computing. Section 4 is Experimental results and analysis that are also consisted of two sub-sections one for grid-based crawler performance and the other for cloud-based crawler performance. Section 5 is the Conclusions and the Future work.

2. RELATED WORKS

The software architecture is the primary factor which has a direct effect to boost customization, availability, interoperability, integration and coverage of WWW. There are numerous researches in the topic of crawling specially design an architecture for the web crawling. From the time when the initiate web crawling system proposed by M. Gray in 1993, this crawling system was called “World Wide Web Wanderer”, to navigate the internet and gather the web sites the same as an automatic web crawl, spider, or agent [19].

The proposed web crawler overcome and outperform the previous web crawlers in its architecture which it is designed based on the service oriented and micro-services that can be deployed on cloud computing and utilized the capabilities of the cloud computing such as scalability and flexibility. The design of the previously proposed crawlers did not touch the service oriented and micro-services architecture as these architectures are novel methodologies in software development that are invited to adapt with cloud computing migration. Any proposed web crawler has a limitation in the expandability due to the limitation in the infrastructure that is prepared to run the crawler. But in the proposed web crawler in this paper, its power and fitness gained from its architecture and cloud computing as an environment for running that has no limitation in its resources and scalable.

Jonathan M. Hsieh [20] explained the design architecture, execution, and assessment of the extensible web crawling model. He stressed on designing a filter language and using document and filtering partitioning for scaling his model implementation. He stated that the system has scalable, low-latency, and high selectivity.

There are research groups that have a set of researches in the distributed computing field. There are research groups that created tools, libraries and middleware. They allowed the collaborative use of physically distributed resources combined for acting as one unit great platform for executing of distributed and parallel software. This computing approach has another name as Internet Computing, Meta-Computing,

Global Computing, Scalable Computing, and latterly as Grid Computing [8, 21, 22].

Data created in numerous sources in the World Wide Web are enormously growing. Finding related data without the announcement overload is still impossible with current technologies. Although some hopeful work in architecture that based on pushing to solve this problem, they drop to meet the need in the big data. Mehmet A. Akyol et al. proposed an architecture for context aware notification for people to discover needed data at its most valuable state [23].

Also, Web crawling topic has numerous research studies developed over grid computing. Some crawling architectures are designed based Grid Computing which its goal is to enhance the web crawling performance. K. Cerbioni has designed a focused crawling architecture on the grid computing, which this architecture designed for medical information [24]. This architecture provides adaptable services to retrieve medical information domain. Cerbioni's architecture is designed to handle the retrieval service for individuals which they are permitted to use the extremely distributed computational capability of the grid computing, and remove the central repository need. His architecture is presented in a three tier models, and highlighting the core of the system that is placed on the middle layer.

Also, M. Ben-Mubarak proposed a system for crawling based on a multi-agent system that improved the system efficiency [25]. He improved his system efficiency by utilizing a multiple agent system that based on crawling system. He proposed a search engine based on this crawler which this search engine is specifically for grid services. A. Guerriero et al. proposed a model of little cost web crawler for distributed environments and built on an effective URL allocation algorithm [26]. In their proposed architecture, the crawling modules are analyzed and have rules to be followed by the crawlers for maintaining the robustness and load balancing of the system as searching in the web. Their architecture is based on a grid computation paradigm and clustering.

J. Song et al. proposed an architecture to crawl the deep web called OGSA-DWC, which their architecture is designed for grid-based middleware [27]. Their middleware enable the developers to implement a system of a grid-based deep web crawling without needing to know how to use distributed computing resources. B. Barla Combazoglu et al. presented an architecture and implementation details of a search engine that is specifically for south Eastern Europe (SE4SEE) [28]. Their architecture is built on the grid computing.

Peter Mika proposed a part of a web crawler called a semantic web component over Cloud [29]. However, he didn't present the architecture design for the web crawling system. In 2015, Mehdi Bahrami proposed architecture design for the web crawling system that based on Cloud Computing [30]. Mehdi's architecture is based on the Map-Reduce model. His suggested web crawling architecture fetches web pages by distributed proxies. Every proxy keeps the fetched web documents in "Cloud Azure Table" and stores the enormous amount of formed and unformed data in "Azure Blob" storage.

3. PROPOSED WEB CRAWLER ARCHITECTURE

Web crawler consists of many operations that sending requests and receiving responses as web documents. Then, it analyses these web documents to obtain URLs from web documents and puts un-visited URLs in the Queue; every URL in the queue is fetched and so on. The goal is to gather many documents and gets URLs from the documents to gather a huge storage area of URLs in addition to documents on the web. The steps execute huge numbers of processes and spend a long time period.

It is important in describe the web crawling phases. Fig 1 show the phases of web crawling according to the native crawling. Native crawling consists of the next phases: Picking URLs "seed" as an first stage, storing the seed URLs in URL queue list using the "DNS Resolver" and cache, crawling the web documents by using the HTTP fetch from the Internet, putting the fetched document in the pages storage area, obtaining hyperlink of URLs from the crawled document, removing repeated URLs in the obtained URLs, cleaning the obtained URLs and storing the filtered URLs into "URL Queue".

Algorithm 1 list the steps of native crawling. The input of the native crawling is a set of URLs that are the seed URL for crawling. The steps of the algorithm are sequential and there is a set of repeated steps until the end of the URL queue or stopping the crawling process manual. The complexity of the native crawler is O (n^2), where n represents the number of URLs that will be fetched by the crawler.

Algorithm 1. Native Crawler Algorithm (Breadth First)

```
Input:
Seed URLs: URLs = {url_1, url_2, …,url_n}.
Outputs:
Set of Web Pages
Steps:
URL_Queue = URLs
URL_Visited = Ø
while URL_Queue is NOT Empty do
    URL ← Dequeue from URL_Queue
    P ← Web page of URL fetch from the web
    URL_Visited = URL_Visited ⋃ URL
    RP = RP ⋃ P
    Parse P to extract Extracted_URLs and Content
    for each url ∈ Extracted_URLs do
        if url ∉ URL_Queue AND url ∉ URL_Visited then
            URL_Queue =  URL_Queue ⋃ Extracted_URLs
        end if
    end for
end while
```

The next two sections are to show the proposed architecture for web crawler. The first section shows the architecture using the Grid Computing paradigm, and the second section shows the architecture using the Cloud Computing.

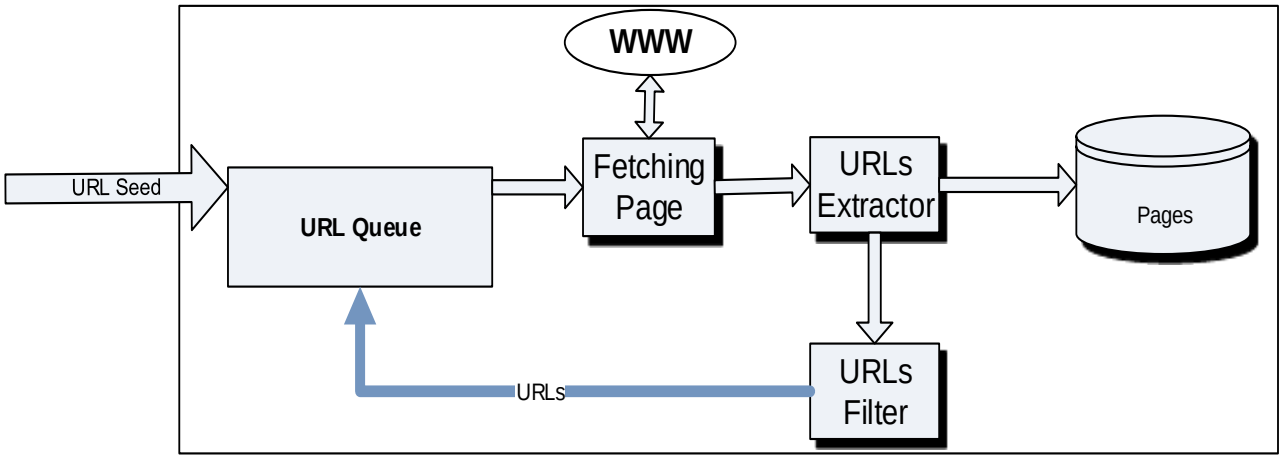

Fig 1: Web crawler stages

3.1 Proposed Architecture using Grid Computing

Grid Computing enables to utilize the remotely computational resources available in many computers that are regionally distributed. So, using Grid Computing in this proposed architecture suggested that distributing the crawler activities into multi-threads and using many separated PCs to run threads in balancing approach for distributing the work and minimizing the consumed periods in crawling. Fig 2 shows the diagram for the proposed web crawler architecture utilizing grid computing.

This proposed architecture is based on distributed and parallel processing. Its aim is to reduce the total period time for gathering the same quantity of pages, and effective use of the computational resource. The important classification of Grid Computing is computing grid and utility grid. Grid system requires powerful computing processing and could be distributed into sub-tasks that can do in parallel manner and merged to return the needed outcome. Branched tasks are accomplished in discrete computers to increase execution time efficiency and resource consumption. Utility Grid System is a set of computing resources that are virtual so that many users submit individual programs to this group. It gets the benefits of maximizing resources usage, workload balance, and best processing time for each application [31].

In the diagram of Fig 2 and Algorithm 2, the inputs are the seed URLs and define nodes that are connected in the grid as executors. The web crawler grid program produces a set of GThreads. This set dispatched to Alchemi Manager System [32] that spreads threads over access executor's nodes joined the manager, and the outcomes are restored to the manager system. Algorithm 2 has a function that is executed in the nodes which the grid managed passes the GThread with a URL to crawl it and extract URLs and return them to the manager to add the new URLs in the queue. The complexity of this algorithm is O (n (n/m)), where n is the number of crawled URLs and m is the number of nodes in the grid.

Algorithm 2. Crawling Algorithm on Grid Computing

Input:
Set of nodes in Grid: N = $\{node_0, node_1, \ldots, node_m\}$
Seed URLs: URLs = $\{url_1, url_2, \ldots, url_n\}$.
Outputs:
Set of Web Pages
Steps:
URL_Queue = URLs
URL_Visited = Ø
URL_Gthreads = Ø
c = 0
while URL_Queue ≠ Ø do
  URL ← Dequeue from URL_Queue
  i = c % length(N)
  c = c + 1
  URL_Gthreads = URL_Gthreads ∪ ($node_i$, G_{th}(URL))
  Extracted_URLs = $node_i$.assign(G_{th}(URL)), $node_i \in$N
  URL_Queue = URL_Queue ∪ Extracted_URLs
end while

function assign(Gthread G_{th}(url))
Start procedure
  p ← fetch (url)
  p: Web page of URL fetch from the web
  RP = RP ∪ p
  RP: repository of web pages
  URLs ← extractURLs(p)
  for each url ∈ URLs do
    if url ∉ URL_Queue AND url ∉ URL_Visited then
      URL_Return = URL_Return ∪ url
    end if
  end for
  return URL_Return
end function

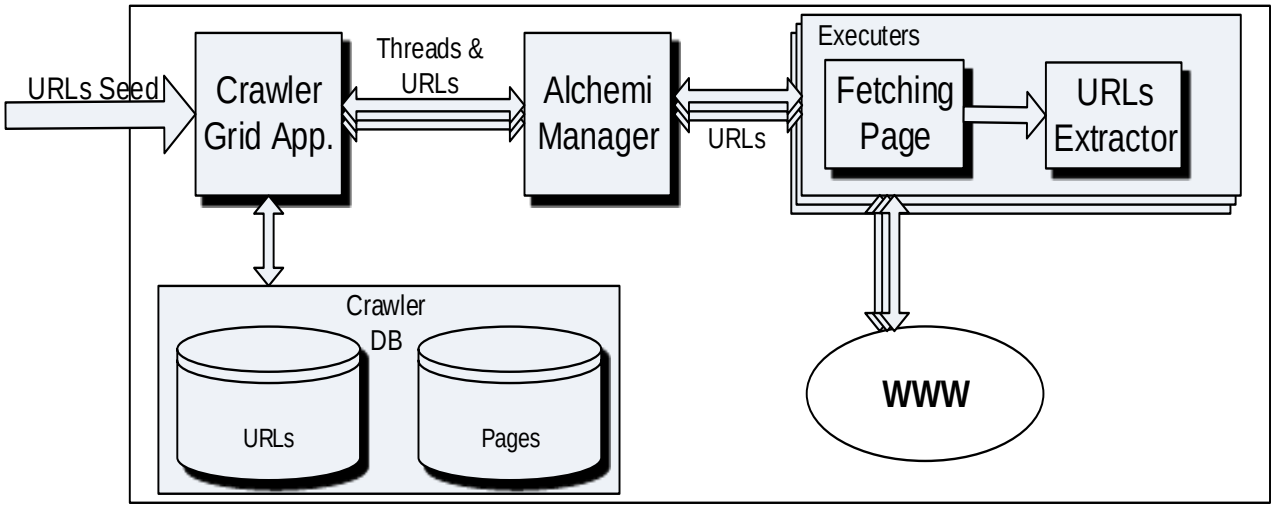

Fig 2: Proposed web crawler architecture on the Grid Computing

3.2 Proposed Architecture using Cloud Computing

In Fig 3, the diagram displays web crawler design over Cloud computing. The design is built on using virtual machines instantiation, and the web crawling processes are executing on one VM (virtual machine) instance. There is a master copy called Machine Image that prepared as a pattern for a full web crawler works independently. The page/document Storage and the URL Queue were designed as a central database for every crawling VM instances. The crawling VM instances are harmonized using the centralized database, and all crawler instances instantiated in the identical region in the cloud environment.

Because the geographical distribution for cloud computing, we proposed another architecture design as displayed in the diagram of Fig 4. It is alike of the previous design, although the VM instances are spread over various regions available in the cloud computing. This design gains the geographical distribution characteristic in the Cloud computing. Each region contains several web crawling VM instances and every region instances have a common document repository and URL queue in this region.

We proposed another architecture design for web crawler which create crawling instances in various regions, although the storage of document and URL queue are designed to be central in one region as displayed in the diagram of Fig 5 and Algorithm 3. This architecture design makes all web documents placed in one region for processing.

The storage used for the web document is “elastic” storage. It can be expanded and represented in various forms. It can be in “No-SQL (Non-Structured Query Language)” or “SQL (Structured Query Language)” form.

Algorithm 3 lists the steps of the crawling in each crawling instance, which the inputs are seed URLs and define the crawling instances (virtual machines). These steps are executed in each instance as it is, and the coordination between the virtual machines achieved through the shared queue and memory in the cloud computing. The complexity of this algorithm is O (n^2/l), where n represents the number of URLs that will be fetched by the crawler, and l represents the number of virtual machines.

By comparing the algorithm of the crawler over grid computing versus the proposed crawler over cloud computing, it is noticed that the time complexity of the first one is greater than the time complexity of the second. The crawler over grid computing divides the time of fetching and parsing but there is a centralized processing executed by the master node in the grid. But in the cloud computing, all processing is divided on all crawling instance as the crawling strategy is fully distributed without central control, so the proposed crawler over cloud achieves better enhancement.

Algorithm 3. Crawling Algorithm on Cloud Computing

Input:
Set of virtual machines: VM = $\{VM_0, VM_1, \ldots, VM_l\}$
Seed URLs: URLs = $\{url_1, url_2, \ldots, url_n\}$.
Outputs:
Set of Web Pages
Steps:
URL_Queue = URLs
URL_Visited = Ø
In each VM:
while URL_Queue Not empty do
    url ← Dequeue from URL_Queue
    p ← fetch (url)
    p: Web page of URL fetch from the web
    RP = RP ⋃ p
    RP: repository of web pages
    URLs ← extractURLs(p)
    for each url ∈ URLs do
        if url ∉ URL_Queue AND url ∉ URL_Visited then
            URL_Queue = URL_Queue ⋃ url
        end if
    end for
end while

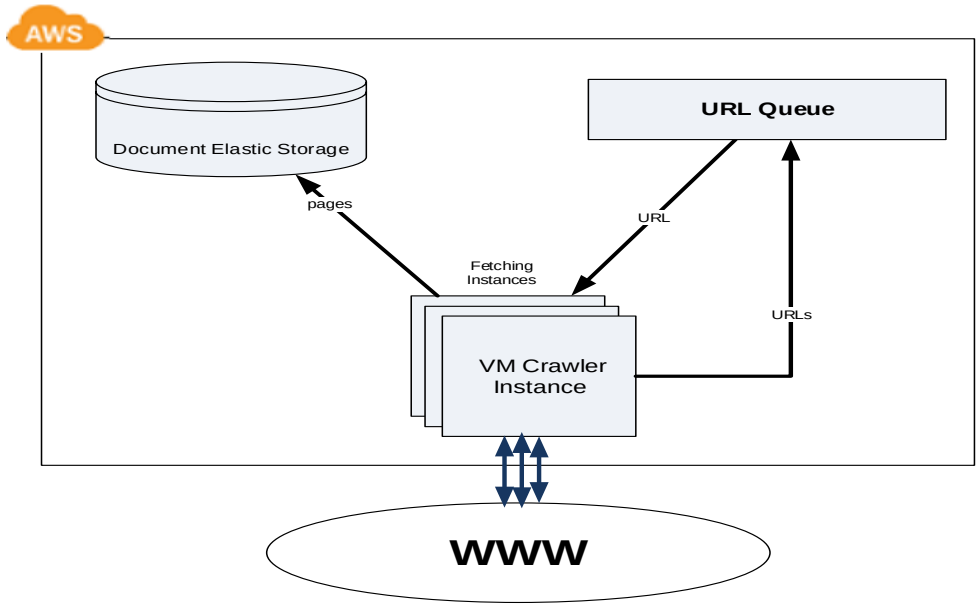

Fig 3: Multiple Crawling instances on the Cloud Computing

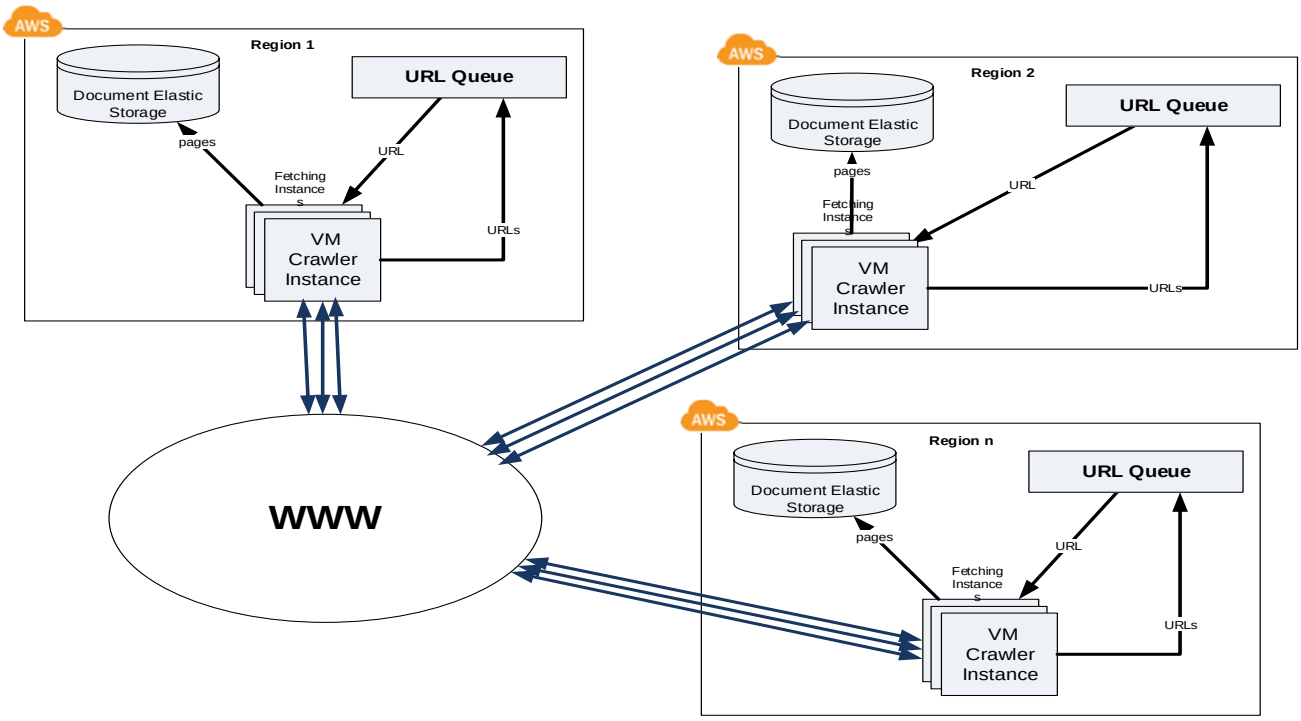

Fig 4: Geographical distributed crawling on the Cloud Computing regions

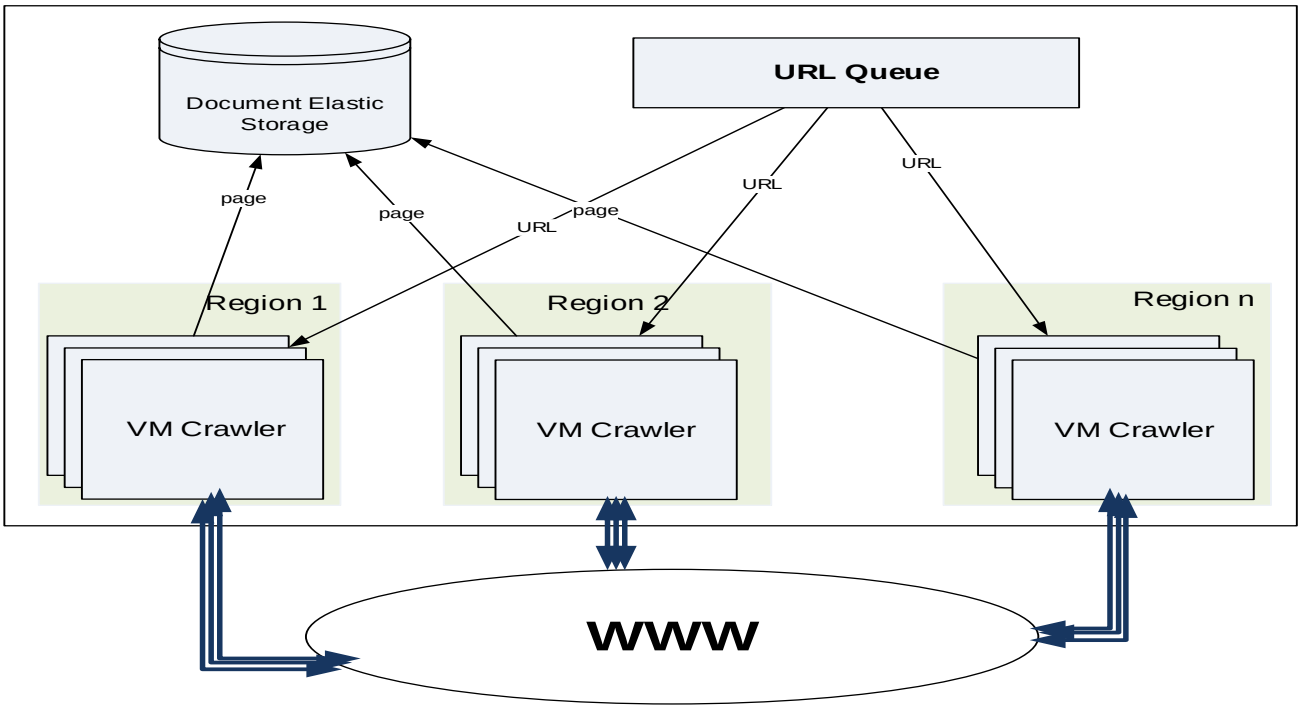

Fig 5: Geographical distributed crawling instances and centralized storage

4. EXPERIMENTAL RESULTS AND ANALYSIS

The experiments in this paper are initiated by selecting URL seed for the web crawling process in Grid-based and Cloud-based architecture, where the selected seed is the URL of "MIT Education" web site http://www.mit.edu.

The aspects influences the performance of any web crawler are the quantity of crawled web documents, the number of collected hyperlinks to others web documents, the number of accessed host servers, the response time for documents fetching from the host server, and volume of transferred/fetched web documents. So, we will quantify all these aspects only in this proposed crawling architecture over Cloud computing. The consumed time is measured for web crawling architecture over Grid computing and Cloud computing for comparison between their performances. A definite amount of documents in various cases were executed to crawl 50, 100, 150, 200, 250, and 300. Their consumed time in the crawling was assessed using the unit of second.

In the Cloud-based architecture, the number of recent URLs obtained from fetched documents, and number of host servers called are measured. Accessing same host server twice in a tiny time period will affect the response time of the crawl, because the servers keep client address IP. If the host server received multiple http request in a small period, then the server will stay http response for some time. This response delay influences the calculated crawl time. Also, response time needs to be measured in second. Because the size of document data influences the crawling rate of the web crawling, the pages' sizes downloaded were measured in "KiloBytes" of all crawled web documents. Evaluation was made in six occasions of running 50, 100, 150, 200, 250 and 300 pages.

4.1 Grid-Based Crawler Performance

In the Grid-based architecture, the experiment done on an environment consists of five personal computers connected with an internet ADSL line with speed 1 Mbps in an Ethernet LAN network its speed 10 Mbps. Each computer contains a 1 GB Random Access Memory and Intel processor 2.3 GHz. This environment to be a Grid computing needs a Grid computing framework to organize the computing resources, so we use Alchemi framework. Alchemi framework is an open source software framework that lets to collect the processing power of connected computers into a simulated super-computer and to develop software applications to execute on the Grid computing [32]. Alchemi components are Alchemi Executor and Alchemi Manager. Alchemi Executor is a program that is installed on all computers to be executor nodes. Microsoft .NET framework 1.1 should be installed on all executors before install Alchemi executor program. Alchemi Manager is a program that is installed on one computer to control all execution nodes connected to it. SQL Server is installed on Alchemi manager node to store the identifier numbers and details of executors, and to store identifier numbers and threads status (running/completed/failed) mapped with the executor id.

The web crawler application is running on Alchemi manager node. Numerous cases have been conducted on the executor nodes and fetched 50, 100, 150, 200, 250, and 300 web documents. As showed in Fig 6, by raising the executors in the grid, the time value reduces in a linear mode.

The experiments using grid computing test bed composed of parameters the first one is the number of nodes in the grid computing, and the second one is the number of pages that should be crawled in the grid. As in Fig 6, the horizontal axis is the number of crawled pages, the colored lines each one represents a case of running the experiment with a specific number of nodes executors, and the vertical axis represents the time spent to complete the crawling processes. The experimental results show that the time of each case is decreased by increasing the number of nodes in the grid. By increasing the number of nodes in the grid the time enhancement decreases. In case of 300 pages the time of crawling in 1, 2, 3, 4 and 5 are 2167.2, 1080.6, 810, 533.4 and 434.4 respectively. The decreasing rate in these experiments are also decreased the decreasing rate of 1 and 2 nodes are 2.01, 2 and 3 is 1.334, 3 and 4 are 1.5186, and 4 and 5 are 1.22. The decreasing rate decrease till become one that means there is no enhancement in the time.

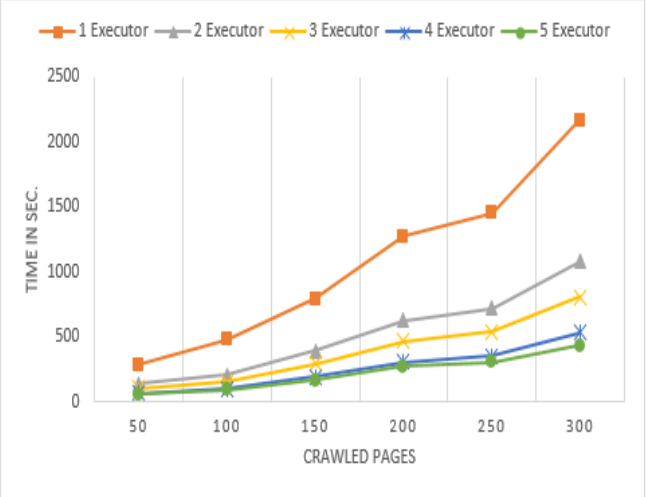

Fig 6: Running the proposed web crawler over Grid Computing

4.2 Cloud-Based Crawler Performance

In the Cloud-based architecture, multiple crawler instances are running in a distinct virtual machine created on Cloud computing. Amazon EC2 is used which it provides a broad collection of instance classes [33]. Instance "t2.micro" is selected for instantiation from Amazon EC2. This instance is comprised of an "Intel Xeon" family CPU (its speed up to 3.3 GHz). "Intel Turbo" one GB memory and "Amazon EBS Storage". The network bandwidth in this instance is small to average. We selected US East (N. Virginia) as the running region while instantiation and the internet speed through this experiments were 130Mbps to 200 Mbps.

The used database was existed on "Amazon Relational Database" Service (Amazon "RDS"). This service is simple to expand the database and makes for emphasis on the business and applications [33]. Database was "MySQL DB" instance,

which it was connected remotely by "MySQL Query Browser". Database instance was "MySQL DB" on "db.t2.micro" instantiated in that region of the crawlers in "US East - N. Virginia".

This experiment is comprised of two parameters as in the grid-based which are number of VM instances that are running rather than executor nodes, and number of fetched web documents.

In Fig 7, the results of executing these experiments show the outcomes of executing on multi-VMs (multiple virtual machines) on "Amazon Cloud Computing". This experiment is conducted five times, in stages from one instance to five instances.

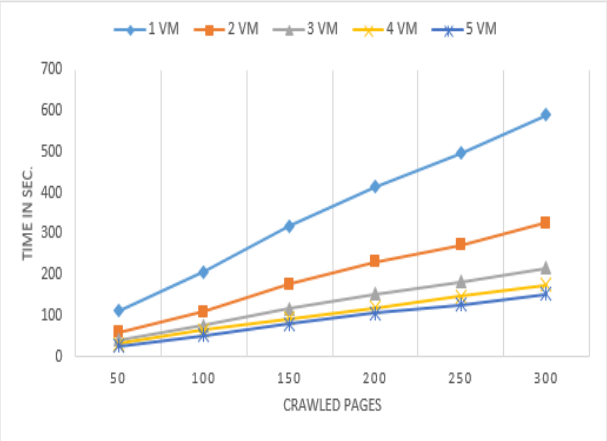

Fig 7: Proposed Crawler on Amazon Cloud Computing

The experiments using cloud computing test bed composed of parameters the first one is the number of the virtual machine in the platform cloud computing, and the second one is the number of pages that should be crawled in the cloud. As in Fig 7, the horizontal axis is the number of crawled pages, the colored lines each one represents a case of running the experiment with a specific number of nodes from executors, and the vertical axis represents the time spent to complete the crawling processes. The experimental results show that the time of each case was reduced by rising number of the virtual machine in the cloud. By increasing the number of instances in cloud the time enhancement decreases. In case of 300 pages the time of crawling in 1, 2, 3, 4 and 5 is 588.957, 326.194, 216.759, 176.699 and 153.045 respectively. The decreasing rate in these experiments is also decreased the decreasing rate of 1 and 2 nodes are 1.805, 2 and 3 is 1.5048, 3 and 4 is 1.2267, and 4 and 5 is 1.15456. The decreasing rate decrease till become one that means there is no enhancement in the time.

We calculated the average of the crawling time for each case in the grid-based crawler and the Cloud-based crawler, and created a comparison of average crawling time in the two paradigms. There is a clear difference between the time of the grid-based crawler and the Cloud-based crawler as showed in the graph of Fig 8, where the Cloud-based crawler gives a better performance through less time than the grid-based crawler time.

The graph in Fig 9 represents the statistics of URLs gathered in cases of crawl which the URLs detected in each one were nearer together, because number of crawled documents were fixed.

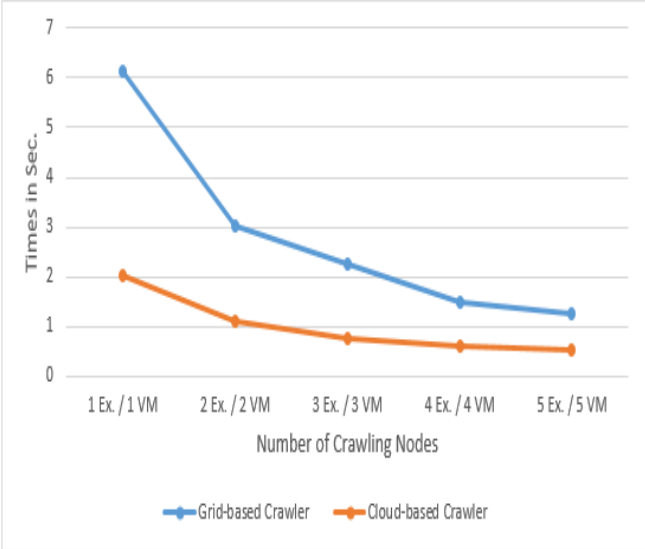

Fig 8: Averages of crawling times in the two proposed web crawlers

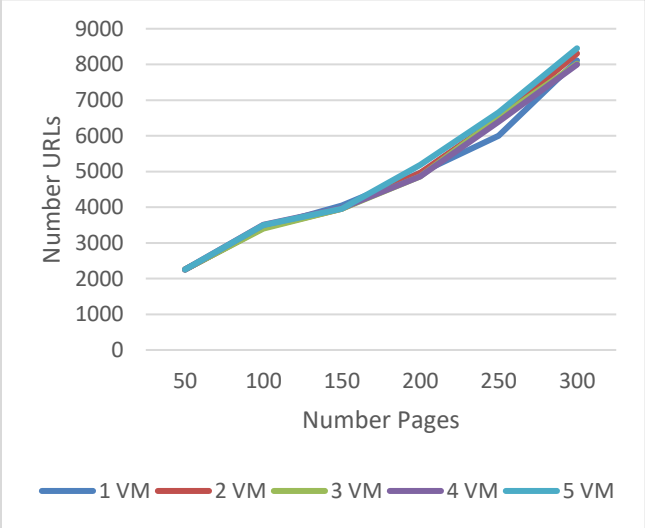

Fig 9: Number of detected URLs using Cloud - based Crawlers

Also we measured the number of domains that were accessed in every case of crawl. The numbers of domains in each case are nearer together, because numbers of web document in all cases were fixed. Although the small difference is explained by the reason mentioned before. The graph displayed in Fig 10 represents the domains accessed using the Cloud-based crawlers.

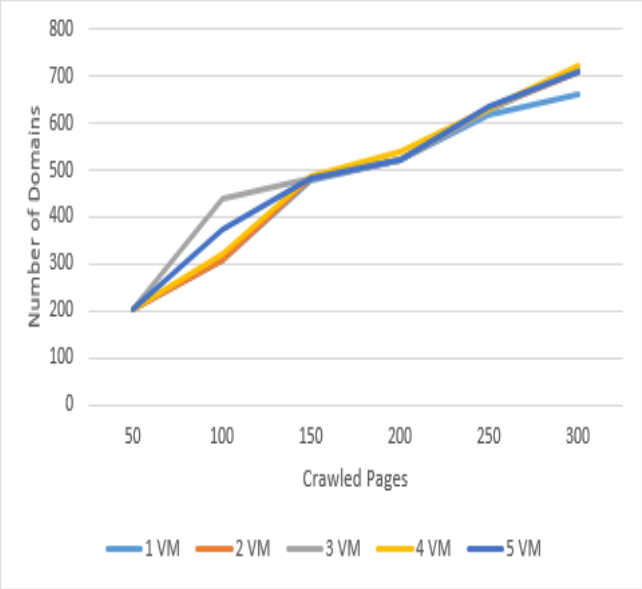

Fig 10: Statistics of accessed web-domains using Cloud-based Crawlers

We measure the size of gathered web documents in all cases of crawling. Also these measured sizes are closer in each case, due to the fixed number of web documents in every case, and the small variances due to the reasons mentioned previously. The graph in Fig 11 represents the sizes of crawled pages in the Cloud-based crawlers.

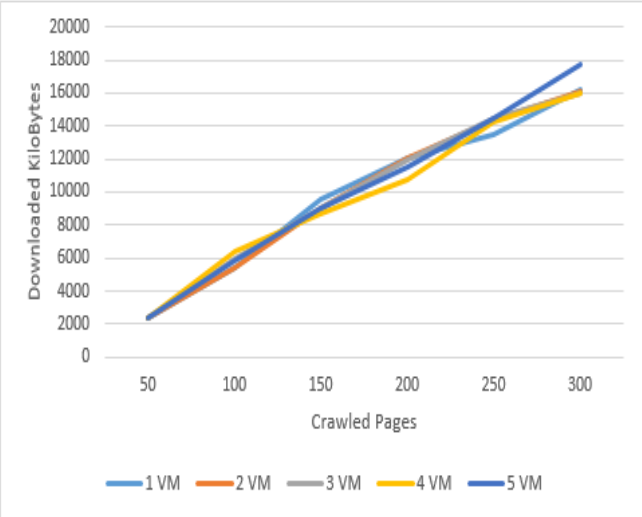

Fig 11: Sizes of fetched pages in Cloud-based Crawlers

The response time is the consumed time between the start of transmitting http request and receive the http response for the requested web page [34]. Two factors affect the response time. The first one is the network delay and the second is the server side latency. Latency time of server side is the time took to produce the http response since the request comes to the server [35]. Therefore the response time is measured in cases Cloud-based crawlers.

The average of the response time is computed for each case in the grid-based crawler and the Cloud-based crawler. The comparison between response time in grid-based crawler and Cloud-based crawler is shown in Fig 12. As showed in the graph of Fig 12, there is a clear difference between the response time of the grid-based crawler and the Cloud-based crawler. This also insures that the Cloud-based crawler is better than the Grid-based crawler.

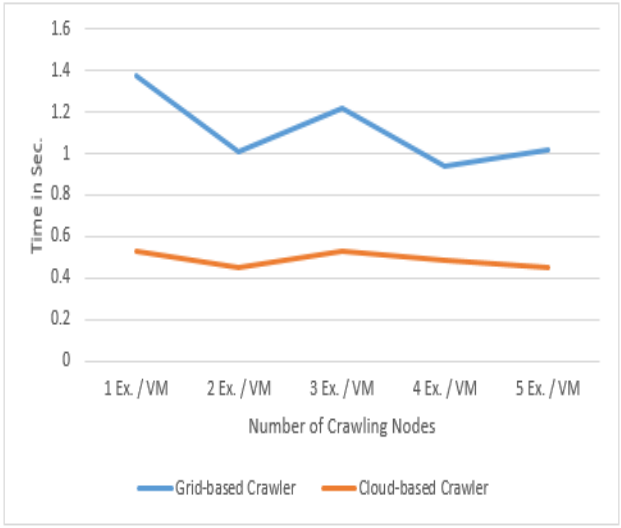

Fig 12: Average of response time in Grid-based and Cloud-based crawlers

In the grid-based crawler, crawling time is categorized into two time, the processing time in the grid manager and the processing time in the grid nodes. So, crawling time in the grid-based crawler is enhanced by dividing time of the grid node by number of nodes in the grid which these times are the fetching, parsing, extracting URLs time. But in the cloud-based crawler, the time of crawling at all is divided by the number of crawling instances in the cloud. So, the cloud-based crawler has a better time enhancement than grid-based crawler, and this proved by the crawling time in Fig 8 and the response time in Fig 12.

5. CONCLUSIONS AND FUTURE WORK

This paper introduced an architecture for the crawling which this architecture is based on cloud computing and its features such as extendibility, scalability, dynamism, and resource provisioning on demand. To show the power of the cloud computing, we proposed another architecture for the web crawler over grid computing as another type of computing paradigm. Grid computing and cloud computing are two paradigms of distributed computing, but each one has its features that affects the performance of the web crawler. Therefore, two architectures design are different where each one tries to adapt web crawler to environment nature. In this article, we showed that cloud-based architecture performance is superior to grid-based architecture through measurements for the time of crawling and the response time for each one.

Our future work will be presenting the services to be configured in web crawling for example focused crawler to crawl a certain region, a restricted field, or a definite language. The service descriptions can be given as input constraints to this service.

6. REFERENCES

[1] M. Kobayashi and K. Takeda. “Information retrieval on the web”, Journal ACM Computing Surveys, Volume 32 Issue 2, Pages 144-173, June 2000.

[2] M. d. Kunder. "Daily Estimated Size of the World Wide Web," 26 September 2016. [Online]. Available: http://www.worldwidewebsize.com/. [Accessed 26 September 2016].

[3] C. D. Manning, P. Raghavan, and H. Schutze. Book: “Introduction to Information Retrieval”, CAMBRIDGE UNIVERSITY PRESS, 2009.

[4] J. Arguello, F. Diaz, J. Callan and J. F. Crespo. Sources of evidence for vertical selection. In: 32nd International conference on Research and development in Information Retrieval, SIGIR’09 pp. 315--322, ACM, New York, USA, 2009.

[5] M. E. ElAraby, M. M. Sakre, M. Z. Rashad and O. Nomir. "Crawler Architecture using Grid Computing," International Journal of Computer Science & Information Technology, vol. 4, no. 3, pp. 113-127, 2012.

[6] B. Kahanwal and T. P. Singh. “The Distributed Computing Paradigms: P2P, Grid, Cluster, Cloud, and Jungle”, International Journal of Latest Research in Science and Technology, Vol. 1, No. 2, pp. 183-187, 2012.

[7] I. Foster, C. Kesselman, J. M. Nick and S. Tuecke. “The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration”, Global Grid Forum, June 22, 2002.

[8] F. Berman, G. Fox and T. Hey. Book: “Grid Computing: Making the Global Infrastructure a Reality”, Vol. 2. John Wiley and sons, March 2003.

[9] D. Garlasu, V. Sandulescu, I. Halcu, G. Neculoiu, O. Grigoriu, M. Marinescu and V. Marinescu. “A big data implementation based on Grid computing”, Roedunet International Conference (RoEduNet), 17-19 Jan. 2013.

[10] I. Foster and C. Kesselman. The Grid2: Blueprint for a New Computing Infrastructure. Morgan Kaufmann, 1999.

[11] D. Leaf. "The NIST Cloud Computing Project," 2 February 2011. [Online]. Available: http://www.nist.gov/itl/csd/Cloud-020111.cfm. [Accessed 4 March 2016].

[12] P. Sasikala. "Research challenges and potential green technological applications in Cloud," Int. J. Cloud Computing, vol. 2, no. 1, pp. 1-19, 2013.

[13] R. Moreno-Vozmediano, R. S. Montero and I. M. Llo. "Key Challenges in Cloud Computing: Enabling the Future Internet of Services," IEEE Internet Computing, vol. 17, no. 4, pp. 18-25, 2013.

[14] Bass, Clements, and Kazman. “Software Architecture in Practice”, Addison-Wesley 1997.

[15] A. Bertolino, P. Inverardi and H. Muccini. “Software architecture-based analysis and testing: a look into achievements and future challenges”, Computing, Springer, Volume 95, Issue 8, pp 633–648, August 2013.

[16] C. Chapman, W. Emmerich, and F. G. Márquez. “Software architecture definition for on-demand Cloud provisioning”, Cluster Computing, Volume 15, Issue 2, pp 79–100, June 2012.

[17] M. Bahrami and M. Singhal. "DCCSOA: A Dynamic Cloud Computing Service-Oriented," in Proceedings of 16th IEEE International Conference on Information Reuse and Integration, San Francisco, USA, 2015.

[18] H. D. Mustafa, S. N. Merchant, U. B. Desai and B. M. Baveja. Introduction. In: Green Symbiotic Cloud Communications, pp. 1-9. Springer, Singapore, 2017.

[19] J. Edwards, K. McCurley and J. Tomlin, "An adaptive model for optimizing performance of an incremental web crawler", Proceedings of the 10th international conference on World, 2001.

[20] J. M. Hsieh, S. D. Gribble and H. M. Levy. "The Architecture and Implementation of an ExtensibleWeb Crawler", in NSDI'10 (the 7th USENIX conference on Networked systems design and implementation), CA, USA, 2010.

[21] I. Foster, C. Kesselman and S. Tuecke. The anatomy of the grid: Enabling scalable virtual organizations, International Journal of High Performance Computing Applications Fall 2001 15: pp. 200-222, 2001.

[22] M. Minsky and S. A. Papert. “Perceptrons: An introduction to computational geometry”. MIT press, 2017.

[23] M. A. Akyol, M. O. Gökalp, K. Kayabay, P. E. Eren and A. Koçyiğit. “A Context Aware Notification Architecture Based on Distributed Focused Crawling in the Big Data Era”, In European, Mediterranean, and Middle Eastern Conference on Information Systems (pp. 29-39). Springer, Cham, September, 2017.

[24] K. Cerbioni, E. Palanca, A. Starita, F. Costa and P. Frasconi. “A GRID FOCUSED COMMUNITY CRAWLING ARCHITECTURE FOR MEDICAL INFORMATION RETRIEVAL SERVICES”, 2nd Int. Conf. on Conf. on Computational Intelligence in Medicine and Healthcare, CIMED, 2005.

[25] M. Ben-Mubarak; A. Hasni and I. Chai. "Multi Agent System-based crawlers for Virtual Organizations ", IEEE International Conference of Distributed Frameworks for Multimedia Applications, May 2006.

[26] A. Guerriero, F. Ragni and C. Martines. "A dynamic URL assignment method for parallel web crawler", IEEE International Conference of CIMSA, on pages 119 – 123, Sept. 2010.

[27] J. Song, D. Choi and Y. Lee. " OGSA-DWC: A Middleware for Deep Web Crawling Using the Grid ", IEEE Fourth International Conference of eScience, on pages 370 - 371, Dec. 2008.

[28] B. B. Cambazoglu, E. Karaca, T. Kucukyilmaz, A. Turk and C. Aykanat. " Architecture of a grid-enabled Web search engine", Available online 11 December 2006, Information Processing and Management number 43 on pages 609–623, 2007.

[29] P. Mika and G. Tummarello. "Web Semantics in the Clouds", IEEE Intelligent Systems, vol. 23, no. 5, pp. 82 - 87, Oct. 2008.

[30] M. Bahrami, M. Singhal and Z. Zhuang. "A Cloud-based web crawler architecture", 18th International Conference in Intelligence in Next Generation Networks (ICIN), Paris, Feb. 2015.

[31] N. Krishnadas. “Designing a Grid Computing Architecture: A Case Study of Green Computing Implementation Using SAS®”, SAS Global Forum 2011 Systems Architecture and Administration, Indian Institute of Management, Kozhikode, Kerala, India, Paper 366-2011.

[32] A. Luther, R. Buyya, R. Ranjan and S. Venugopal. “Alchemi: A. NET-based Enterprise Grid Computing System”. In International Conference on Internet Computing, pp. 269-278, June 2005.

[33] "AWS Amazon", [Online]. Available: https://console.aws.amazon.com/. [Accessed 3 October 2016].

[34] N. Ye, X. Li, T. Farley and X. Xu. "Job scheduling methods for reducing waiting time variance", Computers & Operations Research, Elsevier Ltd., vol. 34, no. 10, pp. 3069–3083, October 2007.

[35] R. Rajamony and M. Elnozahy. "Measuring Client-Perceived Response Times on the WWW", USENIX Symposium on Internet Technologies and Systems (USITS), vol. 3, March 2001.