Online ISSN : 0975-4172
Print ISSN : 0975-4350
DOI : 10.17406/GJCST

Global Journal

of Computer Science and Technology: B

Cloud & Distributed

Analytic Real-Time Framework

Homomorphic Encryption Security

Highlights

Security for Cloud Computing

Clusters and Big Data Frameworks

Discovering Thoughts, Inventing Future

VOLUME 19 ISSUE 1 VERSION 1.0

Global Journal of Computer Science and Technology: B
Cloud & Distributed

Global Journal of Computer Science and Technology: B
Cloud & Distributed

Volume 19 Issue 1 (Ver. 1.0)

Open Association of Research Society

Incorporation No.: 0423089
License No.: 42125/022010/1186
Registration No.: 430374
Import-Export Code: 1109007027
Employer Identification Number (EIN):
USA Tax ID: 98-0673427

Global Journals Inc.

*(A Delaware USA Incorporation with "Good Standing"; **Reg. Number: 0423089**)*

Sponsors: Open Association of Research Society
Open Scientific Standards

Publisher's Headquarters office

Global Journals® Headquarters
945th Concord Streets,
Framingham Massachusetts Pin: 01701,
United States of America
USA Toll Free: +001-888-839-7392
USA Toll Free Fax: +001-888-839-7392

Offset Typesetting

Global Journals Incorporated
2nd, Lansdowne, Lansdowne Rd., Croydon-Surrey,
Pin: CR9 2ER, United Kingdom

Packaging & Continental Dispatching

Global Journals Pvt Ltd
E-3130 Sudama Nagar, Near Gopur Square,
Indore, M.P., Pin:452009, India

Find a correspondence nodal officer near you

To find nodal officer of your country, please email us at *local@globaljournals.org*

eContacts

Press Inquiries: *press@globaljournals.org*
Investor Inquiries: *investors@globaljournals.org*
Technical Support: *technology@globaljournals.org*
Media & Releases: *media@globaljournals.org*

Pricing (Excluding Air Parcel Charges):

Yearly Subscription (Personal & Institutional)
250 USD (B/W) & 350 USD (Color)

Editorial Board

Global Journal of Computer Science and Technology

Dr. Abdurrahman Arslanyilmaz

Computer Science & Information Systems Department
Youngstown State University
Ph.D., Texas A&M University
University of Missouri, Columbia
Gazi University, Turkey
Web: cis.ysu.edu/~aarslanyilmaz/professional_web

Dr. Chutisant Kerdvibulvech

Dept. of Inf. & Commun. Technol.,
Rangsit University
Pathum Thani, Thailand
Chulalongkorn University Ph.D. Thailand
Keio University, Tokyo, Japan

Dr. Sukhvinder Singh Deora

Ph.D., (Network Security), MSc (Mathematics),

Masters in Computer Applications

Dr. Sotiris Kotsiantis

Ph.D. in Computer Science, University of Patras, Greece
Department of Mathematics, University of Patras, Greece

Dr. Ramadan Elaiess

Ph.D.,

Computer and Information Science

Dr. Manpreet Singh

Ph.D.,
(Computer Science)

Nicla Romano

Professor in Cellular and Developmental Biology;
Cytology and Histology; Morfogenesis and Comparative Anatomy

Dr. Muhammad Abid

M.Phil,
Ph.D Thesis submitted and waiting for defense

Dr. K. Venkata Subba Reddy

Ph.D in Computer Science and Engineering

Loc Nguyen

Postdoctoral degree in Computer Science

Faisal Mubuke

M.Sc (IT), Bachelor of Business Computing, Diploma in Finanicial services and Business Computing

Jiayi Liu

Physics, Machine Learning,
Big Data Systems

Dr. Yuanyang Zhang

Ph.D in Computer Science

Asim Gokhan Yetgin

Design, Modelling and Simulation of Electrical Machinery;
Finite Element Method, Energy Saving, Optimization

Anup Badhe

Bachelor of Engineering (Computer Science)

Dr. S. Nagaprasad

M.Sc, M. Tech, Ph.D

Contents of the Issue

i. Copyright Notice
ii. Editorial Board Members
iii. Chief Author and Dean
iv. Contents of the Issue

1. Cloud-based Architecture of Raspberry Pi: Personal Cloud Storage. *1-7*
2. Homomorphic Encryption Security for Cloud Computing. *9-11*
3. An Analytic Real-Time Framework for IoT Based Home Automation System. *13-24*
4. A Taxonomy of Schedulers – Operating Systems, Clusters and Big Data Frameworks. *25-40*
5. Measuring the Performance on Load Balancing Algorithms. *41-49*

v. Fellows
vi. Auxiliary Memberships
vii. Preferred Author Guidelines
viii. Index

Global Journal of Computer Science and Technology: B
Cloud and Distributed
Volume 19 Issue 1 Version 1.0 Year 2019
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals
Online ISSN: 0975-4172 | Print ISSN: 0975-4350 | DOI: 10.17406

Cloud-based Architecture of Raspberry Pi: Personal Cloud Storage

By Faisal Khalil-Ur-Rehman & Muhammad Farooq
Limkokwing University of Creative Technology

Abstract- The research explained the reason why we need personal cloud storage. This research will show steps on how to build a personal cloud storage by using credit card size Raspberry Pi (minicomputer), which will help the user to enable cloud storage mode to their external hard drive. However, other cloud storage services like Dropbox, Google Drive, and iCloud gives limited amount of storage. This research will help the users to use (1TB) or above size external hard drive to be use and have access anywhere from any device over internet. Also the second part of this research focus on replace the laptops to raspberry pi that lecturers use in the classroom to play PowerPoint slides, and videos at university.

Keywords: *raspberry Pi, cloud storage, cost benefit-analysis, low cost-computing, university classrooms.*

GJCST-B Classification : *H.3.m*

Strictly as per the compliance and regulations of:

Cloud-based Architecture of Raspberry Pi: Personal Cloud Storage

Faisal Khalil-Ur-Rehman[α] & Muhammad Farooq[σ]

Abstract- The research explained the reason why we need personal cloud storage. This research will show steps on how to build a personal cloud storage by using credit card size Raspberry Pi (minicomputer), which will help the user to enable cloud storage mode to their external hard drive. However, other cloud storage services like Dropbox, Google Drive, and iCloud gives limited amount of storage. This research will help the users to use (1TB) or above size external hard drive to be use and have access anywhere from any device over internet. Also the second part of this research focus on replace the laptops to raspberry pi that lecturers use in the classroom to play PowerPoint slides, and videos at university.

Universities use laptops to plug and play their educational slides and videos. All these laptops price and maintenance cost lot to the university, if we look deeply just for play slides we do not have to buy a laptop which cost $300 and also the lecturer have to carry the laptops all the times from the faculty to classes, moreover most of the times the laptops are not available. Overcome above statement, all the laptops can be replaced to "Raspberry Pi" which cost $35 and it does not need any maintenance.

Keywords: raspberry Pi, cloud storage, cost benefit-analysis, low cost-computing, university classrooms.

I. Introduction

The aim of this research is to develop a cloud computing project, where the users can use their external hard drive's connected to Raspberry Pi through internet they can have access to anywhere from any device. According to (Jon Brodkin, 2008) Even though Security, Privacy and Trust issues exists since the evolution of Internet, the reason why they are widely spoken these days is because of the Cloud Computing scenario. Any client/small organization/enterprise that processes data in the cloud is subjected to an inherent level of risk because outsourced services bypass the "physical, logical and personnel controls" of the user.

To develop a cloud computing project where users can have large amount of storage with the help of Raspberry Pi. Most of the user have external hard drive 1 TB (Terabyte) or more but the users cannot carry the external hard drives all the time, whereby this project will benefit the users connect to personal cloud storage over internet. The new technology which will be affordable to everyone and also it will enable the user to use their external hard drives over internet possible.

Author α: Postgraduate Centre, Limkokwing University of Creative Technology, Malaysia. e-mail: faisal.albalushi22@gmail.com

Laptops that the lecturers are using for slides provided by Limkokwing University are limited in quantity and most of them are not working (requires maintenance). Replace all the laptops with the Raspberry Pi, which only cost $35 and does not require maintenance (low maintenance). The laptop cost is above $300 and need maintenance but the Raspberry Pi cost is $35 and does not need any maintenance. Raspberry Pi will reduce the cost and effort for lecturers, also it is easy to carry.

Most of the cloud computing services are providing limited storage to the users, at the same time the risk of your data to store at third party is a big issue. Example is the recent incident of exposed nude pictures of the celebrities from iCloud. Which is a clear example that our data is not safe, to overcome these problems this research will come up with the perfect solution with unlimited cloud data storage by the help of "Raspberry Pi".

Rather than store all our media and files on a cloud server in an unknown location, we can keep a cloud at our home and make it personal.

The objectives for this research are

- The primary object of this research is to develop a cloud storage, whereby all the users can connect their external hard drives to raspberry pi and have access everywhere over internet.
- To develop a raspberry pi which can connect to projector and run the slides, videos and also web browsing as well.
- To save the cost for the university by replacing the laptops to raspberry pi, whereby the university can save money and maintenance time as well. Also it will be easy for the lecturers to carry the raspberry pi to classes rather than laptops. The university can save roughly up to RM 200, 000.
- To compare the current cloud computing to the Pi cloud computing. In term of storage because all the cloud computing services provide only limited number of storage to the users or the users have to buy extra storage if they require more space.
- To help the lecturers retrieve files (chapter's slides) from Pi cloud in the classroom. By using raspberry pi in the classrooms will represent the student's creativity and the education level at Limkokwing University.

II. Literature Review

According to William (2014), in order to do effective way of computation, Cloud Computing offers IaaS, PaaS and SaaS levels of service models. The lowest service model is called Infrastructure as a Service (IaaS), which follows by Platform as a Service (PaaS) and last, but not least Software as a Service (SaaS). Each service model helps to add more functionality and abstraction to the technical details.

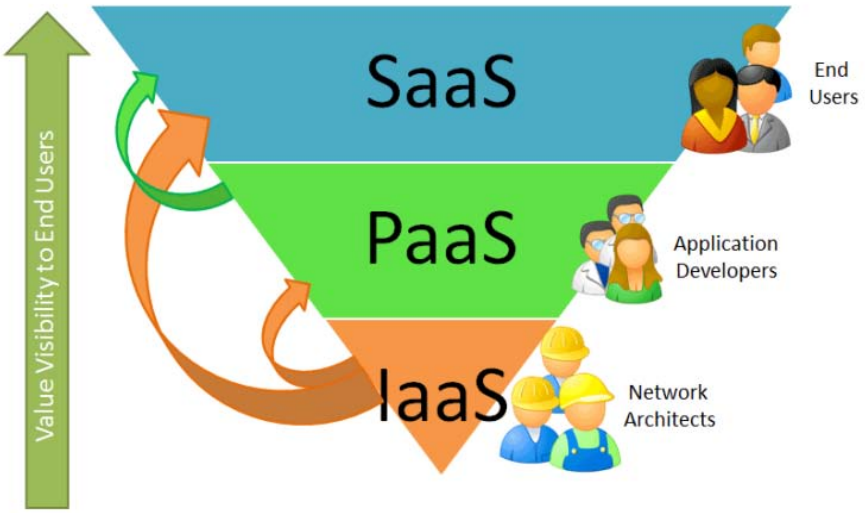

Figure 1: Cloud Computing Stack (Schuller, 2010)

The National Institutes of Standards and Technology (NIST) definition runs to several hundred words but essentially says that:

"*Cloud Computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned" and released with minimal management effort or service provider interaction."*

Armbrust (2009) defined clouds as computers are being able to network anywhere in the world by per-per-use way to pay for used clouds, the actual meaning of this idea is that the resources that are being used will be paid only. Below will be introduces types of clouds. The first one is the Public Clouds. Armbrust (2009) mentioned that public cloud is the traditional cloud computing that will have the opportunity to access to the computing resources from anywhere of the world. The pay-per-use manner will be used in clouds, as defined the only resources that are being used will be paid by transaction fees. On a superficial level, my findings may seem self-evident: a technology company tells its users what it expects of them and users for the most part agree, so long as the technology holds up. However, if we dig deeper, we can extract some important implications from this research. Following the work of other scholars who look at trust in information and communications technologies, I believe that trust is a more useful concept for studying the implications of new technologies than simply looking at privacy.

a) Cloud Computing Deployment Models

There are four deployment models for cloud computing, depending on the owners requirement, the security issues starts from here.

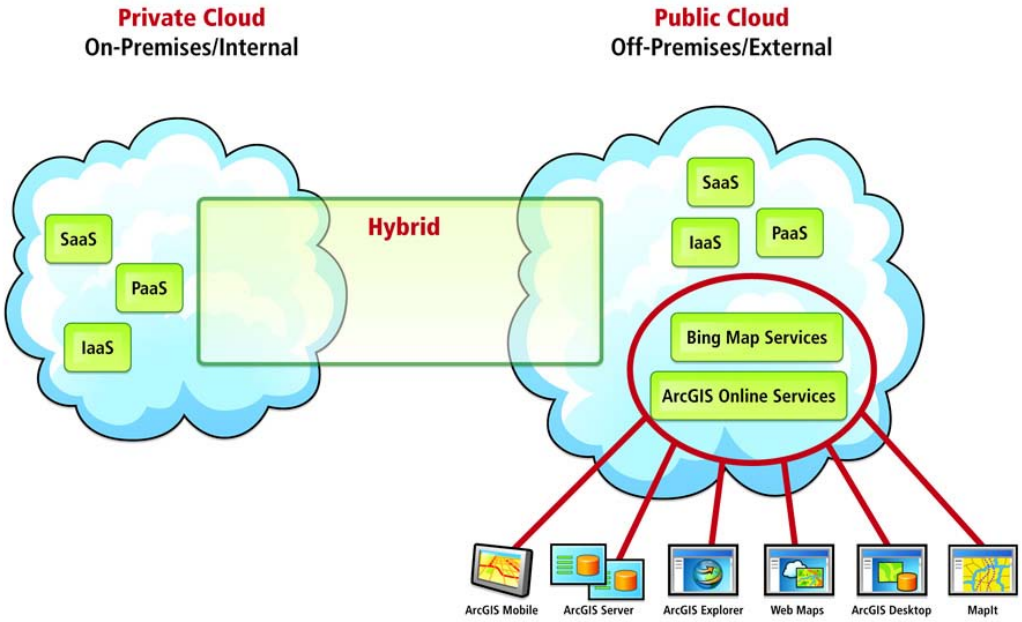

Figure 2: Image representing the clear idea of Cloud Computing Deployment Models

i. *The Public Cloud*

Public cloud computing is based on large-scale offering to general public, the infrastructure is located at premises of provider. The provider owns and manages the cloud infrastructure.

ii. *The Private Cloud*

In this case the infrastructure is provisioned for exclusive use by a single organization. It can be owned, managed, and operated by the organization themselves, a supplier as a third party, or some combination of them. Additionally it can exist on or off premises of the organization (Mell/Grance, 2011). Therefore special forms, can be also considered as a private Clouds. These are listed by some researchers as e.g. virtual private Clouds (e.g. Ried et al. 2011), where the cloud is hosted on dedicated, virtual machines in the data center of the Cloud provider, as well as managed private clouds, where the cloud is hosted by a third party in the data center of the customer.

However, what makes this analysis different than other literature on trust – and what this project is ultimately about on a theoretical level is the nexus of technology, objectivity and trust. A word about each and how they relate is necessary. My analysis and understanding of technology rests on many of the assumptions of actor-network theory, and particularly Bruno Latour's (1991) quip: "Technology is society made durable." I have extensively used the metaphor of the black box to exemplify this principle and stay faithful to the central tenets of Raspberry Pi Cloud Storage. The black box represents a network of associated humans and non-humans that has stabilized such that its heterogeneity is reduced and it appears as a homogenous whole. Its associations (i.e., its social relations) are hidden and become a durable technological entity. Users of this black boxed technology must not understand, follow or even be aware of the internal workings and associations but only need to understand how to use the black box as a whole; they only need to follow the user scripts and interact with the technology's user interface.

III. Methodology

Maxwell (2012) has stated that 'the strengths of qualitative research derive primarily from its inductive approach'. The reason why the inductive approach was used in this research due to the newness of the idea to create your own cloud with Raspberry Pi. A deductive 'approach to this subject would risk restricting the potential avenues of investigation. The inductive approach necessitated attempting to understand the meaning and the context of people's responses to the adoption of Cloud Computing. It attempted to uncover unforeseen trends and detect linkages between Cloud Computing and Raspberry Pi.

a) Data Collection

The focus is on using qualitative data collection techniques. Qualitative data is data that is focusing on delivering information that can be described with terms and theories. It is not like quantitative research that focuses on the numbers behind a survey or something similar. It focuses on delivering numbers and information in terms of quantity. Qualitative research can be interviews or observations where the research is done on the behaviour or theories (Saunders et.al.2010).

Our research will consist of both primary and secondary data. Primary data is data collected by the researcher using different methods. The primary data

that is collected are often more reliable due to that you know where the data comes from and been following the progression all the way. Primary data sources could be (Kelly 2010):

- Observation when observing a system or a research object to see the details that is important to a research. Requires a lot of resources and time.

Secondary data are data that is collected from external sources that already exist. The only thing that has to be done is to look for the data you need. Secondary data has the upside compared to primary that it is cheaper to collect but the reliability, validity and accuracy is not as great. You do not know where the data actually comes from and cannot fully trust is against primary data where you have more control. Secondary data is easier to obtain and cheaper to get also. Some examples of secondary data (Kelly 2010):

- Magazines, Newspapers and Reviews
- Research articles

There is a third data source called tertiary data that is the search tools for obtaining secondary and primary data such as encyclopaedias and indexes. Often it is used in literature search when not knowing where to start searching for a specific topic (Saunders et.al. 2012). Our primary data collection will consist observation in different forms and the secondary data collection will be recent articles in the area of cloud computing and raspberry Pi, internet sources and literature that is within our field of research.

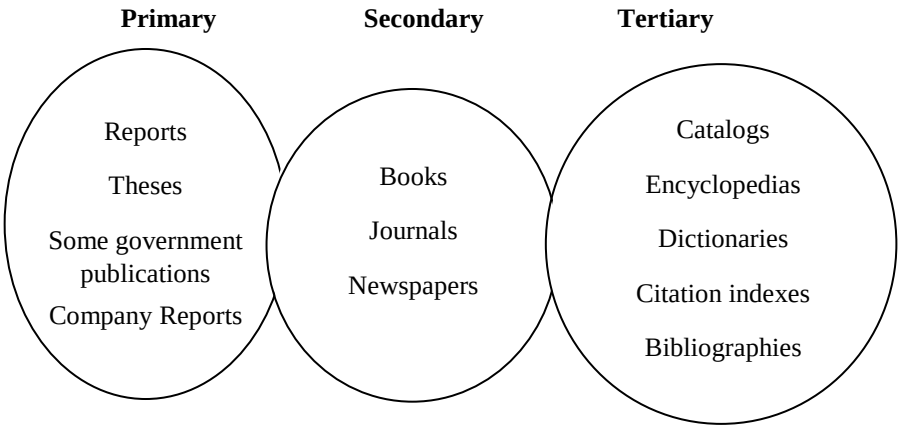

Figure 3: The different literature sources (Saunders et.al. 2009)

IV. Results

Cloud computing is defined by the United States National Institute of Standards and Technology (NIST) as: "A model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction." (National Institute of Standards and Technology 2011).

In a similar vein, Mather, Kumaraswamy and Latif (2009) add that there are five common attributes of all cloud computing services: shared resources, massive scalability, elasticity, pay as you go and the self-provisioning of resources. While these definitions are technical and bureaucratic, they do provide some starting points to discuss what cloud computing is and how it can be analyzed sociologically. In simpler terms, cloud computing broadly refers to computing services, software and platforms that are not owned individually by users and installed locally on their personal computer, but rather accessed via an Internet connection.

From these characteristics of cloud computing, two ideas are of sociological significance. First, the idea of "convenient, on-demand" (National Institute of Standards and Technology 2011) services fits into the idea of what some authors have called a 'convenience culture.' For example, Tierney (1993) argues that one of the defining features of modernity is the consumption of 'conveniences.' He defines convenience as an "ability to mitigate the effects of bodily limits"; for something to be convenient it must make easy and simple an action that was previously difficult, impossible or troublesome.

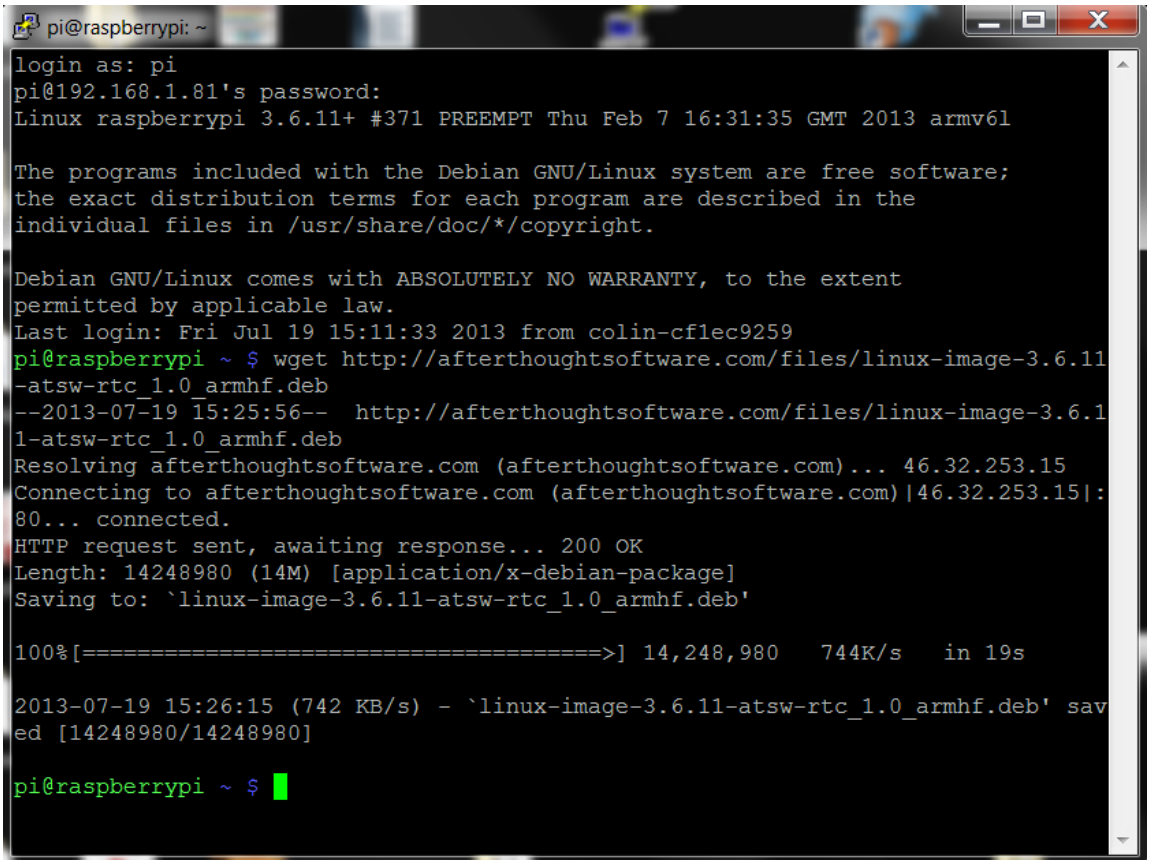

Figure 4: Checkout operator computer with Raspberry Pi installation

a) Raspberry Pi Setup with Owncloud

This is the second installment of the guide to install Owncloud on Raspberry Pi.

There has been certain changes with the latest version of Owncloud 7, and this guide should be able to help you to sail through smooth. Without much ado about what is Owncloud (which you already know) let's dive deep into tech details.

Setup admin account

While setting up the admin account you should provide the path to your data folder. You can ideally set this to your NAS drive or external drive, which you may have mounted. Owncloud will complain and not proceed with admin account if the data directory path is not readable & writable by the user www-data. Also it should not be readable by "others".

b) Setup Mount / Auto Mount USB Hard Drive on Raspberry Pi

Follow the simple steps in the order mentioned below to have your USB drive mounted on your Raspberry Pi every time you boot it.

These steps are required especially if you are setting up a Samba share, or a 24x7 torrent downloader, or alike where your Raspberry Pi must have your external storage already mounted and ready for access by the services / daemons.

Step 0.

Plug in your USB HDD / Drive to Raspberry Pi

If you are using a NTFS formatted drive, install the following

sudo apt-get install ntfs-3g

Step 1.

Log on pi using ssh terminal and execute:

ls -l /dev/disk/by-uuid/

You will see something like the following:

lrwxrwxrwx 1 root root 10 Jan 1 1970 0AC4D607C4D5F543 -> ../../sda1

Note down the value of the UUID --> 0AC4D607C4D5F543

Step 2.

Create a location for mount point:

sudomkdir /media/NASDRIVE

Give proper permission:

sudochmod 770 /media/NASDRIVE

Step 3.

Get the uid, gid for pi user and group with id command (usually 1000)

Step 4.

Mount the USB Drive and then check if it is accessible at /media/NASDRIVE

sudo mount -t ntfs-3g -o uid=1000, gid=1000, umask=007/dev/sda1 /media/NASDRIVE

Note:
ntfs-3g for NTFS Drives
vfat for FAT32 Drives
ext4 for ext4 Drives

Step 5.

Now, we will configure RasPi to do this after every reboot:
Take a backup of current fstab and then edit
sudocp /etc/fstab /etc/fstab.backup
sudonano /etc/fstab

Add the mount information in the fstab file (replace UUID with your own):
UUID=0AC4D607C4D5F543 /media/NASDRIVE ntfs-3g uid=1000, gid=1000, umask=007 0 0

Step 6.
Reboot
sudo reboot

Step 8. (Optional, required if using as data storage for Owncloud)

If you are configuring ownCloud's data directory on your NAS drive, it should be having a 770 permission for www-data user. You can simply add user www-data to pi group, since it's already having 770 as permission as set above in fstab.
sudousermod -a -G pi www-data.

In studies of black boxed technologies, objectivity is always implicit but rarely explicitly discussed. Part of this is likely due to the self-evidence of the objectivity of black boxed technologies. Technological entities are objective in multiple senses of the word: one, they are objects; and two, they behave consistently. Users of a black boxed technology who behave in accordance with the user scripts should expect the technology to behave the same way every time. Its internal workings and associations are reduced such that users provide an input and expect consistent outputs from the black box. The objectivity of technologies is, in this sense, very self-explanatory. Nevertheless, using the metaphor of the black box highlights a key point about the nexus between objectivity and technology: objectivity is an effect of a network or association's durability and stabilization. It is precisely when a network becomes black boxed that it becomes seen as objective. It is when those contingent associations and subjective actors are reduced (in other words, when the social has been reduced) that a technology that behaves consistently and objectively appears.

V. Conclusion

What are some repercussions of this analysis of Raspberry Pi ownCloud storage, users, cloud computing, and data privacy? OpenSSH is an application that allows you to securely access Linux systems remotely over the network. You can use OpenSSH simply for secure file sharing. But it also allows you to log on to a system and control it over the network, even using the GUI, just as if you were sat in front of it. The default installation of Linux on your Raspberry Pi should have "SSH daemon" running. This means that your Raspberry Pi is listening on port 22 for a remote computer asking to make a connection to it. In your case, this will probably mean your normal desktop or laptop computer.

In Chapter Two, I found that there is a dearth of existing literature on Raspberry Pi personal cloud storage and data privacy that examines the role of user agency and that trust is one conceptual tool to avoid this problem. In Chapter Three, I outlined how using principles from Raspberry Pi and bring it in a shape to personal cloud storage, theory one could study trust and reimagine the role of user agency in discussing cloud computing. In particular, I outlined the methodological principles of Akrich's (1992; Akrich and Latour 1992) user scripts and description and the principle of co-production to look at how users and technologies are mutually working on each other. In applying these principles to my case study in Chapters Four and Five, I found that Raspberry Pi personal cloud storage, prescribed trust and privacy protection on its users, while maintaining security as its domain. Users, for the most part, accepted this script; however, some varying levels of trust were observed in users. What was noteworthy about the analysis in Chapter Five, however, was how in instances of breakdown, these user scripts and trust fall to pieces. This has led me to the conclusion that only when trust is shattered does privacy and the trade-off of personal privacy for use become contested and negotiable.

My example of cloud computing, and more specifically Raspberry Pi cloud storage, can highlight this principle. As discussed in the introduction, cloud computing is a great example of a heterogeneous assemblage. It is not a simple technological artefact but is rather an idea that encompasses a wide range of computing programs, hardware and human actors in order to be realized. Through my analysis of users of Raspberry Pi cloud storage, I have found that many of these users simply accept the user scripts of this program; they interact with Raspberry Pi cloud storage as a black boxed technology. They do not need to question or understand how Raspberry Pi cloud storage. Or how cloud computing more generally functions, they just need to know how to interact with Raspberry Pi cloud storage's user interface. To them, Raspberry Pi cloud storage is a program to store and synchronize files to be accessed at a later date, not a heterogeneous network of associated human and non-human actants.

When exploring technology and objectivity in this way, it is possible to conceptualize the understanding of trust I have put forth in this analysis. Trust simply becomes a necessary by-product of the stabilization and reduction of heterogeneous

associations, and the perceived objectivity of a technological entity. Technologies, here understood as stabilized networks, objectivity and trust all necessitate one another. Black boxed technologies are trusted to behave as expected, to behave objectively. If there is distrust in the technology, it is not perceived as completely objective; there is room for error. If the heterogeneity of a network is not completely reduced, there is more room for distrust as users must now trust each of the heterogeneous associations and not simply the homogenous black boxed technology.

Thus, technology, objectivity and trust go hand in hand; they all appear simultaneously as a heterogeneous network is reduced, stabilized and made durable. My findings, in Chapter Four and particularly Chapter Five thus suggest that Raspberry Pi cloud storage as a case study of cloud computing is not a completely stabilized technology, though it is well on its way. It is stable enough that users for the most part trust the technology and behave in accordance with its user scripts. However, there is still a concerted effort on the part of Raspberry Pi cloud storage to communicate its objectivity and trustworthiness and to further reduce its heterogeneity and the marginality of some users.

Acknowledgement

The author would like to thank Mohammed Mahmud for his technical support of the cloud. This work is supported by FICT, Limkokwing University.

References Références Referencias

1. Abraham, (2009). Autonomic Clouds on the Grid. Journal of Grid Computing, 1-18.
2. Aderemi A. Atayero*, Oluwaseyi Feyisetan**. (October 2011). Journal of Emerging Trends in Computing and Information Sciences. *Security Issues in Cloud Computing: The Potentials of Homomorphic Encryption*. VOL. 2 (NO. 10), p 547-550.
3. Andrew Tabona. (2013). *The Top 20 Free Network Monitoring and Analysis Tools for Sys Admins.* Available: http://www.gfi.com/blog/the-top-20-free-network-monitoring-and-analysis-tools-for-sys-admins/. Last accessed 2nd November 2014.
4. Armbrust, M., Fox, A., Griffith, R., Joseph, A., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I. and Zaharia, M. (2009). Above the Clouds: A Berkeley View of Cloud Computing. Technical Report. University of California at Berkeley.
5. Bessie. (December 2012). *Ouya vs Raspberry Pi.* Available: http://ouyaforum.com/showthread.php?755-Ouya-vs-Raspberry-Pi. Last accessed 11 October 2014.
6. Boniface, M., Nasser, B., Papay, J., Phillips, S., Servin, A., Yang, X., et al. (2009). Platform-as-a-Service Architecture for Real-time Quality of Service Management in Clouds.
7. Christodorescu, M., R. Sailer, D. L. Schales, D. Sgandurra, and D. Zamboni (2009). Cloud security is not (just) virtualization security. In Proceedings of the ACM Cloud Computing Security Workshop (CCSW), pp. 97–102.
8. Cisco. (2013). *Cisco Packet Tracer.* Available: http://www.packettracernetwork.com/. Last accessed 2018th November 2014.
9. Cloud Security Alliance, "Top Threat to Cloud Computing V1.0," March 2010. [Online]. Available: https://cloudsecurityalliance.org/topthreats/csathreats.v1.0.pdf
10. CN VR. (17 September 2014). *The Raspberry Pi B.* Available: http://www.ebay.com/itm/CN-Ver-Raspberry-Pi-2-0-Model-B-512MB-ARM11-Linux-System-Development-Board-Kit-/261189408347. Last accessed 11 October 2014.
11. David Oven. (2013). *Ethernet Tutorial - Part I: Networking Basics.* Available: http://www.lantronix.com/resources/net-tutor-etntba.html. Last accessed 2018th November 2014.
12. Foster, I., Zhao, Y., Raicu, I. & Lu, S. (2008). Cloud Computing and Grid Computing 360-Degree Compared, University of Chicago, 10-56.
13. ICO. (2013, Jul.) "Privacy impact assessment handbook," v2.0, Information Commissioner's Office, UK. [Online]. Available: http: //www.ico.org.uk/pia handbook html v2/files/ PIAhandbookV2.pdf
14. ipv6launch.org. 2012. IPV6launch. [ONLINE] Available at: http://www.worldipv6launch.org/support/. [Accessed 09 April 2018].
15. jaipa.or.jp. 2011. Current status and the future direction of IPv6 in Japan. [ONLINE] Available at: http://www.jaipa.or.jp/ipv6day/data/111121_iaetf.pdf. [Accessed 09 April 2018].
16. James Hilton. (June, 2013). *Designing and Building the Best Small Office Network From the Ground Up.* Available: http://www.networkcomputing.com/netdesign/soho1.html. Last accessed 2018th November 2014.
17. Jhony Jackson. (2012). *What is a Computer Network.....?.* Available: http://www.e-tutes.com/. Last accessed 2018th November 2014.
18. Lantronix. (2014). *Networking Tutorials.* Available: http://www.lantronix.com/resources/networking.html. Last accessed 2018th November 2014.
19. Marianthi (2013), "Data protection jurisdiction and cloud computing when cloud users and providers subject to eu data protection law are? the cloud of unknowing, Part 3," International Review of Law, Computers & Technology, vol. 26, pp. 2–3, 2012
20. Meetings.apnic.net. 2012. IPV6 deployement of KDDI. [ONLINE] Available at:http://meetings.apnic.net/__data/assets/pdf_file/0003/44976/ipv6_deployment.pdf. [Accessed 09 April 2018].

This page is intentionally left blank

Global Journal of Computer Science and Technology: B
Cloud and Distributed
Volume 19 Issue 1 Version 1.0 Year 2019
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals
Online ISSN: 0975-4172 | Print ISSN: 0975-4350 | DOI: 10.17406

Homomorphic Encryption Security for Cloud Computing

By J. Phani Prasad & B Seetha Ramulu
Vardhaman College of Engineering

Abstract- As the Data and Technology is increasing day by day the security and privacy issues are becoming a major concern now a days, with the advent of different security mechanisms there are some bottlenecks. In this work we are giving the essence and importance of an encryption scheme called homomorphic encryption and its related issues.

Keywords: *encryption, homomorphic encryption, RSA, security.*

GJCST-B Classification : *K.6.m*

Strictly as per the compliance and regulations of:

Homomorphic Encryption Security for Cloud Computing

J. Phani Prasad [α] & B Seetha Ramulu [σ]

Abstract- As the Data and Technology is increasing day by day the security and privacy issues are becoming a major concern now a days, with the advent of different security mechanisms there are some bottlenecks. In this work we are giving the essence and importance of an encryption scheme called homomorphic encryption and its related issues.

Keywords: encryption, homomorphic encryption, RSA, security.

I. Introduction

Security of the Data or information is a primary concern of the day to day transactional process. There are various domains in which security can be provided as a part of protecting the things and systems, some of the security areas or domains are: home automation and security, providing the security to data pertaining to different organizations. The nominal and quite common security providing methods is usage of passwords in the form of text to our data and having some sort of security to it.

Another form of password protection other than text is providing captcha, or one time passwords now a days. In search engines like Google we can keep multiple authentication schemes like each time one enters in to his Gmail account one can put two factor authentication.

Apart from the above methods we can encrypt our data while storing it/sometimes whenever we send our information to someone, lot of encryption algorithms and methods are available like DES, AES, RSA, Blow fish and many more, and at the other side the receiver will use a method called decryption to receive the actual data from sender. Cryptography is the method of converting plain text to cipher text. By this method one can just read our data and use it for his purpose without altering that data.

II. Literature Review

Rivest et.al.(1978) introduced for the first time the concept of Homomorphic encryption. Taher (1985) introduced an algorithm based on multiplicative property.

Shahzadi et al (2012) done the study on the three homomorphic encryption algorithms. Naser and Bin (2013) surveyed on specific security issues and use of cryptography in cloud computing.

Carlos *et al*. (2013) discussed about the recent advances in homomorphic encryption techniques. They have done survey on recent advances in Somewhat Homomorphic Encryption (SWHE) and Fully Homomorphic Encryption (FHE) algorithms.

Liu (2012) has introduced some cloud computing system and also analyzes cloud computing security problem. He suggested that single security technique cannot be used to solve the cloud security problem therefore, many traditional and some new strategies are required to use together to provide the total security in cloud.

Ustimenko and Wroblewska (2013) proposed an idea for homomorphic encryption and multivariate key for cloud security. They have given detailed discussion on Key Dependent Message (KDM) encryption scheme can be used for cloud security.

Ramgovind *et al*. (2010) highlighted key security considerations currently faced by industry.

Aderemi and Oluwaseyi (2011) discussed about the security issues in cloud computing and the potentials of homomorphic encryption, and proposed an encryption layer on top of the encrypted data on the cloud.

III. Homomorphic Encryption

Homomorphic encryption is a Technique that permits the calculation on encrypted data without prior decryption and after operation, if the data is decrypted by user which is in encrypted form it gives actual result without knowing the actual plain text (yang et al. 2014).

Suppose if plain text is M.

Operation(M) decrypt(Operation(encrypt(M))).

In figure1 (below) homomrphic encryption is applied on some set of integer values using some algorithm. For encryption algorithm is implemented as 7*2=14 where 2 is the encrypted element in the above operation, and for 5 ie 5*2=10. The reverse process is followed for decryption of data here after multiplication (14*10)/2 =70 it is divided by 2 because of homomorphic encryption property, after decryption of result we will get 7*5=35 which is actual result.

Author α: Assistant Professor, Vardhaman College of Engineering, Hyderabad. e-mail: phanimtechcse@gmail.com
Author σ: Professor, Vardhaman College of Engineering, Hyderabad.

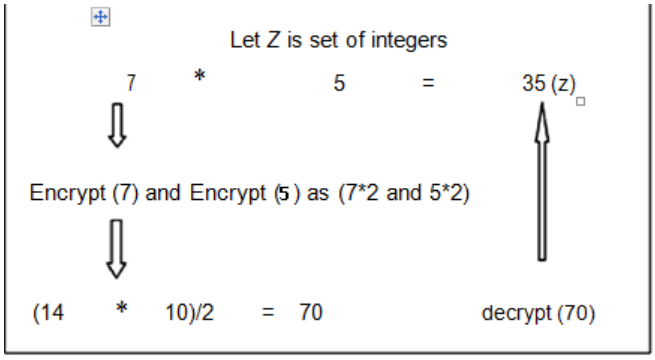

Figure 1: Homomorphic Encryption on Integers

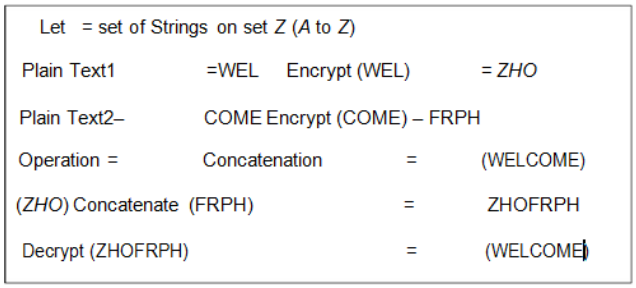

Figure 2: Homomorphic Encryption on Strings

IV. Homomorphic Encryption and Types

There are three types of Homomorphic encryption available:

1. Partial Homomorphic Encryption(PHE)

   In this encryption technique it performs single operation on encrypted data i.e either multiplication or addition but not both.

2. Some What Homomorphic Encryption(SWHE)

   In this technique it support limited number of addition and multiplication operations on encrypted data.

3. Fully Homomorphic Encryption(FHE)

   In this technique it support both multiplication and addition operations and also any other computations also possible on encrypted data.

a) RSA and Homomorphic Encryption

RSA is a asymmetric algorithm used for encryption of the data , which was introduced by Ron Rivest, Shamir and ad leman , it is mainly used for encryption using public and private key concepts till now, but it can be combined with homomorphic encryption.

The properties of Homomorphic encryption are:

Encrypt (p1 p2) = Encrypt(p1) + Encrypt(p2) (additive homomorphic property).

Encrypt (p1* p2) = Encrypt(p1) X Encrypt(p2) (multiplicative homomorphic property.

b) Multiplication Homomorphism using RSA

start

1. Select two large prime numbers r, s
2. Compute n=r*s
3. Pii(n)=(r-1)*(s-1)
4. Select e, where 1<e<pii(n) and e, n are co primes
5. Compute d as (d*e)mod(pii(n))=1
6. Public key{e,n},private key{d, n}
7. Encryption of plaintext $C=M^e \bmod n$
8. Decryption of cipher text $M=c^d \bmod n$
9. RSA follows homomorphic property as:

   Encrypt(p1* p2) = Encrypt(p1) X Encrypt(p2) (multiplicative homomorphic property).

   End.

V. Partial Homomorphic Encryption

Partial homomorphic encryption can be implemented in various domains like network security, cloud computing, Big data and many other fields where security of data and storage is the major concern.

For example if we take cloud to secure the data, so many encryption and decryption algorithms are in to practice, but among them the Partial Homomorphism is the technique which reduces the amount of computation when compared with other algorithms.

VI. Implementation of RSA as PHE using a Case Study on Cloud

We are implementing a small case study by taking the length and breadth of 50 grounds and the number of persons visiting the ground for playing and other activities. Ground shape is assumed to be different i.e (rectangle or square). The data is encrypted and it is stored in a cloud by the user, and whenever it is required user can compute the area of the ground.

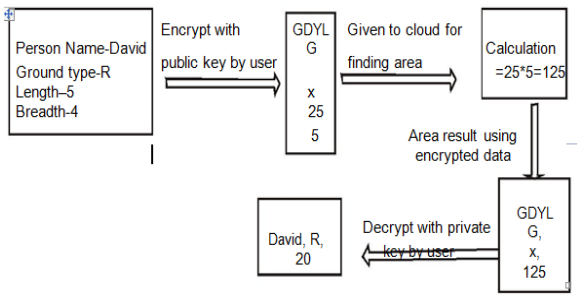

Figure 3: Computation of area in cloud

Here in the above computation the length value 5 is encrypted as 25 and breadth value 4 is encrypted as 5 and the area is computed as 25*5 which is 125, and David is encrypted as GDYLG now, the user at the other end will compute the decryption on the encrypted data , and the actual result is obtained.

By using RSA algorithm with the help of public key the data is encoded as {3,55} that is stored in

cloud, and the area is computed with the help of encrypted data by the formula (encrypt (length)*encrypt(breadth)) and the computed result is sent to user. The user takes the encrypted form of data and decrypts it by using private key as {27,33} to get the actual area of the ground.

VII. Conclusion

In this work we have discussed about the homomorphic encryption technique as a method of providing security to the data in various fields, and mainly on cloud. We have also implemented RSA as a partial homomorphic technique on cloud by taking a case study. In future it may be extended and the research directions has to be driven towards fully homomorphic encryption technique.

References Références Referencias

1. Taher El Gamal (1985), "A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms", in *Advances in Cryptology*, pp. 10-18, Springer Berlin Heidelberg.
2. Shahzadi Farah *et al.* (2012), "An Experimental Study on Performance Evaluation of Asymmetric Encryption Algorithms", Recent Advances in Information Science, Proceeding of the 3rd European Conf. of Computer Science, (EECS-12).
3. Rivest Ronald L, Adleman Leonard M and Dertouzos Michael (1978), "On Data Banks and Privacy Homomorphism", *Foundations of Secure Computation,* Vol. 4, No. 11, pp. 169-180.
4. Carlos Aguilar Melchor, Simon Fau, Caroline Fontaine *et al.* (2013), "Recent Advances in Homomorphic Encryption", *IEEE Singal Processing Magazine*, March, pp. 108-117.
5. Liu Wentao (2012), "Research on Cloud Computing Security Problem and Strategy", Proceedings of IEEE Conference.
6. Ramgovind S, Eloff M M and Smith E (2010), "The Management of Security in Cloud Computing", Proceedings of IEEE Conference
7. Naser A W S and Bin Md Fadli (2013), "Use of Cryptography in Cloud Computing", pp. 179-184, Proceedings of IEEE International Conference on Control System, Malaysia
8. Ogburn Monique, Turner Claude and Dahal Pushkar (2013), "Homomorphic Encryption", *Proceeding Computer Science*, Vol. 20, pp. 502-509
9. Rastogi Garima and Sushil Rama (2015), "Cloud Computing Implementation: Key Issues and Solution", Proceedings of IEEE Conference INDIACOM, pp. 173-179.
10. Gentry C (2009), "Fully Homomorphic Encryption Using Ideal Lattices", *ACM Symposium on Theory of Computing*, pp. 169-178.
11. Garima Rastogi and Rama Sushil (2015),"cloud computing security and homomorphic Encryption" pp 48-58.

This page is intentionally left blank

Global Journal of Computer Science and Technology: B
Cloud and Distributed
Volume 19 Issue 1 Version 1.0 Year 2019
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals
Online ISSN: 0975-4172 | Print ISSN: 0975-4350 | DOI: 10.17406

An Analytic Real-Time Framework for IoT Based Home Automation System

By Nazmul Hossain, Rafia Sultana, Farzana Akter Lima & Md. Arif Rahman
Jessore University of Science & Technology

Abstract- The Internet of Things or IoT means the ability to connect billions of physical devices around the world that are now linking to the internet for collecting and sharing data. Internet of Things (IoT) technologies is used to sense the real-time primordial manufacture data and included the energy alliance data and the provision circumstance data. The Internet of Things (IoT) will approve any contents to be sensed or monitored remotely wherever there any remaining network infrastructure, making amenities for the integration of the actual world into computer-based systems. Real-time narrates the path of flowing media is processing. In the real-time procedure, anyone can entrance information barring to narrate for it and save our time.

Keywords: internet of things (IoT), IOT, home automation, smart home, response time, control system, real-time, performance matric.

GJCST-B Classification : I.2.1

Strictly as per the compliance and regulations of:

An Analytic Real-Time Framework for IoT Based Home Automation System

Nazmul Hossain [α], Rafia Sultana [σ], Farzana Akter Lima [ρ] & Md. Arif Rahman [ω]

Abstract- The Internet of Things or IoT means the ability to connect billions of physical devices around the world that are now linking to the internet for collecting and sharing data. Internet of Things (IoT) technologies is used to sense the real-time primordial manufacture data and included the energy alliance data and the provision circumstance data. The Internet of Things (IoT) will approve any contents to be sensed or monitored remotely wherever there any remaining network infrastructure, making amenities for the integration of the actual world into computer-based systems. Real-time narrates the path of flowing media is processing. In the real-time procedure, anyone can entrance information barring to narrate for it and save our time.

In our proposed system, we build a system where we calculate real-time. To compute real-time, we need an IoT based automated system. And here we use an IoT based home automation system. We are passing data through the system for collecting data with the help of Cisco Packet Tracer simulator. For calculating real-time performance, we use six performance metrics to evaluate the event detection system performance. They are sensitivity, specificity, precision, accuracy, F1-score, and G-mean.

Keywords: internet of things (IoT), IOT, home automation, smart home, response time, control system, real-time, performance matric.

I. Introduction

The Internet of Thing (IoT) refers to gathering and sensing data from various apparatus about our daily life corporal phenomena [1, 2]. IoT has acquired capacious popularity owing to its extensive market in several sectors such as home automation, healthcare, security, etc. [3]. IoT entangles pervading internet connectivity beyond valuable devices to any province of traditionally daft or non-internet-enabled naturalistic devices and quotidian things [4]. IoT was begotten from machine-to-machine (M2M) interaction. Machines are joining to the objects via a network except for human communication [5].

Author α: Assistant Professor, Department of Computer Science & Engineering, Jessore University of Science & Technology (JUST), Jessore-7408, Bangladesh. e-mail: nazmul.justcse@gmail.com
Author σ ρ: Department of Computer Science & Engineering, Jessore University of Science & Technology (JUST), Jessore-7408, Bangladesh. e-mails: rafia130108@gmail.com, farzanaoshmi.cse@gmail.com
Author ω: Assistant Professor, Department of Computer Science & Engineering, Jessore University of Science & Technology (JUST), Jessore-7408, Bangladesh. e-mail: ma.rahman@just.edu.bd

Real-time is a level of computer compassionate that a customer senses as sufficiently early or that qualify the computer to repose up with several processes (for example, to attendant visualizations of the weather as it incessantly mutations) [6]. Real-time is an epithet regarding computers or processes that handle in real-time. Real-time describes a man rather than an engine sense of time [7].

Real-time computing (RTC) is a period of computing decision that has inelastic instant constraints. Real-time computing must be committed in a time structure that is relatively invisible to the user [8]. A real-time system is any message processing system which must answer to externally propagated input incentive between a limited and specified epoch [9]. If it overcomes this time bondages, it's output in accomplishment erosion and malfunction of the process [10]. Denoting to the data-scheming system in which a computer takes every moment transforming data, such as knowledge relating to air-traffic monitoring, travel booking procedure and processes it practically to be capable of controlling the origin of data [11].

a) Characteristics of Real-Time System

- Time Encirclement.
- Modern Rectification Inference.
- Interwoven.
- Security-Criticality.
- Concurrency.
- Allotted and Recompose Formation.
- Ought Troublesome.
- Usage Hardware.
- Durability.

II. Background and Related Work

In 1970s Minicomputers specially built for dedicating embedded systems such as DOG scanners, onward the necessity for low-latency priority-driven with accessing data and operating systems suchlike Data General's RDOS (Real-Time Disk Operating System) and RTOS's background and foreground scheduling as Digital Equipment Corporation's RT-11 date from this age. On time when the MOS Technology 6502 and the Motorola 68000 were exoteric relatives could conduct their abode computer as a real-time system. The chance to deactivate several intermissions allowed for hard-coded loops with fixed timing and the less interrupt

latency allowed the impersonation of a real-time system [12].

We implemented and modified our system with this two-existing system. In the procedure 1, Smart home regulating devices are using for guiding the systems by sending data to govern the actuators. Here they also used the home gateway for relating all home appliances and interconnected with the Internet on the open network part, and the home networks also control the home machinery. The home gateway gives the ability to control the connected equipment on the home networks that controlled with smart instruments and passed data through the system and collect data for calculating real-time [13]. In the procedure 2, there submitted a low cost secure mobile phone-based home automation system [14, 15]. Appliances at home are linking with the Arduino BT board, and the communication within the mobile phone and the Arduino BT board is wireless [16]. Superfluous apparatus can be connected into the system, and it should be faster enough to actualize the power of wireless technology and be cost effective. In this system, they also are measuring data for calculating real-time [17, 18].

In our system, we connected all devices with the server and motion sensor for controlling our automation system. We passed data through our implemented process for measuring the time that is needed to turn on/off the equipment.

III. Proposed System

a) System Description

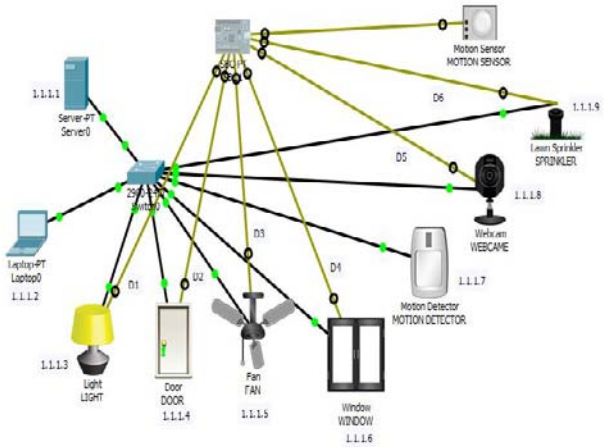

Fig. 1: System Implementation of the Proposed System

For calculating real-time performance, we implement a virtual system in Cisco Packet Tracer. We use server and motion sensor for connecting the devices. The equipments are light, fan, door, window, sprinkler, webcam and motion detector. The sensor is joining to the SBC-PT Board, which is one kind of networking access board. All equipment attached with the server through a unique IP address. Entering to the server and viewing the activity of the system by home authority there use a laptop. To collect real-time data, we passed packets through the system and putting it to the performance matrices equation for calculating the real-time action of our system. We also build the other two existing methods in Cisco Packet Tracer for comparing our systems performances [19].

b) User Case Diagram

User case provides a path, how the user gets the entrance license of the system. A user cannot straightly access the arsenal so that they needed to exact authentication. Here also we see equipment internally linked with the system action. Use cases are typically induced by a user to meet the goals detail activities and variants inlaid in gaining the aim [20].

Flow:

- Admin ⟶ Managing & Controlling Home Appliances ⟶ Logout.
  Admin is the prime controller for managing, measuring and controlling the access of the services.

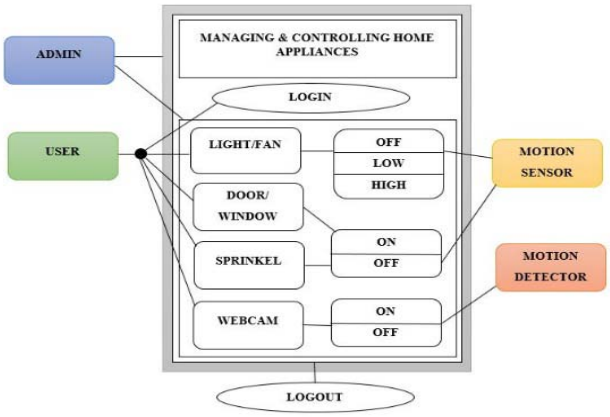

Fig. 2: User Case Diagram

- User ⟶Login ⟶ Using Equipment ⟶ Logout.
  To start using the system, the user must use the login interface to log in into the system. When the user wants to control the equipment, they first log in to the service with their password. If their passwords match, then they get the access. After using the device, they are log Out from the process.
- Ingredient ⟶ Motion Sensor.
  Light, Fan, Door, Window, Sprinkle connected to the motion sensor. When the sensor detected any person, then it is automatically on/off.
- Ingredient ⟶ Motion Detector.
  Webcam connected to the motion detector. If any person passes through a motion detector, then it is automatically on/off. Therefore, we see the passing passenger.

c) Layer Development

Layering is the association of programming into individual functional elements that interact in various

sequential and hierarchical way, including each layer generally having an interface alone to the layer above and below it [21].

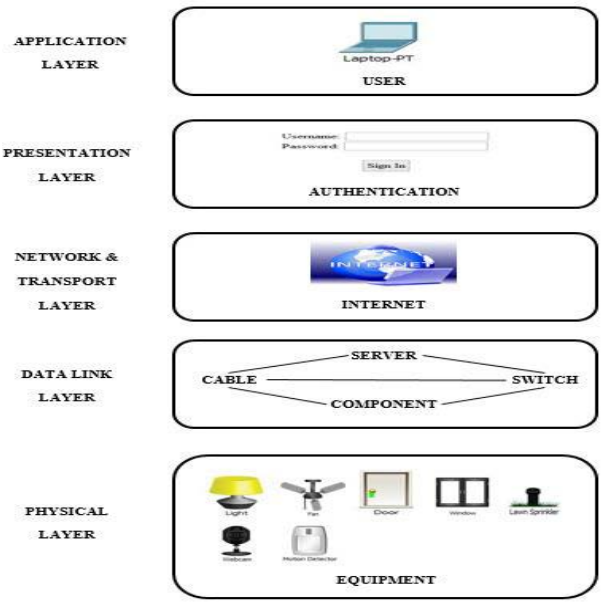

Fig. 3: Layer Development

- *Physical Layer:* It gives the hardware using sending and receiving data on a carrier, together with demarking cables, cards, and physical aspects.
- *Data Link Layer:* This layer shows the functional and procedural means to trek data between network essences and detects errors that occur in the physical layer.
- *Network & Transport Layer:* The network layer is consisting of a variant form of wired and wireless contact networks such as Wireless Local Area Network (WLAN), Internet. This layer is liable for the sending and receiving information from the perceptive layer. Transport layer is a conceptual partition of a scheme in the layered masonry of protocols in the network pile in the Internet Protocol Suite. The transport layer is the foundation of the Internet.
- *Presentation Layer:* It offers protocol conversion, data translation, compression, and encryption. It provides the real-time succession requisite for communication between objects that are layers, systems or networks.
- *Application Layer:* This layer creates an interface between the user and system performance.

IV. Analysis Of Proposed System

a) Data Collection

For calculating real-time, we pass packet through equipment from starting point to end. We pass the packet into two different ways. And the ways are 1) Equipment to server & 2) Server to Equipment. In this process we listed time, some packets are passed that called passing time or true position time. Here true position time denoted as x & y. And some packets are not passed that called obstacle time or false position. We denote false position as x' & y' [22, 23].

Table 1: Confusion Matrix for an Event Detection Problem

	Equipment to Server	Server to Equipment
Equipment to Server	Passing Time(x)	Obstacle Time(x')
Server to Equipment	Obstacle Time(y')	Passing Time(y)

To calculate real-time, we use six different types of performance metrics. The metrics are Sensitivity, Specificity, Precision, Accuracy, F1-score, G-mean. We can apply all the performance metrics into seven different equipment such as light, fan, door, window, sprinkler, webcam and motion detector.

The performance metrics equations and descriptions are:

- *Sensitivity:* Sensitivity contents the proportion of true positives that are identified.

  $$\text{Sensitivity} = \frac{x}{x+x'}$$
- *Specificity:* Specificity extents the proportion of real negatives that are identified.

  $$\text{Specificity} = \frac{y}{y+y'}$$
- *Precision:* Precision is a representation of indiscriminately errors, a sum of statistical inconstancy.

  $$\text{Precision} = \frac{x}{x+y'}$$
- *Accuracy:* Accuracy is the caliber of the instrument to the volume the actual value. It is a tracing of systematic fault, a measure of statistical favor.

  $$\text{Accuracy} = \frac{x+y}{x+x'+y+y'}$$
- *F1-score:* The F1-score (F1-sc.) is the harmonic mean of precision (Prec.) and sensitivity.

  $$\text{F1-score} = \frac{2x}{2x+x'+y'}$$
- *G-mean:* The G-mean is computing by taking the square average value of sensitivity and specificity.

  $$\text{G-mean} = \sqrt{\text{Sensitivity} + \text{Specificity}}$$

b) Comparison

Previously we discussed two different systems. We can also implement these systems and passes packet through that systems for calculating the passing time and obstacle time. With the help of performance metrics, we calculate real-time through this passing time and obstacle time.

Comparison of Light:

Table 2: Data collects for Light (Sensitivity)

Light			
	System 1	System 2	Proposed System
Sensitivity	0.339	0.759	0.368
	0.273	0.740	0.398
	0.412	0.766	0.461
	0.349	0.783	0.484
	0.402	0.793	0.518
	0.353	0.799	0.571
	0.414	0.786	0.671
	0.374	0.776	0.669
	0.414	0.782	0.679
	0.452	0.787	0.805

The Graph for Light (Sensitivity): System 1 goes zigzag, and system 2 goes from high to low but our system goes from low to high.

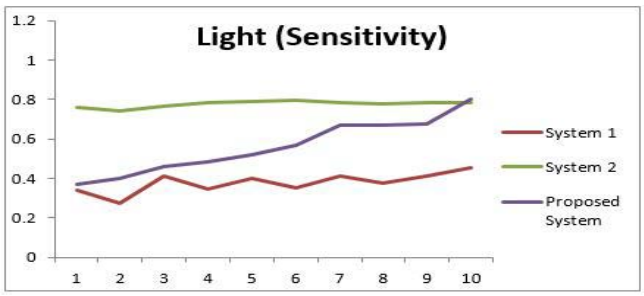

Fig. 4: Light (Sensitivity)

Table 3: Data collects for Light (Specificity)

Light			
	System 1	System 2	Proposed System
Specificity	0.467	0.692	0.394
	0.293	0.705	0.421
	0.388	0.789	0.519
	0.446	0.710	0.538
	0.493	0.719	0.541
	0.457	0.712	0.597
	0.475	0.697	0.669
	0.503	0.687	0.681
	0.525	0.668	0.721
	0.503	0.673	0.881

The Graph for Light (Specificity):

Our system's specification is better than other two process because it is an up-ordering line.

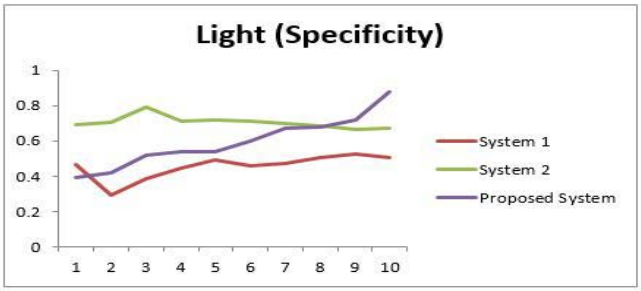

Fig. 5: Light (Specificity)

Table 4: Data collects for Light (Precision)

Light			
	System 1	System 2	Proposed System
Precision	0.437	0.743	0.479
	0.266	0.727	0.503
	0.373	0.761	0.530
	0.332	0.720	0.532
	0.409	0.727	0.539
	0.377	0.718	0.554
	0.411	0.703	0.558
	0.395	0.692	0.575
	0.435	0.673	0.622
	0.459	0.677	0.708

The Graph for Light (Precision):

Our system precision is better than other.

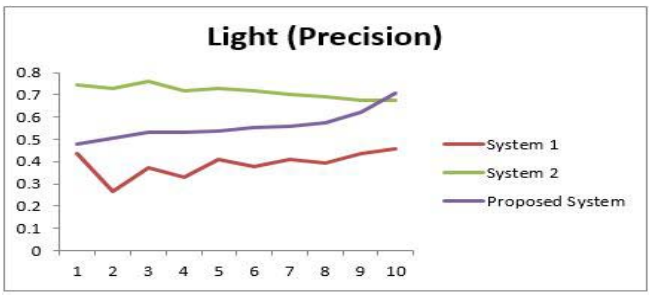

Fig. 6: Light (Precision)

Table 5: Data collects for Light (Accuracy)

Light			
	System 1	System 2	Proposed System
Accuracy	0.397	0.729	0.499
	0.284	0.723	0.500
	0.399	0.758	0.528
	0.403	0.746	0.537
	0.450	0.755	0.551
	0.407	0.754	0.576
	0.447	0.739	0.585
	0.443	0.729	0.625
	0.473	0.721	0.669
	0.478	0.726	0.821

The Graph for Light (Accuracy):

System 1 and system 2 are creating down order line, but our system is high-ordered.

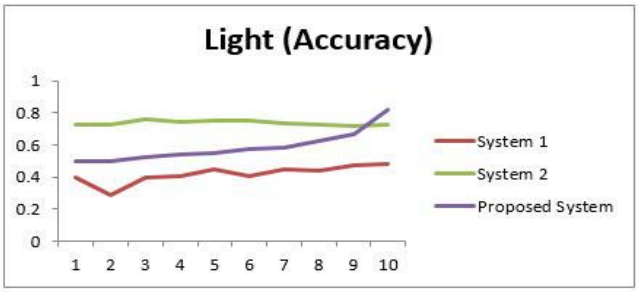

Fig. 7: Light (Accuracy)

Table 6: Data collects for Light (F1-score)

Light			
	System 1	System 2	Proposed System
F1-score	0.382	0.751	0.417
	0.269	0.734	0.456
	0.391	0.764	0.493
	0.340	0.750	0.517
	0.406	0.758	0.529
	0.365	0.757	0.573
	0.413	0.742	0.575
	0.384	0.731	0.609
	0.424	0.723	0.693
	0.455	0.728	0.702

The Graph for Light (F1-score):

The value of F1-score in our system is excellent than the other two methods.

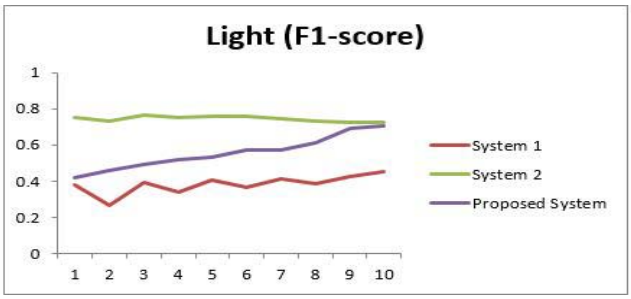

Fig. 8: Light (F1-score)

Table 7: Data collects for Light (G-mean)

Light			
	System 1	System 2	Proposed System
G-mean	0.898	1.205	0.982
	0.752	1.202	1.001
	0.894	1.247	1.028
	0.892	1.222	1.033
	0.946	1.229	1.044
	0.900	1.229	1.044
	0.943	1.218	1.069
	0.936	1.209	1.126
	0.969	1.204	1.160
	0.977	1.208	1.249

The Graph for Light (G-mean):

We are measuring the value of three systems and plotting the value for generating the graph line, and line is increasing.

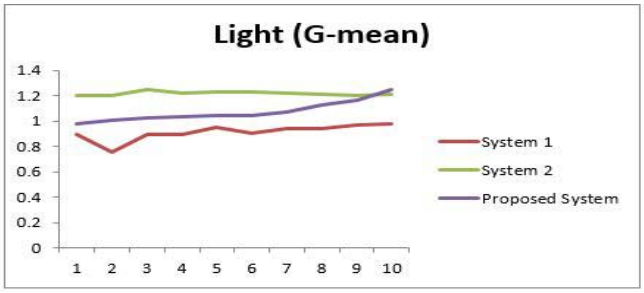

Fig. 9: Light (G-mean)

Comparison of Fan:

Table 8: Data collects for Fan (Sensitivity)

Fan			
	System 1	System 2	Proposed System
Sensitivity	0.497	0.425	0.218
	0.524	0.798	0.232
	0.558	0.825	0.243
	0.586	0.867	0.248
	0.579	0.779	0.252
	0.576	0.791	0.255
	0.566	0.762	0.264
	0.570	0.775	0.283
	0.569	0.752	0.375
	0.555	0.764	0.454

The Graph for Fan (Sensitivity):

For comparing three systems, we saw that our system's sensitivity is better than others.

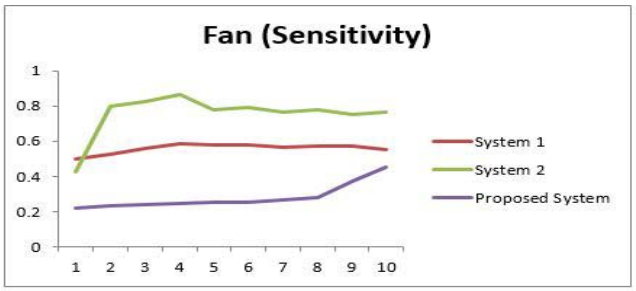

Fig. 10: Fan (Sensitivity)

Table 9: Data collects for Fan (Specificity)

Fan			
	System 1	System 2	Proposed System
Specificity	0.323	0.854	0.292
	0.491	0.752	0.334
	0.595	0.783	0.394
	0.496	0.736	0.437
	0.523	0.767	0.499
	0.571	0.762	0.527
	0.579	0.777	0.572
	0.594	0.787	0.593
	0.573	0.765	0.648
	0.572	0.774	0.843

The Graph for Fan (Specificity):

We collect data and compare with them and saw that the proposed system is proper.

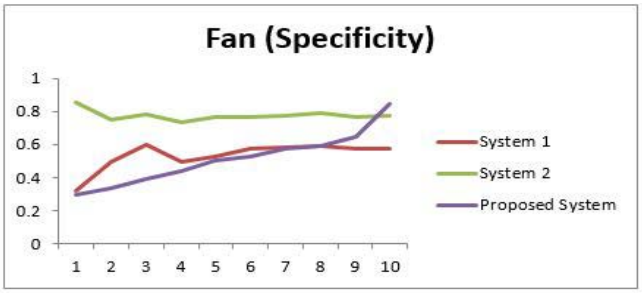

Fig. 11: Fan (Specificity)

Table 10: Data collects for Fan (Precision)

Fan			
	System 1	System 2	Proposed System
Precision	0.545	0.392	0.271
	0.582	0.688	0.280
	0.552	0.752	0.287
	0.538	0.777	0.292
	0.555	0.745	0.305
	0.597	0.789	0.348
	0.601	0.760	0.357
	0.613	0.774	0.367
	0.577	0.751	0.389
	0.563	0.763	0.513

The Graph for Fan (Precision):

From the table and graph, we predicted that the precision of our system is the best than others.

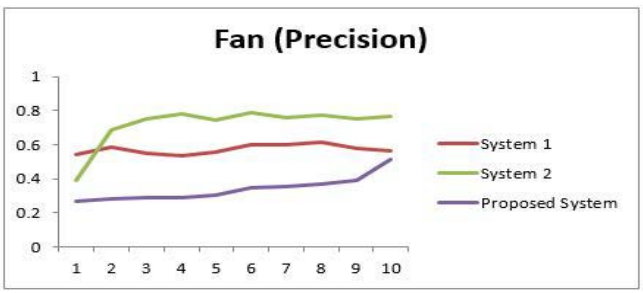

Fig. 12: Fan (Precision)

Table 11: Data collects for Fan (Accuracy)

Fan			
	System 1	System 2	Proposed System
Accuracy	0.431	0.777	0.268
	0.509	0.771	0.294
	0.528	0.802	0.315
	0.541	0.804	0.355
	0.552	0.773	0.374
	0.573	0.778	0.389
	0.572	0.769	0.453
	0.581	0.781	0.489
	0.571	0.758	0.503
	0.564	0.769	0.739

The Graph for Fan (Accuracy):

System 1 and 2 is down order line. Our system is emersion, so it is enriching the graph.

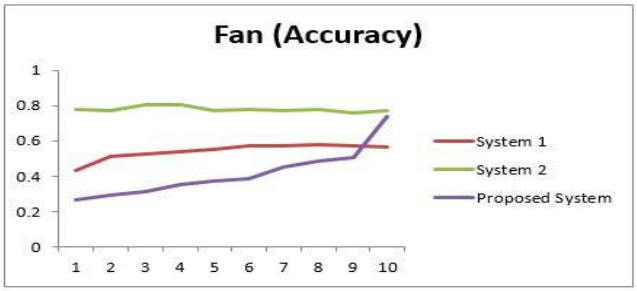

Fig. 13: Fan (Accuracy)

Table 12: Data collects for Fan (F1-score)

Fan			
	System 1	System 2	Proposed System
F1-score	0.520	0.408	0.242
	0.551	0.739	0.254
	0.555	0.787	0.263
	0.561	0.819	0.271
	0.567	0.762	0.274
	0.586	0.791	0.297
	0.583	0.761	0.300
	0.591	0.774	0.319
	0.574	0.751	0.382
	0.559	0.764	0.481

The Graph for Fan (F1-score):

Plotting the value and choose a better system and our system is better than others.

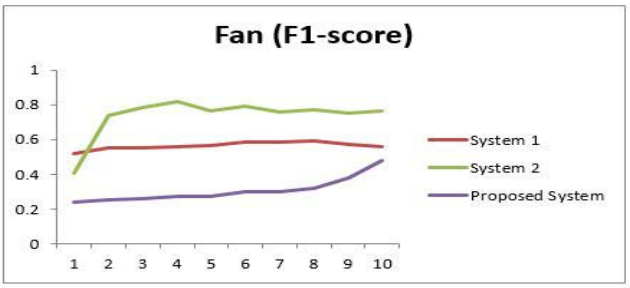

Fig. 14: Fan (F1-score)

Table 13: Data collects for Fan (G-mean)

Fan			
	System 1	System 2	Proposed System
G-mean	0.906	1.131	0.735
	1.007	1.245	0.773
	1.026	1.268	0.806
	1.040	1.266	0.849
	1.049	1.243	0.852
	1.071	1.246	0.863
	1.070	1.241	0.914
	1.079	1.249	0.949
	1.069	1.232	0.973
	1.062	1.240	1.139

The Graph for Fan (G-mean):

For fan, the G-mean is elder than system 1 and system 2 because it is emergence from others.

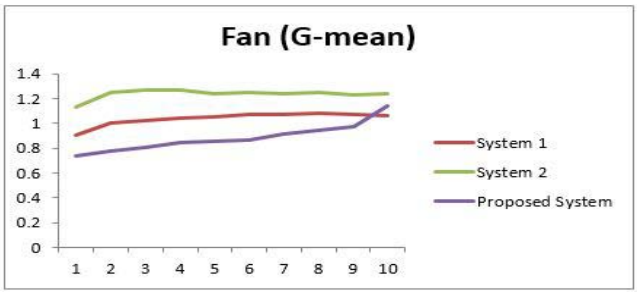

Fig. 15: Fan (G-mean)

Comparison of Door:

Table 14: Data collects for Door (Sensitivity)

Door			
	System 1	System 2	Proposed System
Sensitivity	0.455	0.638	0.096
	0.522	0.542	0.236
	0.607	0.456	0.272
	0.612	0.628	0.289
	0.615	0.704	0.298
	0.578	0.739	0.304
	0.555	0.761	0.309
	0.572	0.772	0.329
	0.552	0.781	0.357
	0.526	0.788	0.367

The Graph for Door (Sensitivity):

System 1 rise low to high and low. System 2 is ascent high to low and high. Our system goes low to high without breaking.

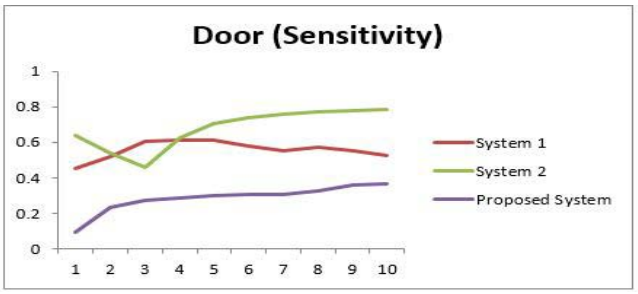

Fig. 16: Door (Sensitivity)

Table 15: Data collects for Door (Specificity)

Door			
	System 1	System 2	Proposed System
Specificity	0.427	0.787	0.486
	0.446	0.820	0.554
	0.486	0.826	0.573
	0.463	0.789	0.615
	0.492	0.773	0.619
	0.495	0.805	0.644
	0.536	0.759	0.649
	0.523	0.687	0.666
	0.520	0.762	0.737
	0.542	0.747	0.949

The Graph for Door (Specificity):

Plotting the value and generate the graph for three system and our system graph is better.

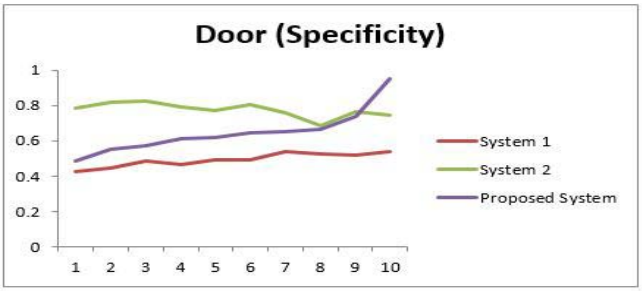

Fig. 17: Door (Specificity)

Table 16: Data collects for Door (Precision)

Door			
	System 1	System 2	Proposed System
Precision	0.492	0.756	0.334
	0.467	0.589	0.334
	0.498	0.486	0.335
	0.473	0.588	0.335
	0.499	0.641	0.336
	0.500	0.717	0.338
	0.540	0.716	0.347
	0.527	0.646	0.365
	0.506	0.732	0.373
	0.508	0.745	0.391

The Graph for Door (Precision):

For metering the effect of the specification of fan we collect and compare data for three system and the proposed system is strong.

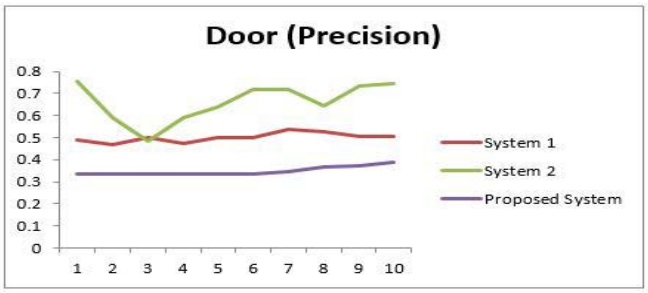

Fig. 18: Door (Precision)

Table 17: Data collects for Door (Accuracy)

Door			
	System 1	System 2	Proposed System
Accuracy	0.443	0.713	0.399
	0.482	0.731	0.438
	0.541	0.728	0.465
	0.529	0.737	0.494
	0.547	0.748	0.496
	0.534	0.779	0.515
	0.545	0.760	0.542
	0.547	0.723	0.548
	0.535	0.771	0.555
	0.534	0.767	0.751

The Graph for Door (Accuracy):

System 1 and 2 are devious, and our system's accuracy is the best because it is a rising line.

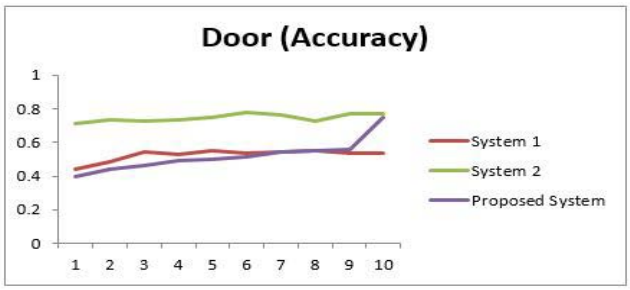

Fig. 19: Door (Accuracy)

Table 18: Data collects for Door (F1-score)

Door			
	System 1	System 2	Proposed System
F1-score	0.983	0.688	0.152
	0.493	0.565	0.278
	0.547	0.470	0.281
	0.533	0.607	0.310
	0.551	0.671	0.315
	0.536	0.728	0.319
	0.548	0.738	0.321
	0.549	0.703	0.338
	0.528	0.756	0.363
	0.516	0.766	0.378

The Graph for Door (F1-score):

Construct the diagram we saw that system 1 and system 2 are decreasing, and our system is increasing.

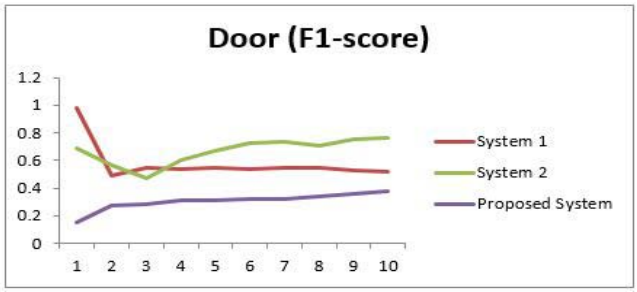

Fig. 20: Door (F1-score)

Table 19: Data collects for Door (G-mean)

Door			
	System 1	System 2	Proposed System
G-mean	0.939	1.194	0.885
	0.984	1.167	0.918
	1.045	1.132	0.939
	1.036	1.190	0.957
	1.052	1.215	0.959
	1.035	1.243	0.986
	1.045	1.233	0.989
	1.046	1.208	0.997
	1.035	1.242	1.008
	1.033	1.239	1.022

The Graph for Door (G-mean):

Our system's G-mean is elder than system 1 and system 2 because it is emergence from others.

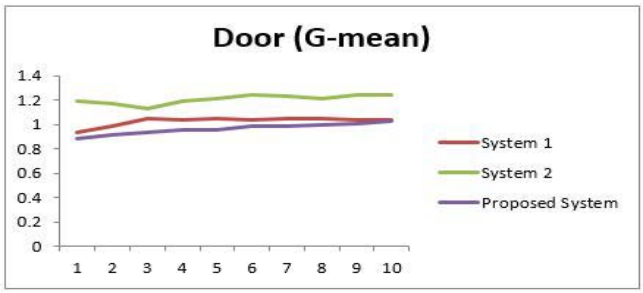

Fig. 21: Door (G-mean)

Comparison of Window:

Table 20: Data collects for Window (Sensitivity)

	Window		
	System 1	System 2	Proposed System
Sensitivity	0.507	0.318	0.150
	0.554	0.805	0.244
	0.601	0.818	0.254
	0.622	0.757	0.256
	0.622	0.781	0.283
	0.587	0.742	0.303
	0.584	0.763	0.317
	0.587	0.735	0.318
	0.584	0.753	0.322
	0.581	0.731	0.325

The Graph for Window (Sensitivity):

From the diagram, we saw that our proposed system sensitivity is good than other systems.

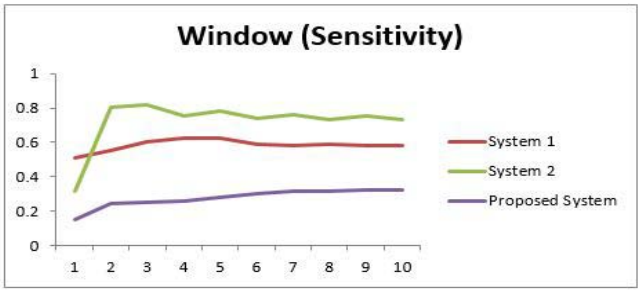

Fig. 22: Window (Sensitivity)

Table 21: Data collects for Window (Specificity)

	Window		
	System 1	System 2	Proposed System
Specificity	0.594	0.187	0.549
	0.647	0.035	0.570
	0.614	0.541	0.617
	0.636	0.499	0.621
	0.576	0.655	0.622
	0.607	0.615	0.649
	0.599	0.673	0.662
	0.611	0.645	0.677
	0.598	0.661	0.716
	0.141	0.639	0.969

The Graph for Window (Specificity):

For monitoring the effect of specificity, we collect data and compare them and choose the better line.

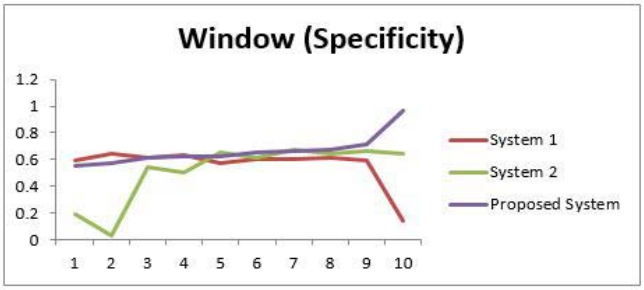

Fig. 23: Window (Specificity)

Table 22: Data collects for Window (Precision)

	Window		
	System 1	System 2	Proposed System
Precision	0.601	0.276	0.249
	0.650	0.690	0.258
	0.616	0.702	0.259
	0.638	0.666	0.288
	0.577	0.740	0.325
	0.608	0.706	0.334
	0.600	0.734	0.334
	0.612	0.708	0.334
	0.599	0.709	0.334
	0.607	0.689	0.339

The Graph for Window (Precision):

System 1 and 2 create sprawl line. Our system makes a straight line. It is favorable.

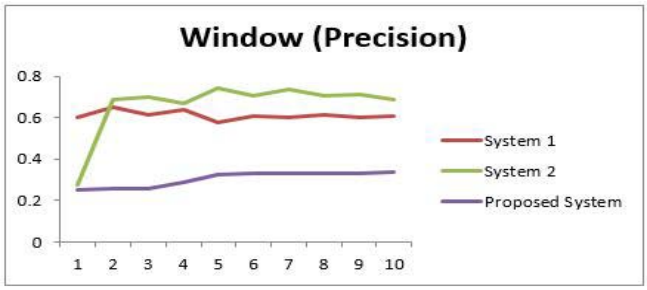

Fig. 24: Window (Precision)

Table 23: Data collects for Window (Precision)

	Window		
	System 1	System 2	Proposed System
Accuracy	0.209	0.252	0.468
	0.597	0.595	0.498
	0.607	0.699	0.502
	0.629	0.646	0.507
	0.598	0.725	0.519
	0.597	0.686	0.546
	0.591	0.722	0.547
	0.599	0.694	0.548
	0.591	0.709	0.585
	0.447	0.687	0.894

The Graph for Window (Accuracy):

Our system is heightening from system 1 and system 2 because it is a prosperous order line graph.

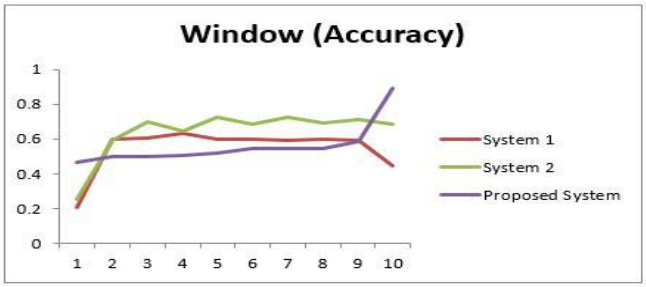

Fig. 25: Window (Accuracy)

Table 24: Data collects for Window (F1-score)

Window			
	System 1	System 2	Proposed System
F1-score	0.549	0.296	0.247
	0.598	0.743	0.254
	0.608	0.755	0.256
	0.629	0.709	0.257
	0.599	0.399	0.286
	0.596	0.723	0.318
	0.592	0.748	0.321
	0.599	0.721	0.325
	0.591	0.730	0.328
	0.594	0.709	0.329

The Graph for Window (F1-score):

From the table, we plot the value and choose a better system, and the implemented system is the best than others.

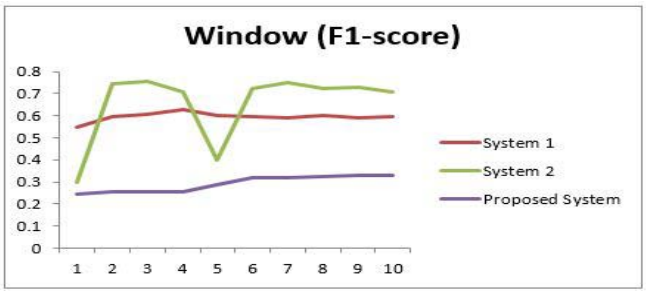

Fig. 26: Window (F1-score)

Table 25: Data collects for Window (G-mean)

Window			
	System 1	System 2	Proposed System
G-mean	1.049	0.711	0.935
	1.095	0.917	0.937
	1.102	1.166	0.942
	1.122	1.121	0.945
	1.095	1.198	0.951
	1.092	1.165	0.965
	1.088	1.198	0.969
	1.095	1.175	0.989
	1.087	1.189	1.009
	1.849	1.170	1.058

The Graph for Window (G-mean):

We are measuring and plotting the value for the diagram and observe that it is increasing.

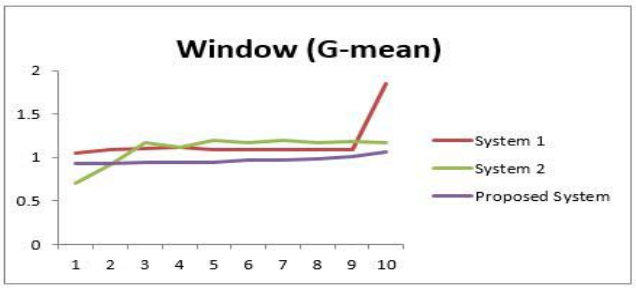

Fig. 27: Light (Sensitivity)

c) Future Work

In modern time almost, people prefer to do their work digitally, so they use the internet to do that work. Through the internet, we connect with those equipments easily which we used in our daily life for controlling that thing. With the help of IoT and the Internet, we can do those activity [24]. We can control and monitor our home equipment when the owner goes outside. For this purpose, this system needs to response in less time because if the materials turn on a long time, it will damage, or its cases more electric bill. We try to remove that problem in our paper. In future, we develop this system for consuming time on demand [25, 26].

V. Result

In this system, we calculate the time for seven equipment, and here we give four equipment's (light, fan, door, window) real-time in-home automation system. In our next paper, we will discuss and calculate the time for other three equipment. We passed data packet through every device for ten different ways. Collecting values and plotting those for creating a graph that helps us to view real-time response. We excrement for measuring a mediocre period and comparing with other existing systems. From the diagram, we see that our systems Sensitivity, Specificity, Precision, Accuracy, F1-score, and G-mean provide a better graph line than others. And we see that our systems response is better than another system.

VI. Conclusion

IoT promotes the proficiency of existence by joining the digital world to the real world via setting different sensors in daily life objects [27]. If we utilize our system correctly, we save our valuable time, energy; electric power moreover comforts of life. This work may be expanded to additional provision from other networks as those rules may provide better insight into the conduct [28].

References Références Referencias

1. J.A. Stankovic. Research Directions for the Internet of Things. IEEE Internet of Things Journal, Volume: 1, Issue: 1, Feb. 2014.
2. J. Gubbi, R. Buyya, S. Marusic, M. Palaniswami. Internet of Things (IoT): A vision, architectural

elements, and future directions. Future Generation Computer Systems, Volume 9, Issue 7, Pages 1645-1660, September 2013.
3. A Bhat, S Sharma, KR Pranav. HOME AUTOMATION USING INTERNET OF THINGS. International Research Journal of Engineering and Technology (IRJET) Volume: 04 Issue: 07, July - 2017.
4. C. Wang, Z. Bi and Li Da Xu. "IOT and Cloud Computing in Automation of Assembly Modeling Systems". IEEE Transactions on Industrial Informatics. Volume: 10, Issue: 2, May 2014.
5. E. Borgia. "The Internet of Things vision: Key features, applications and open issues". Computer Communications. Volume 54, 1 December 2014, Pages 1-31.
6. R. Piyare and S. R. Lee. "Smart Home-Control and Monitoring System using Smart Phone". ICCA, ASTL, 2013 - researchgate.net.
7. K. Juvva. "Real-Time Systems". https://users.ece.cmu.edu/~koopman/des_s99/real_time/ (Spring 1998). [Online; accessed 11-April-2018].
8. D. Pavithra and R. Balakrishnan. "IoT based monitoring and control system for home automation". 2015 Global Conference on Communication Technologies (GCCT). 23-24 April 2015.
9. V. A. Memos, K. E. Psannis, Y. Ishibashi, B. G. Kim and B.B Gupta. "An Efficient Algorithm for Media-based Surveillance System (EAMSuS) in IoT Smart City Framework". Future Generation Computer Systems. 25 April 2017.
10. M. Ashok, P. S. Varma and M. V. R. Sundari. "IoT based Monitoring and Control System for Home Automation". International Journal of Engineering Technology Science and Research (IJETSR). Volume 4, Issue 11, November 2017.
11. T. K. L. Hui, R. S. Sherratt, D. D. S´anchez. "Major Requirements for Building Smart Homes in Smart Cities based on Internet of Things Technologies". Future Generation Computer Systems, 76. pp. 58-369. ISSN 0167-739X. October 3, 2015.
12. "Real-time computing". https://www.pubnub.com/learn/glossary/what-is-real-time-computing/ (2010 - 2018 PubNub Inc). [Online; accessed 22-April-2018].
13. Mr. R. Kadam, Mr. P. Mahamuni and Mr. Parikh. "Smart Home System". International Journal of Innovative Research in Advanced Engineering (IJIRAE) ISSN: 2349-2163 Volume 2 Issue 1 (January 2015).
14. P. Patel, M. Patel, V. Panchal & V. Nirmal. Home Automation Using Internet of Things. Imperial Journal of Interdisciplinary Research (IJIR) Vol-2, Issue-5, 2016.
15. K. Mandula, R. Parupalli, CH. A. S. Murty, E. Magesh, R. Lunagariya. Mobile based home automation using Internet of Things (IoT). 2015 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT), 18-19 Dec. 2015.
16. R. Piyare. Internet of Things: Ubiquitous Home Control and Monitoring System using Android based Smart Phone. International Journal of Internet of Things, 2013.
17. R. Piyare and M. Tazil. "Bltooth Based Home Automation System Using Cell Phone". 2011 IEEE 15th International Symposium on Consumer Electronics. 14-17 June 2011.
18. D.J. Cook, M. Youngblood, E.O. Heierman, K. Gopalratnam, S. Rao, A. Litvin, F. Khawaja. MavHome: An Agent-Based Smart Home. Proceedings of the First IEEE International Conference on Pervasive Computing and Communications, 26 March 2003.
19. N. Hossain, M. A. Hossain, R. Sultana and F. A. Lima. "A Security Framework for IOT based Smart Home Automation System". Global Journal of Computer Science and Technology (GJCST) Volume 18, Issue 3, Version 1.0 (Jun. 2018), ISO 9001:2005 Certified Group.
20. M. Soliman T. Abiodun, T. Hamouda, J. Zhou and C. H. Lung. "Smart Home: Integrating Internet of Things with Web Services and Cloud Computing". 2013 IEEE 5th International Conference on Cloud Computing Technology and Science (2013), Bristol, United Kingdom United Kingdom Dec. 2, 2013 to Dec. 5, 2013.
21. Y. Liu. "Study on Smart Home System Based on Internet of Things Technology". In: Du W. (eds) Informatics and Management Science IV. Lecture Notes in Electrical Engineering, vol 207. Springer, London. 06 December 2012.
22. H. M. Raafat, M. S. Hossain, E. Essa, S. Elmougy, A. S. Tolba, G. Muhammad and A. Ghoneim. "Fog Intelligence for Real-Time IoT Sensor Data Analytics". IEEE Access (Volume: 5). 20 September 2017.
23. Y. Huang, L. Wang, W. Guo, Q. Kang, Q. Wu. "Chance Constrained Optimization in a Home Energy Management System". IEEE Transactions on Smart Grid. Volume: 9, Issue: 1, Jan. 2018.
24. P. Patel, M. Patel, V. Panchal and V. Nirmal. "Home Automation Using Internet of Things". Imperial Journal of Interdisciplinary Research (IJIR). Vol-2, Issue-5, 2016.
25. S. Soumya, M. Chavali, S. Gupta and N. Rao. "Internet of Things based Home Automation System". 2016 IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT). Bangalore, India, 20-21 May 2016.
26. M Chan, D Estève, C Escriba, E Campo. A review of smart homes-Present state and future challenges.

Computer Methods and Programs in Biomedicine, Volume 91, Issue 1, Pages 55-81, July 2008.
27. P. Kumar, C. Martani, L. Morawska, L. Norford, R. Choudhary, M.Bell, M. Leach. "Indoor air quality and energy management through real-time sensing in commercial buildings". Energy and Buildings. Volume 111, 1 January 2016, Pages 145-153.
28. C. Stergiou, K. E. Psannis, B. G. Kim and B. Gupta. "Secure integration of IoT and Cloud Computing". Future Generation Computer Systems. 1 December 2016.

Global Journal of Computer Science and Technology: B
Cloud and Distributed
Volume 19 Issue 1 Version 1.0 Year 2019
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals
Online ISSN: 0975-4172 | Print ISSN: 0975-4350 | DOI: 10.17406

A Taxonomy of Schedulers – Operating Systems, Clusters and Big Data Frameworks

By Leszek Sliwko

Abstract- This review analyzes deployed and actively used workload schedulers' solutions and presents a taxonomy in which those systems are divided into several hierarchical groups based on their architecture and design. While other taxonomies do exist, this review has focused on the key design factors that affect the throughput and scalability of a given solution, as well as the incremental improvements which bettered such an architecture. This review gives special attention to Google's Borg, which is one of the most advanced and published systems of this kind.

Keywords: *schedulers, workload, cluster, cloud, big data, borg.*

GJCST-B Classification : *I.2.8*

Strictly as per the compliance and regulations of:

A Taxonomy of Schedulers – Operating Systems, Clusters and Big Data Frameworks

Leszek Sliwko

Abstract- This review analyzes deployed and actively used workload schedulers' solutions and presents a taxonomy in which those systems are divided into several hierarchical groups based on their architecture and design. While other taxonomies do exist, this review has focused on the key design factors that affect the throughput and scalability of a given solution, as well as the incremental improvements which bettered such an architecture. This review gives special attention to Google's Borg, which is one of the most advanced and published systems of this kind.

Keywords: schedulers, workload, cluster, cloud, big data, borg.

I. Taxonomy of Schedulers

Although managing workload in a Cloud system is a modern challenge, scheduling strategies are a well-researched field as well as being an area where there has been considerable practical implementation. This background review started by analyzing deployed and actively used solutions and presents a taxonomy in which schedulers are divided into several hierarchical groups based on their architecture and design. While other taxonomies do exist (e.g., Krauter et al., 2002; Yu and Buyya, 2005; Pop et al., 2006; Smanchat and Viriyapant, 2015; Rodriguez and Buyya, 2017; Zakarya and Gillam, 2017;Tyagi and Gupta, 2018), this review has focused on the most important design factors that affect the throughput and scalability of a given solution, as well as the incremental improvements which bettered such an architecture.

Figure 1 visualizes how the schedulers' groups are split. The sections which follow discusses each of these groups separately.

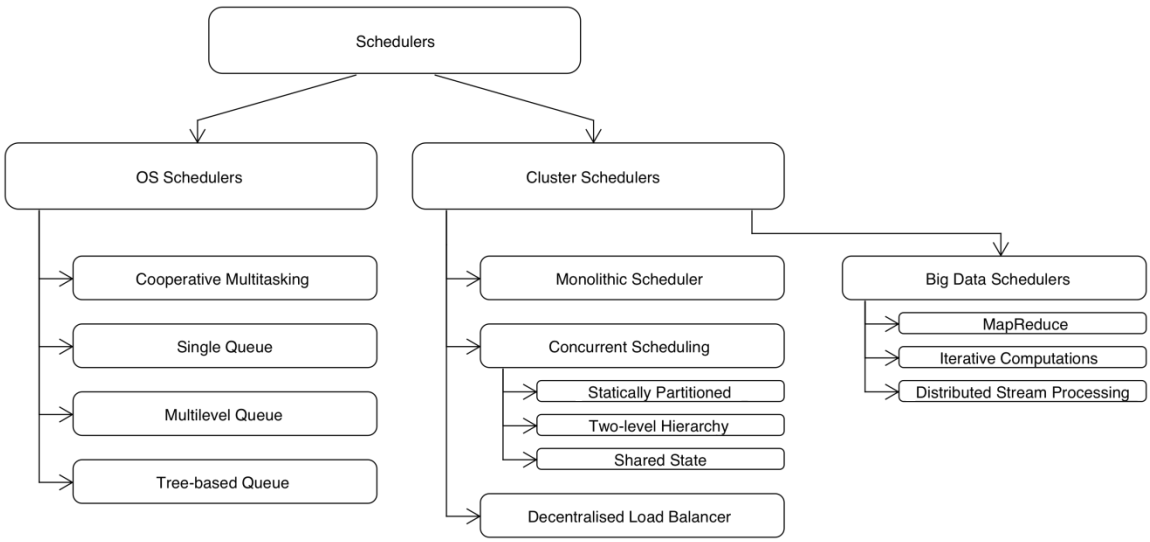

Figure 1: Schedulers taxonomy

II. Metacomputing

The concept of connecting computing resources has been an active area of research for some time. The term 'metacomputing' was established as early as 1987 (Smarr and Catlett, 2003) and since then the topic of scheduling has been the focus of many research projects, such as (i) service localizing idle workstations and utilizing their spare CPU cycles – HTCondor (Litzkow et al., 1988); (ii) the Mentat – a parallel run-time system developed at the University of Virginia (Grimshaw, 1990); (iii) blueprints for a national supercomputer (Grimshaw et al., 1994), and (iv) the Globus metacomputing infrastructure toolkit (Foster and Kesselman, 1997).

Before the work of Foster et al. (2001), there was no clear definition to what 'grid' systems referred. Following this publication, the principle that grid systems should allow a set of participants to share several connected computer machines and their resources became established. A list of rules defines these shared system policies. This includes which resources are

Author: Axis Applications Ltd, London, Uk. e-mail: Lsliwko@gmail.com

being shared, who is sharing these resources, the extent to which they can use those resources, and what quality of service they can expect.

As shown in the following sections, the requirements of a load balancer in a decentralized system varies significantly compared to scheduling jobs on a single machine (Hamscher et al., 2000). One significant difference is the network resources, in that transferring data between machines is expensive because the nodes tend to be geographically distributed. In addition to the high-impact spreading of tasks across networked machines, the load balancer in Clusters generally provides a mechanism for fault-tolerance and user session management. The sections below also explain the workings of several selected current and historical schedulers and distributed frameworks. If we can understand these, we will know more about how scheduling algorithms developed over time, as well as the different ways they have been conceptualized. This paper does not purport to be a complete taxonomy of all available designs, but rather presents an analysis of some of the most important concepts and aspects of the history of schedulers.

III. OS Schedulers

The Operating System (OS) Scheduler, also known as a 'short-term scheduler' or 'CPU scheduler', works within very short time frames, i.e., time-slices. During scheduling events, an algorithm must examine planned tasks and assign them appropriate CPU times (Bulpin, 2005; Arpaci-Dusseau and Arpaci-Dusseau, 2015). This setting requires schedulers to use highly optimized algorithms with very small overheads. Process schedulers face the challenge of how to maintain the balance between throughput and responsiveness (i.e., minimum latency). Prioritizing the execution of processes with a higher sleep/processing ratio is the way this is generally achieved (Pabla, 2009).
At present, the most advanced strategies also take into consideration the latest CPU core where the process ran the previous time, which is known as 'Non-Uniform Memory Access (NUMA) awareness'. The aim is to reuse the same CPU cache memory wherever possible (Blagodurov et al., 2010). The memory access latency differences can be very substantial, for example ca. 3-4 cycles for L1 cache, ca. 6-10 cycles for L2 cache and ca. 40-100 cycles for L3 cache (Drepper, 2007). NUMA awareness also involves prioritizing the act of choosing a real idle core which must occur before its logical SMT sibling, also known as 'Hyper-Threading (HT) awareness'. Given this, NUMA awareness is a crucial element in the design of modern OS schedulers. With a relatively high data load to examine in a short period, implementation needs to be strongly optimized to ensure faster execution.

OS Schedulers tend to provide only a very limited set of configurable parameters, wherein the access to modify them is not straightforward. Some of the parameters can change only during the kernel compilation process and require rebooting, such as compile-time options CONFIG_FAIR_USER_SCHED and CONFIG_FAIR_CGROUP_SCHED, or on the fly using the low-level Linux kernel's tool 'sysctl'.

a) *Cooperative Multitasking*

Early multitasking Operating Systems, such as Windows 3.1x, Windows 95, 96 and Me, Mac OS before X, adopted a concept known as Cooperative Multitasking or Cooperative Scheduling (CS). In early implementations of CS, applications voluntarily ceded CPU time to one another. This was later supported natively by the OS, although Windows 3.1x used a non-pre-emptive scheduler which did not interrupt the program, wherein the program needed to explicitly tell the system that it no longer required the processor time. Windows 95 introduced a rudimentary pre-emptive scheduler, although this was for 32-bit applications only (Hart, 1997). The main issue in CS is the hazard caused by the poorly designed program. CS relies on processes regularly giving up control to other processes in the system, meaning that if one process consumes all the available CPU power then all the systems will hang.

b) *Single Queue*

Before Linux kernel version 2.4, the simple Circular Queue (CQ) algorithm was used to support the execution of multiple processes on the available CPUs. A Round Robin policy informed the next process run (Shreedhar, 1995). In kernel version 2.2, processes were further split into non-real/real-time categories, and scheduling classes were introduced. This algorithm was replaced by O(n) scheduler in Linux kernel versions 2.4-2.6. In O(n), processor time is divided into epochs, and within each epoch every task can execute up to its allocated time slice before being pre-empted. At the beginning of each epoch, the time slice is given to each task; it is based on the task's static priority added to half of any remaining time-slices from the previous epoch (Bulpin, 2005). Thus, if a task does not use its entire time slice in the current epoch, it can execute for longer in the next one. During a scheduling event, an O(n) scheduler requires iteration through all the process which are currently planned (Jones, 2009), which can be seen as a weakness, especially for multi-core processors.

Between Linux kernel versions 2.6-2.6.23 came the implementation of the O(1) scheduler. O(1) further splits the processes list into active and expired arrays. Here, the arrays are switched once all the processes from the active array have exhausted their allocated time and have been moved to the expired array. The O(1) algorithm analyses the average sleep time of the process, with more interactive tasks being given higher

priority to boost system responsiveness. The calculations required are complex and subject to potential errors, where O(1) may cause non-interactive behavior from an interactive process (Wong et al., 2008; Pabla, 2009).

c) Multilevel Queue

With Q(n) and O(1) algorithms failing to efficiently support the applications' interactivity, the design of OS Scheduler evolved into a multilevel queue. In this queue, repeatedly sleeping (interactive) processes are pushed to the top and executed more frequently. Simultaneously, background processes are pushed down and run less frequently, although for extended periods.

Perhaps the most widespread scheduler algorithm is Multilevel Feedback Queue (MLFQ), which is implemented in all modern versions of Windows NT (2000, XP, Vista, 7 and Server), Mac OS X, NetBSD and Solaris kernels (up to version 2.6, when it was replaced with O(n) scheduler). MLFQ was first described in 1962 in a system known as the Compatible Time-Sharing System (Corbató et al., 1962). Fernando Corbató was awarded the Turing Award by the ACM in 1990 'for his pioneering work organizing the concepts and leading the development of the general-purpose, large-scale, time-sharing and resource-sharing computer systems, CTSS and Multics'. MLFQ organizes jobs into a set of queues Q_0, Q_1, …, Q_i wherein a job is promoted to a higher queue if it does not finish within 2^i time units. The algorithm always processes the job from the front of the lowest queue, meaning that short processes have preference. Although it has a very poor worst-case scenario, MLFQ turns out to be very efficient in practice (Becchetti et al., 2006).

Staircase Scheduler (Corbet, 2004), Staircase Deadline Scheduler (Corbet, 2007), Brain F. Scheduler (Groves et al., 2009) and Multiple Queue Skiplist Scheduler (Kolivas, 2016) constitute a line of successive schedulers developed by Con Kolivas since 2004 which are based on a design of Fair Share Scheduler from Kay and Lauder (1988). None of these schedulers have been merged into the source code of mainstream kernels. They are available only as experimental '-ck' patches. Although the concept behind those schedulers is similar to MLFQ, the implementation details differ significantly. The central element is a single, ranked array of processes for each CPU ('staircase'). Initially, each process (P1, P2, …) is inserted at the rank determined by its base priority; the scheduler then picks up the highest ranked process (P) and runs it. When P has used up its time slice, it is reinserted into the array but at a lower rank, where it will continue to run but at a lower priority. When P exhausts its next time-slice, its rank is lowered again. P then continues until it reaches the bottom of the staircase, at which point it is moved up to one rank below its previous maximum and is assigned two time-slices. When P exhausts these two time-slices, it is reinserted once again in the staircase at a lower rank. When P once again reaches the bottom of the staircase, it is assigned another time-slice and the cycle repeats. P is also pushed back up the staircase if it sleeps for a predefined period. The result of this is that that interactive tasks which tend to sleep more often should remain at the top of the staircase, while CPU-intensive processes should continuously expend more time-slices but at a lower frequency. Additionally, each rank level in the staircase has its quota, and once the quota is expired all processes on that rank are pushed down.

Most importantly, Kolivas' work introduced the concept of 'fairness'. What this means is that each process gets a comparable share of CPU time to run, proportional to the priority. If the process spends much of its time waiting for I/O events, then its spent CPU time value is low, meaning that it is automatically prioritized for execution. When this happens, interactive tasks which spend most of their time waiting for user input get execution time when they need it, which is how the term 'sleeper fairness' derives. This design also prevents a situation in which the process is 'starved', i.e., never executed.

d) Tree-Based Queue

While the work of Con Kolivas has never been merged into the mainstream Linux kernel, it has introduced the central concept of 'fairness', which is the crucial feature of the design of most current OS schedulers. At the time of writing, Linux kernel implements Completely Fair Scheduler (CFS), which was developed by Ingo Molnár and introduced in kernel version 2.6.23. A central element in this algorithm is a self-balancing red-black tree structure in which processes are indexed by spent processor time. CFS implements the Weighted Fair Queueing (WFQ) algorithm, in which the available CPU time-slices are split between processes in proportion to their priority weights ('niceness'). WFQ is based on the idea of the 'ideal processor', which means that each process should have an equal share of CPU time adjusted for their priority and total CPU load (Jones, 2009; Pabla, 2009).

Lozi et al. (2016) presents an in-depth explanation of the algorithm's workings, noting potential issues regarding the CFS approach. The main criticism revolves around the four problematic areas:

- Group Imbalance – The authors' experiments have shown that not every core of their 64-core machine is equally loaded: some cores run only one process or sometimes no processes at all, while the rest of the cores were overloaded. It seems that the scheduler was not balancing the load because of the hierarchical design and complexity of the load tracking metric. To limit the complexity of the

scheduling algorithm, the CPU cores are grouped into scheduling groups, i.e., nodes. When an idle core attempts to steal work from another node, it compares only the average load of its node with that of its victim's node. It will steal work only if the average load of its victim's group is higher than its own. The result is inefficiency since idle cores will be concealed by their nodes' average load.

- Scheduling Group Construction – This concern relates to the way scheduling groups are constructed which is not adapted to modern NUMA machines. Applications in Linux can be pinned to a subset of available cores. CFS might assign the same cores to multiple scheduling groups with those groups then being ranked by distance. This could be nodes one hop apart, two hops apart and so on. This feature was designed to increase the probability that processes would remain close to their original NUMA node. However, this could result in the application being pinned to particular cores which are separated by more than one hop, with work never being migrated outside the initial core. This might mean that an application uses only one core.
- Overload-on-Wakeup – This problem occurs when a process goes to sleep on a particular node and is then awoken by a process on the same node. In such a scenario, only cores in this scheduling group will be considered to run this process. The aim of this optimization is to improve cache utilization by running a process close to the waker process, meaning that there is the possibility of them sharing the last-level memory cache. However, the might be the scheduling of a process on a busy core when there are idle cores in alternative nodes, resulting in the severe underutilization of the machine.
- Missing Scheduling Domains – This is the result of a line of code omission while refactoring the Linux kernel source code. The number of scheduling domains is incorrectly updated when a particular code is disabled and then enabled, and a loop exits early. As a result, processes can be run only on the same scheduling group as their parent process.

Lozi et al. (2016) have provided a set of patches for the above issues and have presented experimental results after applying fixes. They have also made available a set of tools on their site which could be used to detect those glitches early in the Linux kernel lifecycle. Moreover, it has been argued that the sheer number of optimizations and modifications implemented into CFS scheduler changed the initially simple scheduling policy into one which was very complex and bug-prone. As of 12thFebruary 2019, there had been780 commits to CFS source code ('fair.c' file in github.com/torvalds/linux repository) since November 2011. As such, an alternative approach is perhaps required, such as a scheduler architecture based on pluggable components. This work demonstrates the immerse complexity of scheduling solutions catering to the complexities of modern hardware.

IV. Cluster Schedulers

There are many differences between distributed computing and traditional computing. For example, the physical size of the system means that there may be thousands of machines involved, with thousands of users being served and millions of API calls or other requests needing processing. While responsiveness and low overheads are often the focus of process schedulers, the focus of cluster schedulers is to focus upon high throughput, fault-tolerance, and scalability. Cluster schedulers usually work with queues of jobs spanning to hundreds of thousands, and indeed sometimes even millions of jobs. They also seem to be more customized and tailored to the needs of the organization which is using them.

Cluster schedulers often provide complex administration tools with a wide spectrum of configurable parameters and flexible workload policies. All configurable parameters can generally be accessed via configuration files or the GUI interface. However, it appears that site administrators seldom stray from a default configuration (Etsion and Tsafrir, 2005). The most used scheduling algorithm is simply a First-Come-First-Serve (FCFS) strategy with backfilling optimization.

The most common issues which cluster schedulers must deal with are:

- Unpredictable and varying load (Moreno et al., 2013);
- Mixed batch jobs and services (ibid.);
- Complex policies and constraints (Adaptive Computing, 2002);
- Fairness (ibid.);
- A rapidly increasing workload and cluster size (Isard et al., 2007);
- Legacy software (ibid.);
- Heterogeneous nodes with a varying level of resources and availability (Thain et al., 2005);
- The detection of underperforming nodes (Zhang et al., 2014);
- Issues related to fault-tolerance (ibid.) and hardware malfunctions (Gabriel et al., 2004).

Another challenge, although one which is rarely tackled by commercial schedulers, is minimizing total power consumption. Typically, idle machines consume around half of their peak power (McCullough et al., 2011). Therefore, a Data Center can decrease the total power it consumes by concentrating tasks on fewer machines and powering down the remaining nodes (Pinheiro et al., 2001; Lang and Patel, 2010).

The proposed grouping of Cluster schedulers loosely follows the taxonomy presented in Schwarzkopf et al. (2013).

a) Monolithic Scheduler

The earliest Cluster schedulers had a centralized architecture in which a single scheduling policy allocated all incoming jobs. The tasks would be picked from the head of the queue and scheduled on system nodes in a serial manner by an allocation loop. Examples of centralized schedulers include Maui (Jackson et al., 2001) and its successor Moab (Adaptive Computing, 2015), Univa Grid Engine (Gentzsch, 2001), Load Leveler (Kannan et al., 2001), Load Sharing Facility (Etsion and Tsafrir, 2005), Portable Batch System (Bode et al., 2000) and its successor TORQUE (Klusáček et al., 2013), Alibaba's Fuxi (Zhang et al., 2014), Docker Swarm (Naik, 2016), Kubernetes (Vohra, 2017) and several others.

Monolithic schedulers implement a wide array of policies and algorithms, such as FCFS, FCFS with backfilling and gang scheduling, Shortest Job First (SJF), and several others. The Kubernetes (Greek: 'κυβερνήτης') scheduler implements a range of scoring functions such as node or pod affinity/anti-affinity, resources best-fit and worst-fit, required images locality, etc. which can be additionally weighted and combined into node's score values (Lewis and Oppenheimer, 2017). As an interesting note – one of the functions (Balanced Resource Allocation routine) implemented in Kubernetes evaluates the balance of utilized resources (CPU and memory) on a scored node.

Monolithic schedulers often face a 'head-of-queue' blocking problem, in which shorter jobs are held when a long job is waiting for a free node. To try and counter this problem, the schedulers often implement 'backfilling' optimization, where shorter jobs are allowed to execute while the long job is waiting. Perhaps the most widespread scheduler is Simple Linux Utility for Resource Management (SLURM)(Yoo et al., 2003). SLURM uses a best-fit algorithm which is based on either Hilbert curve scheduling or fat tree network topology; it can scale to thousands of CPU cores (Pascual, 2009). At the time of writing, the fastest supercomputer in the world is Sunway TaihuLight (Chinese: '神威·太湖之光'), which uses over 40k CPU processors, each of which contains 256 cores. Sunway TaihuLight's workload in managed by SLURM (TOP500 Project, 2017).

The Fuxi (Chinese: '伏羲') scheduler presents a unique strategy in that it matches newly-available resources against the backlog of tasks rather than matching tasks to available resources on nodes. This technique allowed Fuxi to achieve very high utilization of Cluster resources, namely 95% utilization of memory and 91% utilization of CPU. Fuxi has been supporting Alibaba's workload since 2009, and it scales to ca. 5k nodes (Zhang et al., 2014).

While Cluster scheduler designs have generally moved towards solutions which are more parallel, as demonstrated in the next subsection, centralized architecture is still the most common approach in High-Performance Computing. Approximately half the world's supercomputers use SLURM as their workload manager, while Moab is currently deployed on about 40% of the top 10, top 25 and top 100 on the TOP500 list (TOP500 Project, 2017).

b) Concurrent Scheduling

Historically, monolithic schedulers were frequently built on the premise of supporting a single 'killer-application' (Barroso et al., 2003). However, the workload of the data center has become more heterogeneous as systems and a modern Cluster system runs hundreds of unique programs with distinctive resource requirements and constraints. A single code base of centralized workload manager means that it is not easy to add a variety of specialized scheduling policies. Furthermore, as workload size is increased, the time to reach a scheduling decision is progressively limited. The result of this is a restriction in the selection of scheduling algorithms to less sophisticated ones, which affects the quality of allocations. To tackle those challenges, the Cluster schedulers developed designs which are more parallel.

i. *Statically Partitioned*

The solution to the numerous policies and the lack of parallelism in central schedulers was to split Cluster into specialized partitions and manage them separately. Quincy (Isard et al., 2009), a scheduler managing workload of Microsoft's Dryad, follows this approach.

The development of an application for Dryad is modeled as a Directed Acyclic Graph (DAG) model in which the developer defines an application dataflow model and supplies subroutines to be executed at specified graph vertices. The scheduling policies and tuning parameters are specified by adjusting weights and capacities on a graph data structure. The Quincy implements a Greedy strategy. In this approach, the scheduler assumes that the currently scheduled job is the only job running on a cluster and so always selects the best node available. Tasks are run by remote daemon services. From time to time these services update the job manager about the execution status of the vertex, which in the case of failure might be re-executed. Should any task fail more than a configured number of times, the entire job is marked as failed (Isard et al., 2007).

Microsoft has built several frameworks on top of Dryad, such as COSMOS (Helland and Harris, 2011) which provided SQL-like language optimized for parallel execution. COSMOS was designed to support data-

driven search and advertising within the Windows Live services owned by Microsoft, such as Bing, MSN, and Hotmail. It analyzed user behaviors in multiple contexts, such as what people searched for, what links they clicked, what sites they visited, the browsing order, and the ads they clicked on. Although the Dryad project had several preview releases, it was eventually dropped when Microsoft shifted its focus to the development of Hadoop.

The main criticism of the static partitioning is inflexibility, that is, the exclusive sets of machines in a Cluster are dedicated to certain types of workload. That might result in a part of scheduler being relatively idle, while other nodes are very active. This issue leads to the Cluster's fragmentation and the suboptimal utilization of available nodes since no machine sharing is allowed.

ii. *Two-Level Hierarchy*

The solution to the inflexibility of static partitioning was to introduce two-level architecture in which a Cluster is partitioned dynamically by a central coordinator. The actual task allocations take place at the second level of architecture in one of the specialized schedulers. The first two-level scheduler was Mesos (Hindman et al., 2011). It was developed at the University of California (Berkeley) and is now hosted in the Apache Software Foundation. Mesos was a foundation base for other Cluster systems such as Twitter's Aurora (Aurora, 2018) and Marathon (Mesosphere, 2018).

Mesos introduces a two-level scheduling mechanism in which a centralized Mesos Masteracts as a resource manager. It dynamically allocates resources to different scheduler frameworks via Mesos Agents, e.g., Hadoop, Spark and Kafka. Mesos Agents are deployed on cluster nodes and use Linux's cgroups or Docker container (depending upon the environment) for resource isolation. Resources are distributed to the frameworks in the form of 'offers' which contain currently unused resources. Scheduling frameworks have autonomy in deciding which resources to accept and which tasks to run on them.

Mesos is most effective when tasks are relatively small, short-lived and have a high resource churn rate, i.e., they relinquish resources more frequently. In the current version (1.4.1), only one scheduling framework can examine a resource offer at any given time. This resource is effectively locked for the duration of a scheduling decision, meaning that concurrency control is pessimistic. Campbell (2017) presents several practical considerations for using Mesos in the production environment, in addition to advice on best practice.

Two-level schedulers offered a working solution to the lack of parallelization found in central schedulers and the low efficiency of statically partitioned Clusters. Nevertheless, the mechanism used causes resources to remain locked at the same time a specialized scheduled examines the resources offer. This means the benefits from parallelization are limited due to pessimistic locking. Furthermore, the schedulers do not coordinate with each other and must rely on a centralized coordinator to make them offers. This further restricts their visibility of the resources in a Cluster.

iii. *Shared State*

To address the limited parallelism of the two-level scheduling design, the alternative approach taken by some organizations was to redesign schedulers' architecture into several schedulers, all working concurrently. The schedulers work on a shared Cluster's state information and manage their resources' reservations using an optimistic concurrency control method. A sample of such systems includes: Microsoft's Apollo (Boutin et al., 2014); Omega, Google Borg's spinoff (Schwarzkopf et al., 2013); HashiCorp's Nomad (HashiCorp, 2018); and also Borg (Burns et al., 2016) itself. The latter system has been refactored from monolithic into parallel architecture after experimentations with Omega.

By default, Nomad runs one scheduling worker per CPU core. Scheduling workers pick job submissions from the broker queue and then submit it to one of the three schedulers: a long-lived services scheduler, a short-lived batch jobs scheduler and a system scheduler, which is used to run internal maintenance routines. Additionally, Nomad can be extended to support custom schedulers. Schedulers process and generate an action plan, which constitutes a set of operations to create new allocations, or to evict and update existing ones (HashiCorp, 2018).

Microsoft's Apollo design seems to be primarily tuned for high tasks churn, and at peak times is capable of handling more than 100k of scheduling requests per second on a ca. 20k nodes cluster. Apollo uses a set of per-job schedulers called Job Managers (JM) wherein a single job entity contains a multiplicity of tasks which are then scheduled and executed on computing nodes. Tasks are generally short-lived batch jobs (Boutin et al., 2014). Apollo has a centralized Resource Monitor (RM), while each node runs its Process Node (PN) with a queue of tasks. Each PN is responsible for local scheduling decisions and can independently reorder its job queue to allow smaller tasks to be executed immediately, while larger tasks wait for resources to become available. In addition, PN computes a wait-time matrix based on its queue which publicizes the future availability of the node's resources. Scheduling decisions are made optimistically by JMs based on the shared cluster's resource state, which is continuously retrieved and aggregated by RM.

Furthermore, Apollo categorizes tasks as 'regular' and 'opportunistic'. Opportunistic tasks are used to fill resource gaps left by regular tasks. The

system also prevents overloading the cluster by limiting the total number of regular tasks that can be run on a cluster. Apollo implements locality optimization by taking into consideration the location of data for a given task. For example, the system will score nodes higher if the required files are already on the local drive as opposed to machines needing to download data (Boutin et al., 2014).

Historically, Omega was a spinoff from Google's Borg scheduler. Despite the various optimizations acquired by Borg over the years, including internal parallelism and multi-threading, to address the issues of head-of-line blocking and scalability problems, Google decided to create an Omega scheduler from the ground up (Schwarzkopf et al., 2013). Omega introduced several innovations, such as storing the state of the cluster in a centralized Paxos-based store that was accessed by multiple components simultaneously. Optimistic locking concurrency control resolved the conflicts which emerged. This feature allowed Omega to run several schedulers at the same time and improve the throughput. Many of Omega's innovations have since been folded into Borg (Burns et al., 2016).

Omega's authors highlight the disadvantages of the shared state and parallel reservation of resources, namely: (i) the state of a node could have changed considerably when the allocation decision was being made, and it is no longer possible for this node to accept a job; (ii) two or more allocations to the same node could have conflicted and both scheduling decisions are nullified; and (iii) this strategy introduces significant difficulties when gang-scheduling a batch of jobs as (i) or (ii) are happening (Schwarzkopf et al., 2013).

In this research, Google's Borg received special attention, as one of the most advanced and published schedulers. Moreover, while other schedulers are designed to support either a high churn of short-term jobs, e.g., Microsoft's Apollo (Boutin et al., 2014), Alibaba's Fuxi (Zhang et al., 2014), or else a limited number of long-term services, such as Twitter's Aurora (Aurora, 2018), Google's engineers have created a system which supports a mixed workload. Borg has replaced two previous systems, Babysitter and the Global Work Queue, which were used to manage long-running services and batch jobs separately (Burns et al., 2016). Given the significance of Borg's design for this research, it is discussed separately in section 2.4.

iv. *Decentralised Load Balancer*

The research (Sliwko, 2018) proposes a new type of Cluster's workload orchestration model in which the actual scheduling logic is processed on nodes themselves. This is a significant step towards completely decentralized Cluster orchestration. The cluster state is retrieved from a subnetwork of BAs, although this system does not rely on the accuracy of this information and uses it exclusively to retrieve an initial set of candidate nodes where a task could potentially run. The actual task to machine matching is performed between the nodes themselves. As such, this design avoids the pitfalls of the concurrent resource locking, which includes conflicting scheduling decisions and the non-current state of nodes' information. Moreover, the decentralization of the scheduling logic also lifts complexity restrictions on scheduling logic, meaning that a wider range of scheduling algorithms can be used, such as metaheuristic methods.

c) Big Data Schedulers

In taxonomy presented in this paper, Big Data schedulers are visualized as a separate branch from Cluster Schedulers. Although Big Data Schedulers seem to belong to one of the Cluster schedulers designs discussed previously, this separation signifies their over-specialization, and that only a very restricted set of operations is supported (Isard et al., 2007; Zaharia et al., 2010). The scheduling mechanisms are often intertwined with the programming language features, with Big Data frameworks often providing their own API (Zaharia et al., 2009; White, 2012) and indeed sometimes even their own custom programming language, as seen with Skywriting in CIEL (Murray et al., 2011).

Generally speaking, Big Data frameworks provide very fine-grained control over how data is accessed and processed over the cluster, such as Spark RDD objects persist operations or partitioners (Zaharia et al., 2012). Such a deep integration of scheduling logic with applications is a distinctive feature of Big Data technology. At the time of writing, Big Data is also the most active distributed computing research area, with new technologies, frameworks and algorithms being released regularly.

Big Data is the term which describes the storage and processing of any data sets so large and complex that they become unrealistic to process using traditional data processing applications based on relational database management systems. It depends on the individual organization as to how much data is described as Big Data. The following examples provide an idea of scale:

- The NYSE (The New York Stock Exchange) produces about 15 TB of new trade data per day (Singh, 2017);
- Facebook warehouse stores upwards of 300 PB of data, with an incoming daily rate of about 600 TB (Vagata and Wilfong, 2014);
- The Large Hadron Collider (Geneva, Switzerland) produces about fifteen petabytes of data per year (White, 2012).

As a result of a massive size of the stored and processed data, the central element of a Big Data framework is its distributed file system, such as Hadoop

Distributed File System (Gog, 2012), Google File System (Ghemawat et al., 2003) and its successor Colossus (Corbett et al., 2013). The nodes in a Big Data cluster fulfill the dual purposes of storing the distributed file system parts, usually in a few replicas for fault-tolerance means, and also providing a parallel execution environment for system tasks. The speed difference between locally-accessed and remotely stored input data is very substantial, meaning that Big Data schedulers are very focused on providing 'data locality', which means running a given task on a node where input data are stored or are in the closest proximity to it.

The origins of the Big Data technology are in the 'MapReduce' programming model, which implements the concept of Google's inverted search index. Developed in 2003 (Dean and Ghemawat, 2010) and later patented in 2010 (U.S. Patent 7,650,331), the Big Data design has evolved significantly in the years since. It is presented in the subsections below.

i. *Mapreduce*

MapReduce is the most widespread principle which has been adopted for processing large sets of data in parallel. Originally, the name MapReduce only referred to Google's proprietary technology, but the term is now broadly used to describe a wide range of software, such as Hadoop, CouchDB, Infinispan, and MongoDB. The most important features of MapReduce are its scalability and fine-grained fault-tolerance. The 'map' and 'reduce' operations present in Lisp and other functional programming languages inspired the original thinking behind MapReduce (Dean and Ghemawat, 2010):

- 'Map' is an operation used in the first step of computation and is applied to all available data that performs the filtering and transforming of all key-value pairs from the input data set. The 'map' operation is executed in parallel on multiple machines on a distributed file system. Each 'map' task can be restarted individually, and a failure in the middle of a multi-hour execution does not require restarting the whole job from scratch.
- The 'Reduce' operation is executed after the 'map' operations complete. It performs finalizing operations, such as counting the number of rows matching specified conditions and yielding fields frequencies. The 'Reduce' operation is fed using a stream iterator, thereby allowing the framework to process the list of items one at the time, thus ensuring that the machine memory is not overloaded (Dean and Ghemawat, 2010; Gog, 2012).

Following the development of the MapReduce concept, Yahoo! engineers began the Open Source project Hadoop. In February 2008, Yahoo! announced that its production search index was being generated by a 10k-core Hadoop cluster (White, 2012). Subsequently, many other major Internet companies, including Facebook, LinkedIn, Amazon and Last.fm, joined the project and deployed it within their architectures. Hadoop is currently hosted in the Apache Software Foundation as an Open Source project.

As in Google's original MapReduce, Hadoop's users submit jobs which consist of 'map' and 'reduce' operation implementations. Hadoop splits each job into multiple 'map' and 'reduce' tasks. These tasks subsequently process each block of input data, typically 64MB or 128MB (Gog, 2012). Hadoop's scheduler allocates a 'map' task to the closest possible node to the input data required – so-called 'data locality' optimization. In so doing, we can see the following allocation order: the same node, the same rack and finally a remote rack (Zaharia et al., 2009). To further improve performance, the Hadoop framework uses 'backup tasks' in which a speculative copy of a task is run on a separate machine. The purpose of this is to finish the computation more quickly. If the first node is available but behaving poorly, it is known as a 'straggler', with the result that the job is as slow as the misbehaving task. This behavior can occur for many reasons, such as faulty hardware or misconfiguration. Google estimated that using 'backup tasks' could improve job response times by 44% (Dean and Ghemawat, 2010).

At the time of writing, Hadoop comes with a selection of schedulers, as outlined below:

- 'FIFO Scheduler' is a default scheduling system in which the user jobs are scheduled using a queue with five priority levels. Typically, jobs use the whole cluster, so they must wait their turn. When another job scheduler chooses the next job to run, it selects jobs with the highest priority, resulting in low-priority jobs being endlessly delayed (Zaharia et al., 2009; White, 2012).
- 'Fair Scheduler' is part of the cluster management technology Yet Another Resource Negotiator (YARN) (Vavilapalli et al., 2013), which replaced the original Hadoop engine in 2012. In Fair Scheduler, each user has their own pool of jobs, and the system focuses on giving each user a proportional share of cluster resources over time. The scheduler uses a version of 'max-min fairness' (Bonald et al., 2006) with minimum capacity guarantees that are specified as the number of 'map' and 'reduce' task slots to allocate tasks across users' job pools. When one pool is idle, and the minimum share of the tasks slots is not being used, other pools can use its available task slots.
- 'Capacity Scheduler' is the second scheduler introduced within the YARN framework. Essentially, this scheduled is a number of separate MapReduce engines, which contains FCFS scheduling for each user or organization. Those queues can be

hierarchical, with a queue having children queues, and with each queue being allocated task slots capacity which can be divided into 'map' and 'reduce' tasks. Task slots allocation between queues is similar to the sharing mechanism between pools found in Fair Scheduler (White, 2012).

The main criticism of MapReduce is the acyclic dataflow programming model. The stateless 'map' task must be followed by a stateless 'reduce' task, which is then executed by the MapReduce engine. This model makes it challenging to repeatedly access the same dataset, a common action during the execution of iterative algorithms (Zaharia et al., 2009).

ii. *Iterative Computations*

Following the success of Apache Hadoop, several alternative designs were created to address Hadoop's suboptimal performance when running iterative MapReduce jobs. Examples of such systems include HaLoop (Bu et al., 2010) and Spark (Zaharia et al., 2010).

HaLoop has been developed on top of Hadoop, with various caching mechanisms and optimizations added. This makes the framework loop-aware, for example by adding programming support for iterative application and storing the output data on the local disk. Additionally, HaLoop's scheduler keeps a record of every data block processed by each task on physical machines. It considers inter-iteration locality when scheduling tasks which follow. This feature helps to minimize costly remote data retrieval, meaning that tasks can use data cached on a local machine (Bu et al., 2010; Gog, 2012).

Similar to HaLoop, Spark's authors noted a suboptimal performance of iterative MapReduce jobs in the Hadoop framework. In certain kinds of application, such as iterative Machine Learning algorithms and interactive data analysis tools, the same data are repeatedly accessed in multiple steps and then discarded; therefore, it does not make sense to send it back and forward to a central node. In such scenarios, Spark will outperform Hadoop (Zaharia et al., 2012).

Spark is built on top of HDSF, but it does not follow the two-stage model of Hadoop. Instead, it introduces resilient distributed datasets (RDD) and parallel operations on these datasets (Gog, 2012):

- 'reduce' - combines dataset elements using a provided function;
- 'collect' - sends all the elements of the dataset to the user program;
- 'foreach' - applies a provided function onto every element of a dataset.

Spark provides two types of shared variables:

- 'accumulators' - variables onto each worker can apply associative operations, meaning that they are efficiently supported in parallel;
- 'broadcast variables' - sent once to every node, with nodes then keeping a read-only copy of those variables (Zecevic, 2016).

The Spark job scheduler implementation is conceptually similar to that of Dryad's Quincy. However, it considers which partitions of RDD are available in the memory. The framework then re-computes missing partitions, and tasks are sent to the closest possible node to the input data required (Zaharia et al., 2012).

Another significant feature implemented in Spark is the concept of 'delayed scheduling'. In situations when a head-of-line job that should be scheduled next cannot launch a local task, Spark's scheduler delays the task execution and lets other jobs start their tasks instead. However, if the job has been skipped long enough, typically a period of up to ten seconds, it launches a non-local task. Since a typical Spark workload consists of short tasks, meaning that it has a high task slots churn, tasks have a higher chance of being executed locally. This feature helps to achieve 'data locality' which is nearly optimal, and which has a very small effect on fairness; in addition, the cluster throughput can be almost doubled, as shown in an analysis performed on Facebook's workload traces (Zaharia et al., 2010).

iii. *Distributed Stream Processing*

The core concept behind distributed stream processing engines is the processing of incoming data items in real time by modelling a data flow in which there are several stages which can be processed in parallel. Other techniques include splitting the data stream into multiple sub-streams and redirecting them into a set of networked nodes (Liu and Buyya, 2017).

Inspired by Microsoft's research into DAG models (Isard et al., 2009), Apache Storm (Storm) is a distributed stream processing engine used by Twitter following extensive development (Toshniwal et al., 2014). Its initial release was 17 September 2011, and by September 2014 it had become open-source and an Apache Top-Level Project.

The defined topology acts as a distributed data transformation pipeline. The programs in Storm are designed as a topology in the shape of DAG, consisting of 'spouts' and 'bolts':

- 'Spouts' read the data from external sources and emit them into the topology as a stream of 'tuples'. This structure is accompanied by a schema which defines the names of the tuples' fields. Tuples can contain primitive values such as integers, longs, shorts, bytes, strings, doubles, floats, booleans, and byte arrays. Additionally, custom serializers can be defined to interpret this data.
- The processing stages of a stream are defined in 'bolts' which can perform data manipulation, filtering, aggregations, joins, and so on. Bolts can also constitute more complex transforming

structures that require multiple steps (thus, multiple bolts). The bolts can communicate with external applications such as databases and Kafka queues (Toshniwal et al., 2014).

In comparison to MapReduce and iterative algorithms introduced in the subsections above, Storm topologies, once created, run indefinitely until killed. Given this, the inefficient scattering of application's tasks among Cluster nodes has a lasting impact on performance. Storm's default scheduler implements a Round Robin strategy. For resource allocation purposes, Storm assumes that every worker is homogenous. This design results in frequent resource over-allocation and inefficient use of inter-system communications (Kulkarni et al., 2018). To try and solve this issue, more complex solutions are proposed such as D-Storm (Liu and Buyya, 2017). D-Storm's scheduling strategy is based on a metaheuristic algorithm Greedy, which also monitors the volume of the incoming workload and is resource-aware.

Typical examples of Storm's usage include:

- Processing a stream of new data and updating databases in real time, for example in trading systems wherein data accuracy is crucial;
- Continuously querying and forwarding the results to clients in real time, for example streaming trending topics on Twitter into browsers, and
- A parallelization of a computing-intensive query on the fly, i.e., a distributed Remote Procedure Call (RPC) wherein a large number of sets are probed (Marz, 2011).

Storm has gained widespread popularity and is used by companies such as Groupon, Yahoo!, Spotify, Verisign, Alibaba, Baidu, Yelp, and many more. A comprehensive list of users is available at the storm.apache.org website.

At the time of writing, Storm is being replaced at Twitter by newer distributed stream processing engine – Heron (Kulkarni et al., 2018) which continues the DAG model approach, but focuses on various architectural improvements such as reduced overhead, testability, and easier access to debug data.

V. Google's Borg

To support its operations, Google utilizes a high number of data centers around the world, which at the time of writing number sixteen. Borg admits, schedules, starts, restarts and monitors the full range of applications run by Google. Borg users are Google developers and system administrators, and users submit their workload in the form of jobs. A job may consist of one or more tasks that all run the same program (Burns et al., 2016).

a) Design Concepts

The central module of the Borg architecture is BorgMaster, which maintains an in-memory copy of most of the state of the cell. This state is also saved in a distributed Paxos-based store (Lamport, 1998). While BorgMaster is logically a single process, it is replicated five times to improve fault-tolerance. The main design priority of Borg was resilience rather than performance. Google services are seen as very durable and reliable, the result of multi-tier architecture, where no component is a single point of failure exists. Current allocations of tasks are saved to Paxos-based storage, and the system can recover even if all five BorgMaster instances fail. Each cell in the Google Cluster in managed by a single BorgMaster controller. Each machine in a cell runs BorgLet, an agent process responsible for starting and stopping tasks and also restarting them should they fail. BorgLet manages local resources by adjusting local OS kernel settings and reporting the state of its node to the BorgMaster and other monitoring systems.

The Borg system offers extensive options to control and shape its workload, including priority bands for tasks (i.e., monitoring, production, batch, and best effort), resources quota and admission control. Higher priority tasks can pre-empt locally-running tasks to obtain the resources which are required. The exception is made for production tasks which cannot be pre-empted. Resource quotas are part of admission control and are expressed as a resource vector at a given priority, for some time (usually months). Jobs with insufficient quotas are rejected immediately upon submission. Production jobs are limited to actual resources available to BorgMaster in a given cell. The Borg system also exposes a web-based interface called Sigma, which displays the state of all users' jobs, shows details of their execution history and, if the job has not been scheduled, also provides a 'why pending?' annotation where there is guidance about how to modify the job's resource requests to better fit the cell (Verma et al., 2015).

The dynamic nature of the Borg system means that tasks might be started, stopped and then rescheduled on an alternative node. Google engineers have created the concept of a static Borg Name Service (BNS) which is used to identify a task run within a cell and to retrieve its endpoint address. The BNS address is predominantly used by load balancers to transparently redirect RPC calls to the endpoint of a given task. Meanwhile, the Borg's resource reclamation mechanisms help to reclaim under-utilized resources from cell nodes for non-production tasks. Although in theory users may request high resource quotas for their tasks, in practice they are rarely fully utilized continuously. Instead, they have peak times of the day or are used in this way when coping with a denial-of-service attack. BorgMaster has routines that estimate resource usage levels for a task and reclaim the rest for

low-priority jobs from the batch or the best effort bands (Verma et al., 2015).

b) Jobs Schedulers

Early versions of Borg had a simple, synchronous loop that accepted jobs requests and evaluated on which node to execute them. The current design of Borg deploys several schedulers working in parallel – the scheduler instances use a shared state of the available resources, but the resource offers are not locked during scheduling decisions (optimistic concurrency control). Where there is a conflicting situation where two or more schedulers allocate jobs to the same resources, all the jobs involved are returned to the jobs queue (Schwarzkopf et al., 2013).

When allocating a task, Borg's scheduler scores a set of available nodes and selects the most feasible machine for this task. Initially, Borg implemented a variation of the Enhanced Parallel Virtual Machine algorithm (E-PVM) (Amir et al., 2000) for calculating the task allocation score. Although this resulted in the fair distribution of tasks across nodes, it also resulted in increased fragmentation and later difficulties when fitting large jobs which required the most of the node's resources or even the whole node itself. An opposite to the E-PVM approach is a best-fit strategy, which, in turn, packs tasks very tightly. The best-fit approach may result in the excessive pre-empting of other tasks running on the same node, especially when the user miscalculates the resources required, or when the application has frequent load spikes. The current model used by Borg's scheduler is a hybrid approach that tries to reduce resource usage gaps (Verma et al., 2015).

Borg also takes advantage of resources pre-allocation using 'allocs' (short for allocation). Allocs can be used to pre-allocate resources for future tasks to retain resources between restarting a task or to gather class-equivalent or related tasks, such as web applications and associated log-saver tasks, onto the same machine. If an alloc is moved to another machine, its tasks are also rescheduled.

One point to note is that, similar to MetaCentrum users (Klusáček and Rudová, 2010), Google's users tend to overestimate the memory resources needed to complete their jobs, to prevent jobs being killed due to exceeding the allocated memory. In over 90% of cases, users overestimate how many resources are required, which in certain cases can waste up to 98% of the requested resource (Moreno et al., 2013; Ray et al., 2017).

c) Optimisations

Over the years, Borg design has acquired several optimizations, namely:

- Score caching – checking the node's feasibility and scoring it is a computation-expensive process. Therefore, scores for nodes are cached and small differences in the required resources are ignored;
- Equivalence classes – submitted jobs often consist of several tasks which use the same binary and which have identical requirements. Borg's scheduler considers such a group of tasks to be in the same equivalence class. It evaluates only one task per equivalence class against a set of nodes, and later reuses this score for each task from this group;
- Relaxed randomization – instead of evaluating a task against all available nodes, Borg examines machines in random order until it finds enough feasible nodes. It then selects the highest scoring node in this set.

While the Borg architecture remains heavily centralized, this approach does seem to be successful. Although this eliminates head-of-line job blocking problems and offers better scalability, it also generates additional overheads for solving resource collisions. Nevertheless, the benefits from better scalability often outweigh the incurred additional computation costs which arise when scalability targets are achieved (Schwarzkopf et al., 2013).

VI. Summary and Conclusions

This paper has presented a taxonomy of available schedulers, ranging from early implementations to modern versions. Aside from optimizing throughput, different class schedulers have evolved to solve different problems. For example, while OS schedulers maximize responsiveness, Cluster schedulers focus on scalability, provide support a wide range of unique (often legacy) applications, and maintain fairness. Big Data schedulers are specialized to solve issues accompanying operations on large datasets, and their scheduling mechanisms are often extensively intertwined with programming language features.

Table 1 presents a comparison of the presented schedulers with their main features and deployed scheduling algorithms:

Table 1: Schedulers comparison

Scheduler class	Requirements known pre-execution	Fault-tolerance mechanisms	Configuration	Common algorithms	Scheduling decision overhead	Design focus (aside throughput)
OS Schedulers	No	No	Simple (compile-time and runtime parameters)	CS, CQ, MLFQ, O(n), O(1), Staircase, WFQ	very low – low	• single machine • NUMA awareness • Responsiveness • simple configuration
Cluster Schedulers	Yes[1]	Yes	Complex (configuration files and GUI)	FCFS (backfilling and gang-scheduling), SJF, Best-Fit, Scoring	low - high	• distributed nodes • fairness • complex sharing policy • power consumption
Big Data Schedulers	Yes[2]	Yes	Complex (configuration files and GUI)	Best-Fit, FCFS (locality and gang-scheduling), Greedy, Fair	low - medium	• specialized frameworks • parallelism • distributed data storage

1. Cluster users are notorious in overestimating resources needed for the completion of their tasks, which results in cluster system job schedulers often over-allocating resources (Klusáček and Rudová, 2010; Moreno et al., 2013).
2. MapReduce jobs tend to have consistent resource requirements, i.e., in majority of cases, every 'map' task processes roughly the same amount of data (input data block size is constant), while 'reduce' task requirements shall be directly correlated to the size of returned data.

OS schedulers have evolved in such a way that their focus is on maximizing responsiveness while still providing good performance. Interactive processes which sleep more often should be allocated time-slices more frequently, while background processes should be allocated longer, but less frequent execution times. CPU switches between processes extremely rapidly which is why modern OS scheduling algorithms were designed with very low overhead (Wong et al., 2008; Pinel et al., 2011). Most end-users for this class of schedulers are non-technical. As such, those schedulers usually have a minimum set of configuration parameters (Groves et al., 2009).

OS scheduling was previously deemed to be a solved problem (Torvalds, 2001), but the introduction and popularization of multi-core processors by Intel (Intel Core™2 Duo) and AMD (AMD Phenom™ II) in the early 2000s enabled applications to execute in parallel. This meant that scheduling algorithms needed to be re-implemented tobe efficient once more. Modern OS schedulers also consider NUMA properties when deciding which CPU core the task will be allocated to. Furthermore, the most recent research explores the potential application of dynamic voltage and frequency scaling technology in scheduling to minimize power consumption by CPU cores (Sarood et al., 2012; Padoin et al., 2014). Given that it is hard to build a good universal solution which caters to the complexities of modern hardware, it is reasonable to develop the modular scheduler architecture suggested in Lozi et al. (2016).

Cluster schedulers have a difficult mission in ensuring 'fairness'. In this context, namely a very dynamic environment consisting of variety of applications, fairness means sharing cluster resources proportionally while simultaneously ensuring a stable throughput. Cluster systems tend to allow administrators to implement complex resource sharing policies with multiple input parameters (Adaptive Computing, 2002). Cluster systems implement extensive fault-tolerance strategies and sometimes also focus on minimizing power consumption (Lang and Patel, 2010). Surprisingly, it appears that the most popular scheduling approach is a simple FCFS strategy with variants of backfilling. However, due to the rapidly increasing cluster size, the current research focuses on parallelization, as seen with systems such as Google's Borg and Microsoft's Apollo.

Big Data systems are still rapidly developing. Nodes in Big Data systems fulfil the dual purposes of storing distributed file system parts and providing a parallel execution environment for system tasks. Big

Data schedulers inherit their general design from the cluster system's jobs schedulers. However, they are usually much more specialized for the framework and are also intertwined with the programming language features. Big Data schedulers are often focused on 'locality optimization' or running a given task on a node where input data is stored or in the closest proximity to it.

The design of modern scheduling strategies and algorithms is a challenging and evolving field of study. While early implementations often used simplistic approaches, such as a CS, modern solutions use complex scheduling schemas. Moreover, the literature frequently mentions the need for a modular scheduler architecture (Vavilapalli et al., 2013; Lozi et al., 2016) which could customize scheduling strategies to hardware configuration or applications.

References Références Referencias

1. "Apache Aurora." Aurora. Available from: http://aurora.apache.org/ Retrieved December 5, 2018. Version 0.19.0.
2. "Marathon: A container orchestration platform for Mesos and DC/OS." Mesosphere, Inc. January 10, 2018. Available from: https://mesosphere.github.io/marathon/ Retrieved February 7, 2018.
3. "Maui Administrator's Guide." Adaptive Computing Enterprises, Inc. May 16, 2002. Available from: http://docs.adaptivecomputing.com/maui/pdf/mauiadmin.pdf Retrieved November 5, 2014. Version 3.2.
4. "Nomad - Easily Deploy Applications at Any Scale", HashiCorp. Available from: https://www.nomadproject.ioRetrieved March 19, 2018. Version 0.7.1.
5. "Top500 List - November 2017". TOP500 Project. November, 2017. Available from: https://www.top500.org/lists/2017/11/ Retrieved November 17, 2017.
6. "TORQUE Resource Manager. Administration Guide 5.1.2." Adaptive Computing Enterprises, Inc. November 2015. Available from: http://docs.adaptivecomputing.com/torque/5-1-2/torqueAdminGuide-5.1.2.pdf Retrieved November 15, 2016.
7. Amir, Yair, Baruch Awerbuch, Amnon Barak, R. Sean Borgstrom, and Arie Keren. "An opportunity cost approach for job assignment in a scalable computing cluster." IEEE Transactions on parallel and distributed Systems 11, no. 7 (2000): 760-768.
8. Arpaci-Dusseau, Remzi H., and Andrea C. Arpaci-Dusseau. "Operating systems: Three easy pieces." Arpaci-Dusseau Books, 2015.
9. Barroso, Luiz André, Jeffrey Dean, and UrsHölzle. "Web search for a planet: The Google cluster architecture." Micro, IEEE 23, no. 2 (2003): 22-28.
10. Becchetti, L, Stefano Leonardi, Alberto Marchetti-Spaccamela, Guido Schäfer, and TjarkVredeveld. (2006) "Average-case and smoothed competitive analysis of the multilevel feedback algorithm." Mathematics of Operations Research 31, no. 1: 85-108.
11. Blagodurov, Sergey, Sergey Zhuravlev, Alexandra Fedorova, and Ali Kamali. "A case for NUMA-aware contention management on multicore systems." In Proceedings of the 19th international conference on Parallel architectures and compilation techniques, pp. 557-558. ACM, 2010.
12. Bode, Brett, David M. Halstead, Ricky Kendall, Zhou Lei, and David Jackson. "The Portable Batch Scheduler and the Maui Scheduler on Linux Clusters." In Annual Linux Showcase & Conference. 2000.
13. Bonald, Thomas, Laurent Massoulié, Alexandre Proutiere, and JormaVirtamo. "A queueing analysis of max-min fairness, proportional fairness and balanced fairness." Queueing systems 53, no. 1 (2006): 65-84.
14. Boutin, Eric, JaliyaEkanayake, Wei Lin, Bing Shi, Jingren Zhou, Zhengping Qian, Ming Wu, and Lidong Zhou. "Apollo: Scalable and Coordinated Scheduling for Cloud-Scale Computing." In *OSDI*, vol. 14, pp. 285-300. 2014.
15. Bu, Yingyi, Bill Howe, Magdalena Balazinska, and Michael D. Ernst. "HaLoop: Efficient iterative data processing on large clusters." Proceedings of the VLDB Endowment 3, no. 1-2 (2010): 285-296.
16. Bulpin, James R. "Operating system support for simultaneous multithreaded processors." No. UCAM-CL-TR-619. University of Cambridge, Computer Laboratory, 2005.
17. Burns, Brendan, Brian Grant, David Oppenheimer, Eric Brewer, and John Wilkes. "Borg, Omega, and Kubernetes." Communications of the ACM 59, no. 5 (2016): 50-57.
18. Campbell, Matthew. "Distributed Scheduler Hell." DigitalOcean. SREcon17 Asia/Australia. May 24, 2017.
19. Corbató, Fernando J., Marjorie Merwin-Daggett, and Robert C. Daley. "An experimental time-sharing system." In Proceedings of the May 1-3, 1962, spring joint computer conference, pp. 335-344. ACM, 1962.
20. Corbet, Jonathan. "The staircase scheduler." LWN.net. June 2, 2004. Available from: https://lwn.net/Articles/87729/ Retrieved September 25, 2017.
21. Corbet, Jonathan. "The Rotating Staircase Deadline Scheduler." LWN.net. March 6, 2007. Available from: https://lwn.net/Articles/224865/ Retrieved September 25, 2017.
22. Corbett, James C., Jeffrey Dean, Michael Epstein, Andrew Fikes, Christopher Frost, Jeffrey John Furman, Sanjay Ghemawat et al. "Spanner: Google's globally distributed database." ACM

Transactions on Computer Systems (TOCS) 31, no. 3 (2013): 8.
23. Dean, Jeffrey, and Sanjay Ghemawat. "MapReduce: a flexible data processing tool." Communications of the ACM 53, no. 1 (2010): 72-77.
24. Drepper, Ulrich. "What every programmer should know about memory." Red Hat, Inc. 11 (2007): 2007.
25. Etsion, Yoav, and Dan Tsafrir. "A short survey of commercial cluster batch schedulers." School of Computer Science and Engineering, the Hebrew University of Jerusalem 44221 (2005): 2005-13.
26. Foster, Ian, and Carl Kesselman. "Globus: A metacomputing infrastructure toolkit." The International Journal of Supercomputer Applications and High Performance Computing 11, no. 2 (1997): 115-128.
27. Foster, Ian, Carl Kesselman, and Steven Tuecke. "The anatomy of the grid: Enabling scalable virtual organizations." The International Journal of High Performance Computing Applications 15, no. 3 (2001): 200-222.
28. Gabriel, Edgar, Graham E. Fagg, George Bosilca, TharaAngskun, Jack J. Dongarra, Jeffrey M. Squyres, Vishal Sahay et al. "Open MPI: Goals, concept, and design of a next generation MPI implementation." In European Parallel Virtual Machine/Message Passing Interface Users' Group Meeting, pp. 97-104. Springer Berlin Heidelberg, 2004.
29. Gentzsch, Wolfgang. "Sun grid engine: Towards creating a compute power grid." In Cluster Computing and the Grid, 2001. Proceedings. First IEEE/ACM International Symposium on, pp. 35-36. IEEE, 2001.
30. Ghemawat, Sanjay, Howard Gobioff, and Shun-Tak Leung. "The Google file system." In ACM SIGOPS operating systems review, vol. 37, no. 5, pp. 29-43. ACM, 2003.
31. Gog, I. "Dron: An Integration Job Scheduler." Imperial College London (2012).
32. Grimshaw, Andrew S. "The Mentat run-time system: support for medium grain parallel computation." In Distributed Memory Computing Conference, 1990., Proceedings of the Fifth, vol. 2, pp. 1064-1073. IEEE, 1990.
33. Grimshaw, Andrew S., William A. Wulf, James C. French, Alfred C. Weaver, and Paul Reynolds Jr. "Legion: The next logical step toward a nationwide virtual computer." Technical Report CS-94-21, University of Virginia, 1994.
34. Groves, Taylor, Jeff Knockel, and Eric Schulte. "BFS vs. CFS - Scheduler Comparison." The University of New Mexico, 11 December 2009.
35. Hamscher, Volker, Uwe Schwiegelshohn, Achim Streit, and RaminYahyapour. "Evaluation of job-scheduling strategies for grid computing." Grid Computing—GRID 2000 (2000): 191-202.
36. Hart, Johnson M. "Win32 systems programming." Addison-Wesley Longman Publishing Co., Inc., 1997.
37. Helland, Pat, and Harris Ed "Cosmos: Big Data and Big Challenges." Stanford University, October 26, 2011.
38. Hindman, Benjamin, Andy Konwinski, Matei Zaharia, Ali Ghodsi, Anthony D. Joseph, Randy H. Katz, Scott Shenker, and Ion Stoica. "Mesos: A Platform for Fine-Grained Resource Sharing in the Data Center." In NSDI, vol. 11, no. 2011, pp. 22-22. 2011.
39. Isard, Michael, Mihai Budiu, Yuan Yu, Andrew Birrell, and Dennis Fetterly. "Dryad: distributed data-parallel programs from sequential building blocks." In ACM SIGOPS operating systems review, vol. 41, no. 3, pp. 59-72. ACM, 2007.
40. Isard, Michael, Vijayan Prabhakaran, Jon Currey, Udi Wieder, Kunal Talwar, and Andrew Goldberg. "Quincy: fair scheduling for distributed computing clusters." In Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles, pp. 261-276. ACM, 2009.
41. Jackson, David, Quinn Snell, and Mark Clement. "Core algorithms of the Maui scheduler." In Workshop on Job Scheduling Strategies for Parallel Processing, pp. 87-102. Springer, Berlin, Heidelberg, 2001.
42. Jones, M. Tim. "Inside the Linux 2.6 Completely Fair Scheduler - Providing fair access to CPUs since 2.6.23" In IBM Developer Works. December 15, 2009.
43. Kannan, Subramanian, Mark Roberts, Peter Mayes, Dave Brelsford, and Joseph F. Skovira. "Workload management with LoadLeveler." IBM Redbooks 2, no. 2 (2001).
44. Kay, Judy, and Piers Lauder. "A fair share scheduler." Communications of the ACM 31, no. 1 (1988): 44-55.
45. Klusáček, Dalibor, and Hana Rudová. "The Use of Incremental Schedule-based Approach for Efficient Job Scheduling." In Sixth Doctoral Workshop on Mathematical and Engineering Methods in Computer Science, 2010.
46. Klusáček, Dalibor, Václav Chlumský, and Hana Rudová. "Optimizing user oriented job scheduling within TORQUE." In Super Computing the 25th International Conference for High Performance Computing, Networking, Storage and Analysis (SC'13). 2013.
47. Kolivas, Con. "linux-4.8-ck2, MuQSS version 0.114." -ck hacking. October 21, 2016. Available from: https://ck-hack.blogspot.co.uk/2016/10/linux-48-ck2-muqss-version-0114.html Retrieved December 8, 2016.
48. Krauter, Klaus, Rajkumar Buyya, and Muthucumaru Maheswaran. "A taxonomy and survey of grid resource management systems for distributed

computing." Software: Practice and Experience 32, no. 2 (2002): 135-164.
49. Kulkarni, Sanjeev, Nikunj Bhagat, Maosong Fu, VikasKedigehalli, Christopher Kellogg, Sailesh Mittal, Jignesh M. Patel, Karthik Ramasamy, and Siddarth Taneja. "Twitter Heron: Stream processing at scale." In Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, pp. 239-250. ACM, 2015.
50. Lamport, Leslie. "The part-time parliament." ACM Transactions on Computer Systems (TOCS) 16, no. 2 (1998): 133-169.
51. Lang, Willis, and Jignesh M. Patel. (2010) "Energy management for mapreduce clusters." Proceedings of the VLDB Endowment 3, no. 1-2: 129-139.
52. Lewis, Ian, and David Oppenheimer. "Advanced Scheduling in Kubernetes". Kubernetes.io. Google, Inc. March 31, 2017. Available https://kubernetes.io/blog/2017/03/advanced-scheduling-in-kubernetes Retrieved January 4, 2018.
53. Litzkow, Michael J., MironLivny, and Matt W. Mutka. "Condor-a hunter of idle workstations." In Distributed Computing Systems, 1988., 8th International Conference on, pp. 104-111. IEEE, 1988.
54. Liu, Xunyun, and Rajkumar Buyya. "D-Storm: Dynamic Resource-Efficient Scheduling of Stream Processing Applications." In Parallel and Distributed Systems (ICPADS), 2017 IEEE 23rd International Conference on, pp. 485-492. IEEE, 2017.
55. Lozi, Jean-Pierre, Baptiste Lepers, Justin Funston, Fabien Gaud, Vivien Quéma, and Alexandra Fedorova. "The Linux scheduler: a decade of wasted cores." In Proceedings of the Eleventh European Conference on Computer Systems, p. 1. ACM, 2016.
56. Marz, Nathan. "A Storm is coming: more details and plans for release." Engineering Blog. Twitter, Inc. August 4, 2011. Available from: https://blog.twitter.com/engineering/en_us/a/2011/a-storm-is-coming-more-details-and-plans-for-release.html Retrieved July 16, 2018.
57. McCullough, John C., Yuvraj Agarwal, Jaideep Chandrashekar, Sathyanarayan Kuppuswamy, Alex C. Snoeren, and Rajesh K. Gupta. "Evaluating the effectiveness of model-based power characterization." In USENIX Annual Technical Conf, vol. 20. 2011.
58. Moreno, Ismael Solis, Peter Garraghan, Paul Townend, and Jie Xu. "An approach for characterizing workloads in google cloud to derive realistic resource utilization models." In Service Oriented System Engineering (SOSE), 2013 IEEE 7th International Symposium on, pp. 49-60. IEEE, 2013.
59. Murray, Derek G., Malte Schwarzkopf, Christopher Smowton, Steven Smith, Anil Madhavapeddy, and Steven Hand. "CIEL: a universal execution engine for distributed data-flow computing." In Proc. 8th ACM/USENIX Symposium on Networked Systems Design and Implementation, pp. 113-126. 2011.
60. Naik, Nitin. "Building a virtual system of systems using Docker Swarm in multiple clouds." In Systems Engineering (ISSE), 2016 IEEE International Symposium on, pp. 1-3. IEEE, 2016.
61. Pabla, Chandandeep Singh. "Completely fair scheduler." Linux Journal 2009, no. 184 (2009): 4.
62. Padoin, Edson L., Márcio Castro, Laércio L. Pilla, Philippe OA Navaux, and Jean-François Méhaut. "Saving energy by exploiting residual imbalances on iterative applications." In High Performance Computing (HiPC), 2014 21st International Conference on, pp. 1-10. IEEE, 2014.
63. Pascual, Jose, Javier Navaridas, and Jose Miguel-Alonso. "Effects of topology-aware allocation policies on scheduling performance." In Job Scheduling Strategies for Parallel Processing, pp. 138-156. Springer Berlin/Heidelberg, 2009.
64. Pinel, Frédéric, Johnatan E. Pecero, Pascal Bouvry, and Samee U. Khan. "A review on task performance prediction in multi-core based systems." In Computer and Information Technology (CIT), 2011 IEEE 11th International Conference on, pp. 615-620. IEEE, 2011.
65. Pinheiro, Eduardo, Ricardo Bianchini, Enrique V. Carrera, and Taliver Heath. "Load balancing and unbalancing for power and performance in cluster-based systems." In Workshop on compilers and operating systems for low power, vol. 180, pp. 182-195. 2001.
66. Pop, Florin, CiprianDobre, Gavril Godza, and Valentin Cristea. "A simulation model for grid scheduling analysis and optimization." In Parallel Computing in Electrical Engineering, 2006. PAR ELEC 2006. International Symposium on, pp. 133-138. IEEE, 2006.
67. Ray, Biplob R., Morshed Chowdhury, and Usman Atif. "Is High Performance Computing (HPC) Ready to Handle Big Data?" In International Conference on Future Network Systems and Security, pp. 97-112. Springer, Cham, 2017.
68. Rodriguez, Maria Alejandra, and Rajkumar Buyya. "A taxonomy and survey on scheduling algorithms for scientific workflows in IaaS cloud computing environments." Concurrency and Computation: Practice and Experience 29, no. 8 (2017).
69. Sarood, Osman, Phil Miller, Ehsan Totoni, and Laxmikant V. Kale. ""Cool" Load Balancing for High Performance Computing Data Centers." IEEE Transactions on Computers 61, no. 12 (2012): 1752-1764.
70. Schwarzkopf, Malte, Andy Konwinski, Michael Abd-El-Malek, and John Wilkes. "Omega: flexible, scalable schedulers for large compute clusters." In

Proceedings of the 8th ACM European Conference on Computer Systems, pp. 351-364. ACM, 2013.
71. Shreedhar, Madhavapeddi, and George Varghese. "Efficient fair queueing using deficit round robin." In ACM SIGCOMM Computer Communication Review, vol. 25, no. 4, pp. 231-242. ACM, 1995.
72. Singh, Ajit. "New York Stock Exchange Oracle Exadata – Our Journey." Oracle, Inc. November 17, 2017. Available from: http://www.oracle.com/technetwork/database/availability/con8821-nyse-2773005.pdf Retrieved June 28, 2018.
73. Sliwko, Leszek. "A Scalable Service Allocation Negotiation For Cloud Computing." Journal of Theoretical and Applied Information Technology, Vol.96. No 20, pp. 6751-6782, 2018.
74. Smanchat, Sucha, and Kanchana Viriyapant. "Taxonomies of workflow scheduling problem and techniques in the cloud." Future Generation Computer Systems 52 (2015): 1-12.
75. Smarr, Larry, and Charles E. Catlett. "Metacomputing." Grid Computing: Making the Global Infrastructure a Reality (2003): 825-835.
76. Thain, Douglas, Todd Tannenbaum, and MironLivny. "Distributed computing in practice: the Condor experience." Concurrency and computation: practice and experience 17, no. 2-4 (2005): 323-356.
77. Torvalds, Linus "Re: Just a second …" The Linux Kernel Mailing List. December 15, 2001. Available from http://tech-insider.org/linux/research/2001/1215.html Retrieved September 27, 2017.
78. Toshniwal, Ankit, Siddarth Taneja, Amit Shukla, Karthik Ramasamy, Jignesh M. Patel, Sanjeev Kulkarni, Jason Jackson et al. "Storm @Twitter." In Proceedings of the 2014 ACM SIGMOD international conference on Management of data, pp. 147-156. ACM, 2014.
79. Tyagi, Rinki, and Santosh Kumar Gupta. "A Survey on Scheduling Algorithms for Parallel and Distributed Systems." In Silicon Photonics & High Performance Computing, pp. 51-64. Springer, Singapore, 2018.
80. Vavilapalli, Vinod Kumar, Arun C. Murthy, Chris Douglas, Sharad Agarwal, Mahadev Konar, Robert Evans, Thomas Graves et al. "Apache hadoop yarn: Yet another resource negotiator." In Proceedings of the 4th annual Symposium on Cloud Computing, p. 5. ACM, 2013.
81. Verma, Abhishek, Luis Pedrosa, Madhukar Korupolu, David Oppenheimer, Eric Tune, and John Wilkes. "Large-scale cluster management at Google with Borg." In Proceedings of the Tenth European Conference on Computer Systems, p. 18. ACM, 2015.
82. White, Tom. "Hadoop: The definitive guide." O'Reilly Media, Inc. 2012.
83. Wong, C. S., I. K. T. Tan, R. D. Kumari, J. W. Lam, and W. Fun. "Fairness and interactive performance of O(1) and cfslinux kernel schedulers." In Information Technology, 2008. International Symposium on, vol. 4, pp. 1-8. IEEE, 2008.
84. Vohra, Deepak. "Scheduling pods on nodes. " In Kubernetes Management Design Patterns, pp. 199-236. Apress, Berkeley, CA. 2017.
85. Vagata, Pamela, and Kevin Wilfong. "Scaling the Facebook data warehouse to 300 PB." Facebook, Inc. April 10, 2014. Available from: https://code.fb.com/core-data/scaling-the-facebook-data-warehouse-to-300-pb/ Retrieved June 28, 2018.
86. Yoo, Andy B., Morris A. Jette, and Mark Grondona. "Slurm: Simple linux utility for resource management." In Workshop on Job Scheduling Strategies for Parallel Processing, pp. 44-60. Springer, Berlin, Heidelberg, 2003.
87. Yu, Jia, and Rajkumar Buyya. "A taxonomy of scientific workflow systems for grid computing." ACM Sigmod Record 34, no. 3 (2005): 44-49.
88. Zaharia, Matei, Dhruba Borthakur, J. Sen Sarma, Khaled Elmeleegy, Scott Shenker, and Ion Stoica. Job scheduling for multi-user mapreduce clusters. Vol. 47. Technical Report UCB/EECS-2009-55, EECS Department, University of California, Berkeley, 2009.
89. Zaharia, Matei, Mosharaf Chowdhury, Michael J. Franklin, Scott Shenker, and Ion Stoica. "Spark: Cluster computing with working sets." HotCloud 10, no. 10-10 (2010): 95.
90. Zaharia, Matei, Mosharaf Chowdhury, Tathagata Das, Ankur Dave, Justin Ma, Murphy McCauley, Michael J. Franklin, Scott Shenker, and Ion Stoica. "Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing." In Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation, pp. 2-2. USENIX Association, 2012.
91. Zakarya, Muhammad, and Lee Gillam. "Energy efficient computing, clusters, grids and clouds: A taxonomy and survey." Sustainable Computing: Informatics and Systems 14 (2017): 13-33.
92. Zecevic, Petar, and Marko Bonaci. "Spark in Action." (2016).
93. Zhang, Zhuo, Chao Li, Yangyu Tao, Renyu Yang, Hong Tang, and Jie Xu. "Fuxi: a fault-tolerant resource management and job scheduling system at internet scale." Proceedings of the VLDB Endowment 7, no. 13 (2014): 1393-1404.

Global Journal of Computer Science and Technology: B
Cloud and Distributed
Volume 19 Issue 1 Version 1.0 Year 2019
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals
Online ISSN: 0975-4172 | Print ISSN: 0975-4350 | DOI: 10.17406

Measuring the Performance on Load Balancing Algorithms

By Saumendu Roy, Dr. Md. Alam Hossain, Sujit Kumar Sen, Nazmul Hossain & Md. Rashid Al Asif

Jashore University of Science & Technology (JUST)

Abstract- Load balancing is an integrated aspect of the environment in cloud computing. Cloud computing has lately outgoing technology. It has getting exoteric day by day residence widespread chance in close to posterior. Cloud computing is defined as a massively distributed computing example that is moved by an economic scale in which a repertory of abstracted virtualized energetically. The number of clients in cloud computing is increasing exponentially. The huge amount of user requests attempt to entitle the collection for numerous applications. Which alongside with heavy load not far afield off from cloud server. Whenever particular (Virtual Machine) VMs are overloaded then there are no more duties should be addressed to overloaded VM if under loaded VMs are receivable. For optimizing accomplishment and better response or reaction time the load has to be balanced between overloaded VMs (virtual machines). This Paper describes briefly about the load balancing accession and identifies which is better than others (load balancing algorithm).

Keywords: *cloud computing, load balancing, virtualization, load balancing algorithms.*

GJCST-B Classification : *C.m*

Strictly as per the compliance and regulations of:

Measuring the Performance on Load Balancing Algorithms

Saumendu Roy[α], Dr. Md. Alam Hossain[σ], Sujit Kumar Sen[ρ], Nazmul Hossain[ω] & Md. Rashid Al Asif[¥]

Abstract- Load balancing is an integrated aspect of the environment in cloud computing. Cloud computing has lately outgoing technology. It has getting exoteric day by day residence widespread chance in close to posterior. Cloud computing is defined as a massively distributed computing example that is moved by an economic scale in which a repertory of abstracted virtualized energetically. The number of clients in cloud computing is increasing exponentially. The huge amount of user requests attempt to entitle the collection for numerous applications. Which alongside with heavy load not far afield off from cloud server. Whenever particular (Virtual Machine) VMs are overloaded then there are no more duties should be addressed to overloaded VM if under loaded VMs are receivable. For optimizing accomplishment and better response or reaction time the load has to be balanced between overloaded VMs (virtual machines). This Paper describes briefly about the load balancing accession and identifies which is better than others (load balancing algorithm).

Keywords: cloud computing, load balancing, virtualization, load balancing algorithms.

I. Introduction

Cloud computing is extensively received IT-based service. However, there is a number of issues that have not been fully figured out such as load balancing, real-time scheduling, VM migrations and many more. The load balancer is a device that distributes application or network traffic across a bunch of servers. The load balancing enhances responsiveness and raises the appearance of its. A load balancer lies between the server farm and the client. This is responsible for receiving oncoming network traffic and application. Also distributing the traffic across various backend servers using different techniques. By balancing the requests of application across numerous servers, a load balancer diminishes the load of the single server. Side by side it hinders any one application server from fitting an odd point of failure. Thus flourishing overall application responsiveness and availability.

Balancing of the load is a vital issue in the cloud that discuss with the skilled and scalable distribution of workload. Side by side it takes care that all the computing materials must be distributed. The number of requests and consumers for the services are increasing in cloud computing day by day. For this reason, load balancing is the significant research area for conducting the user's requests efficiently. The load can be network load, memory capacity, CPU load or delay. Load balancing is the process of allocating the work-load among different nodes of a distributed system. It enhances both the job response time and resource utilization. While also ignoring a situation where a number of the nodes are loaded while several nodes are inactive or doing the very tiny amount of work. Load balancing assures that every node or all the processor in the system of the network does nearly the equable number of work at any required time. This procedure can be a sender, receiver-initiated or symmetric type (a coalition of sender-initiated and receiver-initiated types). For skilled & fruitful management and behaviors of cloud service provider's resources, various algorithms of load balancing have mentioned. Additionally, the motives are to minimize resource usage, maximize throughput and reduce response time - two genres of load balancing algorithms that are static and dynamic. Static load balancing algorithms are easy to design and take very less execution time. This algorithm works on the base of the knowledge of tasks and resources of the system. But these kinds of algorithms can only be practical in the scenario, where most of the hosts in the cluster have identical processing capability. Examples of static load balancing algorithms are least connection scheduling, Round Robin, weighted round robin algorithm and many more. Dynamic load balancing algorithms are more supple and reliable. Which also capable of handling a great number of user's requests than static algorithms. These algorithms stand on the current state of the system and are the best suited for switching environment.

Author α: Lecturer at the department of Computer Science & Engineering in Nothern University of Business & Technology Khulna, Bangladesh. e-mail: saumocse3j6@gmail.com
Author σ: Assistant Professor at the department of Computer Science & Engineering in Jashore University of Science & Technology (JUST), Jashore, Bangladesh. e-mail: alamcse_iu@yahoo.com
Author ρ: Student at the department of Computer Science & Engineering in Jashore University of Science & Technology (JUST), Jashore, Bangladesh. e-mail: sujit2j8@gmail.com
Author ω: Assistant Professor at the department of Computer Science & Engineering in Jashore University of Science & Technology (JUST), Jashore, Bangladesh. e-mail: n.hossain@just.edu.bd
Author¥: Lecturer at the department of Computer Science & Engineering in Barisal University, Barisal.
e-mail: rashid.al.asif@gmail.com

II. Load Balancing

Cloud computing is the outbound internet based procedure which strengthens commercial computing. Cloud is a podium providing dynamic pool resources and virtualization. It also simplifies the scalable IT resources such as infrastructure, services and, applications. These materials work with internet on pay-per-use criteria. Which aids in the coordination of retention in a quick way. The cloud computing steps both data and computing from portable desktop and PCs to the data centers. Cloud computing can synthesize on need changes. Which in turn decreases the expense of capital required in hardware and software. Thus, cloud computing helps to provide a structure for compatible access to computing resources and that too in on-demand style. By excellence of cloud computing, resources can be accepted and shifted very firstly. As well as the interaction with less service provider. Cloud computing also enhances the availability of resources. The key point on cloud computing is scheduling (RAS) and resource allocation which is gained by using its strategies and algorithms. These have a straight outcome on cloud performance and cost. Load balancing is one of the vital concerns on cloud computing [2], i.e. When one or more elements of any job failed, it (load balancing) helps in the regularity of the services. By accomplishing provisioning and de-provisioning of examples of applications without any fail. Thus, Load Balancing is a technique for distributing the dynamic local workload evenly across all the nodes in the entire cloud. It will also ignore the condition where some nodes are loaded. While others are inactive or doing very little amount of work. Load balancing maximizes the overall performance of the system alongside with its resource utilization characteristics. This working theory of load balancing helps to gain high user contentment. By enhancing the overall system performance of the skilled distribution of every resource can be done. In cloud computing outlay of resources and preservation of energy are not considered. But their insertion along with exact load balancing helps in reducing it and forming enterprises greener. Another important criterion is the scalability in cloud computing that is enabled by load balancing. All these criteria raise resource utilization in such a way that will minimize energy outgoing and carbon foot prints, resulting in acquiring green computing.

III. Working Process of Load Balancing

Load balancing in clouds is a procedure that allocates the intemperance dynamic local workload evenly across all the nodes. It is used for gaining a service provisioning and resource utilization rate and therefore improving the overall system performance for incoming tasks. Which are coming from various positions are accepted by the load balancer and then allotted to the data center, for the accurate load distribution.

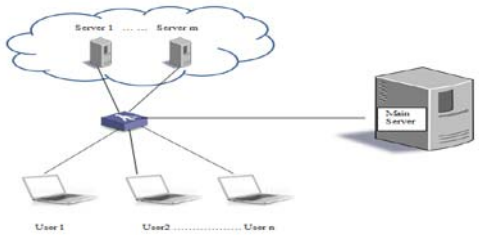

Fig. 1: Load Balancing

The target of load balancing is as follows:

- To improve service availability
- To raise the user pleasure
- To maximize resource utilization
- To minimize the waiting time and the time of task coming from a various position
- To increase the performance
- To maintain the stability of the system
- To build system fault tolerance
- To accommodate future modification

a) Central Load Balancing Decision Module (CLBDM)

To give valued information and force the decision-making method of a load balancer, thus sustaining satisfactory load balancing in cloud environments. It is not sufficient to give information from the networking part of the computer system or from the exterior load balancer. Including weighted round-robin algorithm and session-switching. So that it can insist sessions on one of the web application servers. Which can catch all data for that session attainable untilit is concluded manually or it just timeouts. CLBDM organizes information from virtualized environments and ends user experience. To be able to proactively force load balancing decisions or reactively switch it in handling critical conditions.

CLBDM solves the problem depicted in the weighted round-robin algorithm and session-switching. In a fashion that introduces a central module that is forcing decisions made by load balancers.

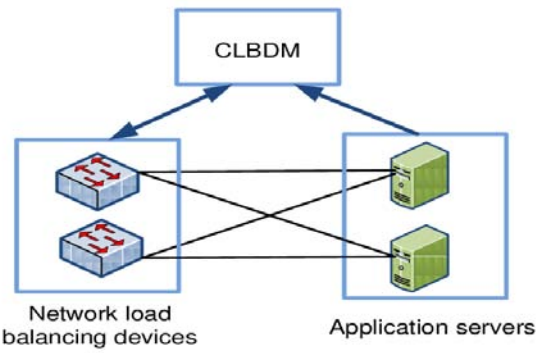

Fig. 2: CLBDM-Placement in-a-computer

The benefit of this module is to interact (monitor) whole parts of our computer system, with application servers and load balancers. Then, based on the gathered information and interior computation, CLBDM will interference forwarding decisions on load balancers. Facilitated designed is shown in Figure-2, which also delineates the direction of data flooding to and from CLBDM.

CLBDM is submitted with other data to fine tune of a user experience sensor. Which can be accomplished as an unresisting taping device or software on the application servers. This sensor is controlling the session of users including whole requests sent and received by end the user. And metering time wanted for every action to finish. If the tendency of response time for end user on an individual application server will launch to rise, this is a crystal clear signal that something is running incorrectly. When collected information rises above inartificial numbers, CLBDM is getting action and steps load to other application servers.

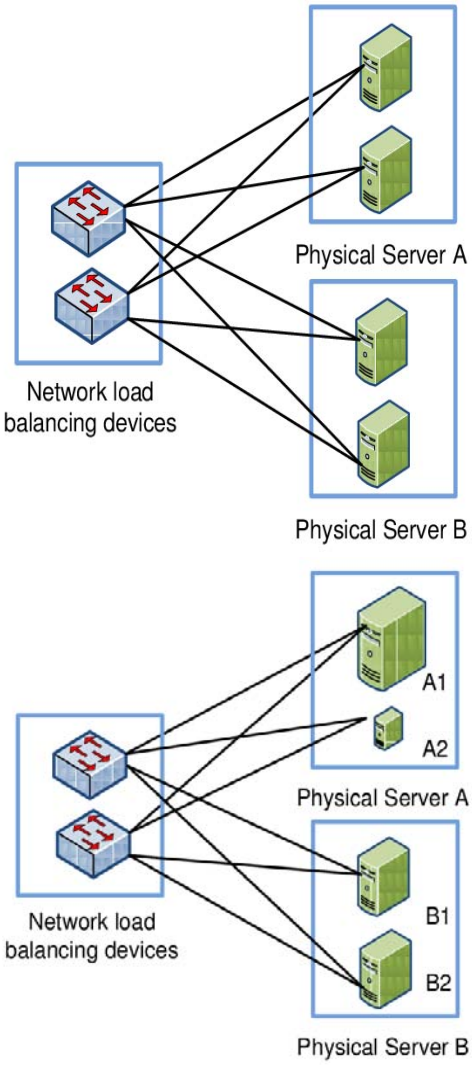

Fig. 3: Dividing & Reallocation-of-resources on physical-servers-in-several-smaller-virtualized-servers

Figure 3, can severely reduce the number of potentially attacked users. The idea here is to raise the number of application servers by using virtualization technology; it would be contingent on increasing amount of available application servers from two to four. This would instantly decrease the number of potentially attacked users from 50% to 25%.Virtualization technology that gives a dynamical allocation of resources. Which is used on virtualized servers with live migration. A move that CLBDM can make when it reveals issue with specific virtual server. It would dynamically allocate resources from one virtual server (A2) to another (A1). It is thus giving the possibility to server A1 to still complete action that has used all of its resources. But, at this time CLBDM will do even more activities. For example, it would point to load balancers that the server A1 is in trial and that it cannot take new user sessions. This thought will dynamically be returned if CLBDM gets information from server A1 that the load has been normalized.

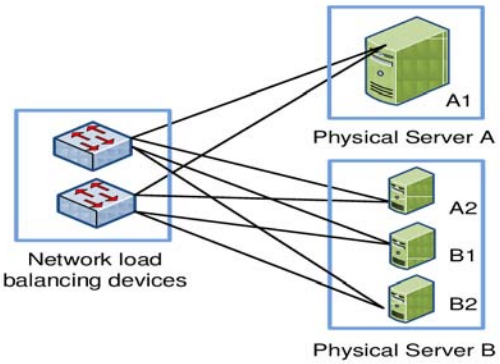

Fig. 4: Virtualization technology to live migrate virtual Server A2 to Physical Server B

Figure 4, exhibits an even further development of the formerly depictured condition. Where the load on Server A1 is still growing, now CLBDM has a choice that allows virtualized servers to be shifted from one physical server to another. It is accurately that here has occurred in a way that Server A2 has been live migrated (without any pause) to Physical Server B. While the Server A1 now has full resources of Physical Server A, giving it even more probabilities to complete jobs presently running on it.

b) Drawbacks of CLBDM Model

One of probable fact is that CLBDM can turn in a way, that it can enter unexpected loops and start to spray its decision between nodes. It is resulting in low performance and end-user experience. This kind of facts can be reduced if not ignored by cautiously designing CLBDM decision algorithms. But there should also be ignored if it is possible to find such an unwanted state.

IV. Virtualization

Virtualization means that do not exist in, but it distributes everything as actual or real. The software implementation of a machine is Virtualization that will run different programs as an actual machine. By virtualization, the end user can use the various services or applications of the cloud. For this reason, this is the master part of the cloud environment. There is a number of kind's virtualization that is used in the cloud.

Two kinds of virtualization. They are:
a. Full Virtualization
b. Para virtualization

a) Full Virtualization

Full virtualization which means that a machine is installed on another machine. This machine (virtual) gives all the functionalies that exists on the original machine. It facilitates when real machine not independent then the user uses the virtual machine.

b) Para virtualization

Para virtualization, the hardware accommodates many operating systems to run the only machine. It also allows skilled use of resources such as processor and memory.

V. Basic Types of Load Balancing Algorithms

There is an excessively large necessity for load balancing in large distributed and complicated systems. Load balancer gets a decision to shift the job to the distant server for balancing. The load balancer which can work in two ways: Cooperative and non-cooperative are the two ways. A cooperative way, to gain the satisfactory response times, entire nodes work together. A non-cooperative way, response time is incremented by the freely running the tasks. Some load balancing algorithms are smeared in this paper.

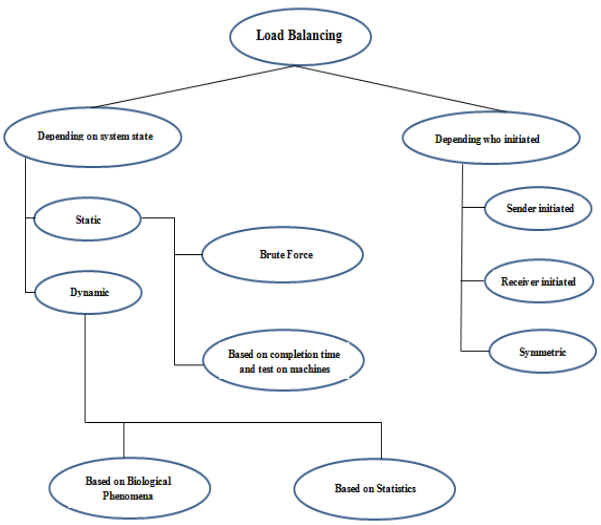

Fig. 5: Basic Types of the algorithm

- Based on the recent status of the system, the algorithms of load balancing can be categorized in two ways:
- Static algorithm: The present situation of the node is not taken into estimation [3]. All the nodes and their features are known in advance. Based on this earlier sense, the algorithm works. Since it does not use recent system status information, it is simple to implement.
- Dynamic algorithm: This type of algorithm is the basis for the recent situation of the system [3]. The algorithm works following to the dynamic changes in the instance of nodes. Status Table controls the status of the entire nodes in the cloud environment. Dynamic algorithms are not easy to implement, but it balances the load in a useful manner.

The initiator of algorithms, Load Balancing algorithms can be divided into three types:

- Sender-Initiated: Sender recognizes that the nodes are in number so that the sender initiates the perfections of load balancing algorithm.
- Receiver Initiated: The necessity of Load balancing condition can be recognized by the receiver/server in the cloud. And that server initiates the accomplishment of Load Balancing algorithm.
- Symmetric: It is the summation of both types that means the initiator of sender and receiver.

VI. Existing Load Balancing Algorithms

Following load balancing algorithms are currently prevalent in cloud.

a) Round Robin

In this algorithm [6] the round robin technique pursues a time plot or slice while processing the data. Every process is going to perform in the time slice and then change to other and follow on ring the path. In round robin, until all processes completed their task, a balancing procedure is followed so that it can balance the process in a group. The process is going to happen in round robin until all processes finish their task. Such that a balance method is implemented to equal the process in a group pattern. This algorithm is extensively used in web servers where HTTP requests are very analogous and distributed equally.

b) Min-Min

The manager of the cloud recognizes the minimum completion and execution time of the unassigned each task. Which are waiting to execute and store in a queue. Min-min is static load balancing algorithm. So the metrics concerned to the job are known in advance. In these types of the algorithm, the cloud manager firstly concerns with the jobs having minimum performance time. By bestowing those to the corresponding processors with the capability of

complete the job in fixed finishing time. The jobs have to wait for the un-specific period because of maximum execution time. In the processor, until all the tasks are assigned, the assigned tasks are updated in the processors. As well as the task is cleared and the process is periodic till all the tasks are mapped in the waiting queue. Min-min starts by a set of all unassigned jobs and found each task minimum finishing time. Amongst these least times, the minimum value is chosen. That is the minimum time between all the tasks on any resources. Then following to that minimal time, the task is cadastral on the corresponding processor. After then the perfection time for all other jobs is updated on that machine. It is done by adding the execution time of the assigned task to the execution times of other task on that machine. And finally, assigned task is dismissed from the list of the jobs that are to be assigned to the machines. Then again the similar method is followed until all the tasks are assigned on the resources. But this way has a drawback that it can lead to starvation.

c) Opportunistic Load Balancing Algorithm (OLB)

It is a static load balancing algorithm. For this reason, it does not consider the recent workload of the Virtual Machines. Its effort to repose each node busy. This algorithm deals rapidly with the unexecuted tasks in indiscriminate order to the recently available node. Every task is assigned to the node lamely. It gives a load balance scheme without good results. The task will process sluggishly because it does not compute the present execution time of the node. Advantages that it endeavors' to keep each node engaged. This algorithm deals rapidly with the unexecuted tasks in random order to the recently available node. Every task is assigned to the node randomly. Disadvantages that it provides load balance scheme with poor results. The task will process in a slack manner because it does not compute the current execution time of the node.

d) Max Min algorithm

Max Min algorithm which works as similar to the Min-Min algorithm without the following: after discovering the minimum execution time, the cloud manager handles with tasks having maximum execution time. The assigned task is dismissed from the list of the tasks. Which are to be assigned to the processor and the working time for all other tasks is updated on that processor. Because of its static way the necessity are informed in advance then the algorithm completed well. An improved rending of max-min algorithm is mentioned in [7]. It is the basis of the cases, where meta-tasks include kindred tasks of their execution and completion time. Development in the ability of the algorithm is gained by increasing the scope of concurrent execution of tasks on resources. Advantages of Max-min technique which resolves the rigidity of Min-min. By giving Precedence to a large number of tasks. The Max-min algorithm picks the task with the Maximum completion time and assigns it to the resources on which it acquires minimum execution time. It is crystal clear that the Max-min feels better option. Whenever the number of little tasks is much more than large ones, one of the properties of the Max-min method is that selects large tasks to be completed quickly. Which in turn little task stays for high time and it is the Disadvantages of Max-Min.

e) Honeybee Foraging load balancing Algorithm

The idea astern the BCO (Bee Colony Optimization) is to make the multi-agent system (colony of artificial bees) able to effectively solve stiff combinatorial optimization problems. A colony of honey bee can enhance itself over long ranges as to trace multiple food sources. Like as flower patches and then these bees harvest nectar or faring from these sources. A little amount of the colony discovers the environment looking for new flower patches when the food source is encountered the scout bees. Going in the field that surrounds the hive and control for quality beneficial. When they back to the hive, the scouts gather the food harvested. There is a location in the hive named as the "dance floor.'' Where waggle dance is completed by the bees that found a very profitable food. By the dance (waggle) a scout bee goes through the location of its search to idle viewers, which aids in the using of the flower patch. Here the term of the dance is following to the scout's rating of the food source. To harvest the best-rated flower patches, more foragers get recovered. When the dance is finished, the scout back to the food source. It found to see more food. Till the food is profitable, food sources will be posted by the scouts when they back to their hive. Foragers who are recovered currently may waggle dance as well. Those will step-up the recruitment for profitable flower patches. This method will go on to discover the most profitable flower patches. The Honey Bee nature stimulated load balancing algorithm was mentioned, which aims to gain well-balanced load cross VMs to maximize the throughput. And side by side, to balance the precedence of tasks on the VMs. So, the amount of waiting time of the tasks on the list is minimum. Using this algorithm reduction in waiting time and average execution time of tasks on the list was improved. This algorithm works for heterogeneous type of systems. And for balancing non-preemptive independent tasks. Some advantages that it has low overhead, less migration time, less response time and optimum resource utilization. On the other side, it has some disadvantages that it has fewer throughputs, less fault tolerant and less scalability.

f) Active Clustering

The idea of clustering in cloud computing is active Clustering is a clustering based algorithm. By making a bunch or cluster of nodes, the performance of

an algorithm can be increased. Every cluster can be occupied as a group. The main idea behind an active clustering is to group the same nodes together and then work on these groups. The method of forming a cluster revolves around the idea of the node. In this technique, the first node points a neighbor node named the matchmaker node which is of a variant type. This matchmaker node creates a connection with its surroundings, which is of similar types as the earlier node. At last the matchmaker node gets separated. This technique is followed repetitively. The system performance is increased with handsome availability of resources. Thereby enhancing the throughput. This increment in throughput is due to the proficient utilization of resources.

g) Ant colony optimization

The main idea of ACO is to copy the foraging characteristics of ant colonies. When ant groups try to find for the food, they uses an exceptional type of chemical to link with each other. That, chemical is named to as pheromone. At the very fast ants begin to search their foods loosely. Once the ants find a way to the food source, they consent pheromone on the path. To sense pheromone on the ground, an ant can pursue the trails of the other ants to the food source. As this method continues, maximum ants attract to select the shortest path as there have been a massive amount of pheromones congealed on this path. This united pheromone is saving and pheromone following characteristic of ants turn the illuminative source of ACO.

h) OLB+LBMM

Opportunistic Load Balancing and Load Balance Min-Min are referred as OLB and LBMM algorithms to use better-executing efficiency. And control the system of load balancing. This linked way helps in a skilled utilization of resources and increases work efficiency. It provides a better output than the above-argued algorithms. It is the summation of Opportunistic Load Balancing and Load Balance Min-Min scheduling algorithms to use better execution efficiency. And control the load balancing of the system. OLB scheduling algorithm stays every node in active state to acquire the aim of load balance. And also the algorithm of LBMM scheduling is used to reduce the completion time of every task on the node. Thereby reducing the total completion time. This algorithm employs to increase the utilization of resources and increases the work efficiency.

i) Biased Random Sampling load balancing Algorithm

Biased Random Sampling Load Balancing Algorithm is a dynamic approach. The network is presented in the form of a virtual graph. Each server is taken as a vertex of the node and the in degree represents the available free resources the nodes have. By the in degree, the load balancer distributes the job to the node. The nodes have at least one in degree then load balancer distributes the job to that node. When the job is allocated to the node then the in degree is decrement by one. And it's get incremented again when the job gets executed. Random sampling technique is used in the addition and deletion of the processes. The processes are centralized by the threshold value, which points the maximum traversal from one node to the destination node. The length of traversal is known as walk length. Surrounding the node of the present node is selected for the traversal. After getting the request, load balancer selects a node loosely and compares the present walk length with the threshold value. If the recent walk length is greater than or equal to the threshold value, the job is completed at that node. Otherwise, the walk length of the job is increased and another neighbor node is selected randomly. The performance decreases as the number of servers increase.

j) Generalized Priority Algorithm

In this algorithm [19] the tasks are prioritized according to the size of the tasks. Such that the task with the highest size gets the highest priority in the system and execute first at its best. Also, the virtual servers are prioritized according to their million instructions per second (MIPS) value in the virtual server distribution system. Such that the Server with the highest MIPS value gets the highest priority. Hence the load balancing is done accordingly and, it gets the maximum utilization of the resources according to the data size in progress.

k) Join-Idle-Queue

Load balancing algorithm of Join-Idle-Queue proposed by Y. Lua et al. [20] which is an algorithm for web services and systems. It simplifies the large scale load balancing with distributed dispatchers. In each dispatch firstly load balancing algorithm idles the processors for the availability. And then does allotment of the task to processors in such a way that reduces the queue length at each server. This algorithm removes the load balancing work from the troublesome path of request processing, which helps in effective reduction of the system load.

l) Genetic Algorithm based Load Balancing

It is introduced by KousikDasgupta and BrototiMandal [21]. This algorithm prospers to balance the load of the cloud infrastructure at the time of trying to shorten span of a job. Genetic-based approach follow some rules and randomization according to the network load effectively.

m) Stochastic Hill Climbing Technique

KousikDasgupta and BrototiMandal proposed [23] a novel load balancing technique by utilizing the algorithm of Stochastic Hill Climbing. It selects randomly

to form the uphill moves with effective possibility-author of a local optimization way Stochastic Hill climbing utilize the resources. And the algorithm is used for distribution of incoming jobs to the virtual machines (VMs) or servers.

n) Decentralized Content Aware Load Balancing

A different content animate load balancing method known as Workload and Client Aware Policy (WCAP) that is proposed by H. Mehta et al. [24] which explain the identical and exceptional feature of the requests. As well as computing nodes by an identical and exceptional feature (USP). USP aids the scheduler in deciding the best and fit resources to execute the process. This technique had less overhead and implemented in a decentralized manner. This technique enhances the searching performance by using the kernel or content information. It also improves the utilization of resources by shortening the lazy time of the computing nodes.

o) Server-Load Balancing for Internet Distributed Services

A. M. Nakai et al. [25] presented a distributed server-based technique for web servers. It facilitates the reduction in service response time by using a protocol that bounds the redirection of requests to the closest remote servers without overloading them. Middleware is used in this technique to implement this protocol. To endure overload, the web server uses Heuristic. Heuristic scheme provides a surety to get the balance of load based on the job size. And also guarantee to not get a repetition of same size job in a single node.

VII. Testing Parameters

Existing load balancing procedures have been measured by the metrics that are discussed below:

a) Throughput

This metric is used to calculate the whole number of tasks, whose completion has been finished successfully. For overall system performance high throughput is needed.

b) Overhead

Overhead combined with any load balancing algorithm prefaces the extra cost engaged in implementing the algorithm. Overhead Associated specifies the amount of overhead involved while completing a load-balancing algorithm. It covers overhead due to movement of tasks, inter processor and inter-process contact. It should be as low as possible.

c) Fault Tolerance

Measuring the ability of an algorithm to execute identical load balancing in case of any shortcoming. It must be highly faulted.

d) Migration Time

It is described as, the entire time needed in migrating the resources or jobs from one node to another. It should be reduced or minimized.

e) Response Time

Response time can be determined as, the time interim between sending a request and getting its response. To boost the overall performance, it should be minimized.

f) Resource Utilization

It is used to assure the accurate utilization of all those resources, which covered the entire system. This issue must be optimized to have an efficient load balancing algorithm.

g) Scalability

It is the ability to perform uniform load balancing in a system with the rise in the number of nodes, according to the requirements. Higher scalability is preferred.

h) Performance

It is used to investigate, how efficient the system is. This has to be uplifted at a reasonable cost, e.g., reducing the response time though keeping the receivable delays.

VIII. Comparison among Different Algorithms

Table 1: Parameters Performance among various algorithms

	Throughput	Response Time	Resource Utilization	Overhead	Fault Tolerance	Scalability	Performance	Migration Time
Round Robin	Yes	Yes	Yes	Yes	No	No	Yes	No
Min Min	Yes	Yes	Yes	Yes	No	No	Yes	No
OLB	No	No	Yes	No	No	No	Yes	No
Max Min	Yes	Yes	Yes	Yes	No	No	Yes	No
Honey Bee	Yes	No	No	No	No	Yes	Yes	No
Active Clustering	No	No	Yes	Yes	No	No	No	Yes
Ant Colony	Yes	No	Yes	No	No	Yes	Yes	Yes
OLB+LBMM	Yes	Yes	Yes	Yes	No	No	Yes	No
Biased Random Sampling	Yes	No	No	No	No	No	Yes	No
Generalized Priority	Yes	No	Yes	No	No	No	No	Yes
Join-Idle-queue	No	No	No	Yes	No	No	Yes	No
Genetic Algorithm Based Load Balancing	No	No	Yes	No	No	No	Yes	No
Stochastic Hill Climbing	Yes	Yes	Yes	No	No	No	Yes	No
Decentralized Content Aware Balancing	No	Yes	Yes	Yes	No	Yes	Yes	No
Server Load Balancing for Internet Distributed Services	No	Yes	No	No	No	Yes	Yes	No

IX. Result & Discussion

From the analysis of various load balancing algorithm, it is very easy to say that OLB + LBMM is better than others. In this paper, we compare load balancing algorithms on several performances measuring metrics. Which helps us to find a conclusion. OLB + LBMM are better than others because it has a great throughput, response time, resource utilization and overhead which determines its performance. On the other hand round robin, max-min, min-min is almost the same as OLB+LBMM. But there is the cute difference between them to OLB+LBMM, which is the percentage of throughput, response time and so on. For this reason, OLB+LBMM is very suitable load balancing algorithm than others. OLB+LBMM have some limitations that it has scalability problems, less capability of fault tolerance and also less migration time.

X. Conclusion & Future Work

In this review, Different types of the algorithm of load balancing are analyzed. Side by side honey bee and round-robin algorithms are properly implemented but, others are not. Various issues are also discussed which must be apprehended into account during designing of new load balancing algorithms. The piece, existing load balancing algorithms are discussed and, comparative analysis is performed by different metrics parameters. Like performance, throughput, scalability, resource utilization, fault tolerance, response time, migration time. In future, we plan to improve this algorithm by consideration of Ant colony optimization and two-phases (OLB + LBMM). Other factors by exploring new effective load balancing algorithm DLB3M which can maintain better balance among parameters and also helps to acquire green computing.

References Références Referencias

1. Rajkumar Buyya, Chee Shin Yeo, Srikumar Venugopal, James Broberg, and Ivona Brandic, “Cloud Computing and Emerging IT Platforms: Vision, Hype, and Reality for Delivering Computing as the 5th Utility, Future Generation Computer Systems”, Volume 25, Number 6, Pages: 599-616, ISSN: 0167-739X, Elsevier Science, Amsterdam, The Netherlands, June 2009.
2. Ram Prasad Padhy and P Goutam Prasad Rao, “Load balancing in cloud computing systems”, Department of Computer Science and Engineering, National Institute of Technology, Rourkela Rourkela-769008, Orissa, India May, 2011.
3. Venubabu Kunamneni, "Dynamic Load Balancing for the cloud", International Journal of Computer Science and Electrical Engineering, 2012.
4. Saroj Hiranwal and Dr. K.C. Roy, “Adaptive Round Robin Scheduling using Shortest Burst Approach Based on Smart Time Slice” International Journal of Computer Science And Communication July-December 2011, Vol. 2, No. 2, Pp. 319-323.
5. Jasmin James and Dr. Bhupendra Verma, “Efficient VM load balancing algorithm for a cloud computing environment”, International Journal on Computer Science and Engineering (IJCSE), 09 Sep 2012.
6. Nusrat Pasha, Dr. Amit Agarwal and Dr.Ravi Rastogi, “Round Robin Approach for VM Load Balancing Algorithm in Cloud Computing Environment” International Journal of Advanced Research in Computer Science and Software Engineering Volume 4, Issue 5, May 2014.
7. Upendra Bhoi, Purvi N. Ramanuj, - Enhanced Max-min Task Scheduling Algorithm in Cloud Computing‖ International Journal of Application or Innovation in Engineering &Management (IJAIEM), Volume 2, Issue 4, April 2013.
8. Fulsoundar, Pritam, and Rajesh Ingle. "Prediction of Performance Degradation in Cloud Computing."
9. Shu-Ching Wang, Kuo-Qin Yan, Wen-Pin Liao, and Shun-ShengWang, "Towards a Load Balancing in a Three-Level Cloud Computing Network,".
10. Hein Nguyen Van, Frederic Dang Tran, and Jean-Marc Menaud, "Performance and Power Management for Cloud Infrastructures," in 2010 IEEE 3rd International Conference on Cloud Computing, 2010.
11. Rahmeh OA, Johnson P, Taleb-Bendiab A.,‖A Dynamic Biased Random Sampling Scheme for scalable and reliable Grid Networks‖, The INFOCOMP Journal of Computer Science, vol. 7, 1-10.
12. Ram Prasad Padhy, P Goutam Prasad Rao, —Load balancing in Cloud Computing Systems‖, National Institute of Technology, Rourkela, India, 2011.
13. Tushar Desai, Jignesh Prajapati, ‖ A Survey of Various Load Balancing Techniques and Challenges in Cloud Computing ‖ International Journal of Scientific & Technology Research, Volume 2, Issue 11, November 2013.
14. Maria Spinola, — An Essential Guide to Possibilities and Risks of Cloud Computing: a Pragmatic Effective and Hype Free Approach for Strategic Enterprise Decision Making‖. (white paper) 2009.
15. Ratan Mishra and Anant Jaiswal, — Ant Colony Optimization: A solution of Load Balancing in Cloud‖, International Journal of Web & Semantic Technology (IJWesT), April 2012.
16. Venubabu Kunamneni, "Dynamic Load Balancing for the cloud", International Journal of Computer Science and Electrical Engineering, 2012.
17. Pooja Samal, Pranati Mishra, ‖Analysis of variants in Round Robin Algorithms for load balancing in Cloud Computing‖ (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 4 (3) , 2013, 416-419.

18. Che-Lun Hung1, Hsiao-hsi Wang2 and Yu-Chen Hu2, Efficient Load Balancing Algorithm for Cloud Computing Network‖. IEEE Vol. 9, pp: 70-78, 2012.
19. Dr. Amit Agarwal, Saloni Jain "Efficient optimal algorithm of task scheduling in cloud computing environment" International Journal of computer Trends and Technology (IJCTT). V9 (7):344-349, March 2014. ISSN: 2231-2803.
20. Kousik Dasgupta, Brototi Mandal, Paramartha Dutta, Jyotsna Kumar Mandal, Santanu Dam, "A Genetic Algorithm (GA) based Load Balancing Strategy for Cloud Computing", Elsevier (CIMTA) 2013.
21. Anamika Jain, Ravinder Singh, "Review of Peer to Peer Grid Load Balancing Model Based on Ant Colony Optimization with Resource Management" Volume 3, Issue 4, April 2013 IJARCSSE.
22. Kousik Dasgupta, Brototi Mandal, Paramartha Dutta, "Load Balancing in Cloud Computing using Stochastic Hill Climbing-A Soft Computing Approach", Elsevier (C3IT) 2012.
23. H. Mehta, P. Kanungo, and M. Chandwani, "Decentralized content aware load balancing algorithm for distributed computing environments", Proceedings of the International Conference Workshop on Emerging Trends in Technology (ICWET), pp.370-375, February 2011.
24. A. M. Nakai, E. Madeira, and L. E. Buzato, "Load Balancing for Internet Distributed Services Using Limited Redirection Rates", 5th IEEE Latin-American Symposium on Dependable Computing (LADC), pp.156- 165 2011.
25. Xi. Liu, Lei. Pan, Chong-Jun. Wang, and JunYuan. Xie, "A Lock-Free Solution for Load Balancing in Multi-Core Environment", 3rd IEEE International Workshop on Intelligent Systems and Applications (ISA), pp.1-4 2011.

Global Journals Guidelines Handbook 2019

www.GlobalJournals.org

FELLOW OF ASSOCIATION OF RESEARCH SOCIETY IN COMPUTING (FARSC)

Global Journals Incorporate (USA) is accredited by Open Association of Research Society (OARS), U.S.A and in turn, awards "FARSC" title to individuals. The 'FARSC' title is accorded to a selected professional after the approval of the Editor-in-Chief/Editorial Board Members/Dean.

- The "FARSC" is a dignified title which is accorded to a person's name viz. Dr. John E. Hall, Ph.D., FARSC or William Walldroff, M.S., FARSC.

FARSC accrediting is an honor. It authenticates your research activities. After recognition as FARSC, you can add 'FARSC' title with your name as you use this recognition as additional suffix to your status. This will definitely enhance and add more value and repute to your name. You may use it on your professional Counseling Materials such as CV, Resume, and Visiting Card etc.

The following benefits can be availed by you only for next three years from the date of certification:

FARSC designated members are entitled to avail a 40% discount while publishing their research papers (of a single author) with Global Journals Incorporation (USA), if the same is accepted by Editorial Board/Peer Reviewers. If you are a main author or co-author in case of multiple authors, you will be entitled to avail discount of 10%.

Once FARSC title is accorded, the Fellow is authorized to organize a symposium/seminar/conference on behalf of Global Journal Incorporation (USA).The Fellow can also participate in conference/seminar/symposium organized by another institution as representative of Global Journal. In both the cases, it is mandatory for him to discuss with us and obtain our consent.

You may join as member of the Editorial Board of Global Journals Incorporation (USA) after successful completion of three years as Fellow and as Peer Reviewer. In addition, it is also desirable that you should organize seminar/symposium/conference at least once.

We shall provide you intimation regarding launching of e-version of journal of your stream time to time. This may be utilized in your library for the enrichment of knowledge of your students as well as it can also be helpful for the concerned faculty members.

The FARSS can go through standards of OARS. You can also play vital role if you have any suggestions so that proper amendment can take place to improve the same for the benefit of entire research community.

As FARSS, you will be given a renowned, secure and free professional email address with 100 GB of space e.g. johnhall@globaljournals.org. This will include Webmail, Spam Assassin, Email Forwarders,Auto-Responders, Email Delivery Route tracing, etc.

The FARSS will be eligible for a free application of standardization of their researches. Standardization of research will be subject to acceptability within stipulated norms as the next step after publishing in a journal. We shall depute a team of specialized research professionals who will render their services for elevating your researches to next higher level, which is worldwide open standardization.

The FARSS member can apply for grading and certification of standards of their educational and Institutional Degrees to Open Association of Research, Society U.S.A.
Once you are designated as FARSS, you may send us a scanned copy of all of your credentials. OARS will verify, grade and certify them. This will be based on your academic records, quality of research papers published by you, and some more criteria. After certification of all your credentials by OARS, they will be published on your Fellow Profile link on website https://associationofresearch.org which will be helpful to upgrade the dignity.

The FARSS members can avail the benefits of free research podcasting in Global Research Radio with their research documents. After publishing the work, (including published elsewhere worldwide with proper authorization) you can upload your research paper with your recorded voice or you can utilize chargeable services of our professional RJs to record your paper in their voice on request.

The FARSS member also entitled to get the benefits of free research podcasting of their research documents through video clips. We can also streamline your conference videos and display your slides/ online slides and online research video clips at reasonable charges, on request.

The FARSS is eligible to earn from sales proceeds of his/her researches/reference/review Books or literature, while publishing with Global Journals. The FARSS can decide whether he/she would like to publish his/her research in a closed manner. In this case, whenever readers purchase that individual research paper for reading, maximum 60% of its profit earned as royalty by Global Journals, will be credited to his/her bank account. The entire entitled amount will be credited to his/her bank account exceeding limit of minimum fixed balance. There is no minimum time limit for collection. The FARSS member can decide its price and we can help in making the right decision.

The FARSS member is eligible to join as a paid peer reviewer at Global Journals Incorporation (USA) and can get remuneration of 15% of author fees, taken from the author of a respective paper. After reviewing 5 or more papers you can request to transfer the amount to your bank account.

MEMBER OF ASSOCIATION OF RESEARCH SOCIETY IN SCIENCE (MARSS)

The ' MARSS ' title is accorded to a selected professional after the approval of the Editor-in-Chief / Editorial Board Members/Dean.

The "MARSS" is a dignified ornament which is accorded to a person's name viz. Dr. John E. Hall, Ph.D., MARSS or William Walldroff, M.S., MARSS.

MARSS accrediting is an honor. It authenticates your research activities. After becoming MARSS, you can add 'MARSS' title with your name as you use this recognition as additional suffix to your status. This will definitely enhance and add more value and repute to your name. You may use it on your professional Counseling Materials such as CV, Resume, Visiting Card and Name Plate etc.

The following benefitscan be availed by you only for next three years from the date of certification.

MARSS designated members are entitled to avail a 25% discount while publishing their research papers (of a single author) in Global Journals Inc., if the same is accepted by our Editorial Board and Peer Reviewers. If you are a main author or co-author of a group of authors, you will get discount of 10%.

As MARSS, you will be given a renowned, secure and free professional email address with 30 GB of space e.g. johnhall@globaljournals.org. This will include Webmail, Spam Assassin, Email Forwarders,Auto-Responders, Email Delivery Route tracing, etc.

We shall provide you intimation regarding launching of e-version of journal of your stream time to time. This may be utilized in your library for the enrichment of knowledge of your students as well as it can also be helpful for the concerned faculty members.

The MARSC member can apply for approval, grading and certification of standards of their educational and Institutional Degrees to Open Association of Research, Society U.S.A.

Once you are designated as MARSC, you may send us a scanned copy of all of your credentials. OARS will verify, grade and certify them. This will be based on your academic records, quality of research papers published by you, and some more criteria.

It is mandatory to read all terms and conditions carefully.

AUXILIARY MEMBERSHIPS

Institutional Fellow of Open Association of Research Society (USA)-OARS (USA)

Global Journals Incorporation (USA) is accredited by Open Association of Research Society, U.S.A (OARS) and in turn, affiliates research institutions as "Institutional Fellow of Open Association of Research Society" (IFOARS).

The "FARSC" is a dignified title which is accorded to a person's name viz. Dr. John E. Hall, Ph.D., FARSC or William Walldroff, M.S., FARSC.

The IFOARS institution is entitled to form a Board comprised of one Chairperson and three to five board members preferably from different streams. The Board will be recognized as "Institutional Board of Open Association of Research Society"-(IBOARS).

The Institute will be entitled to following benefits:

The IBOARS can initially review research papers of their institute and recommend them to publish with respective journal of Global Journals. It can also review the papers of other institutions after obtaining our consent. The second review will be done by peer reviewer of Global Journals Incorporation (USA) The Board is at liberty to appoint a peer reviewer with the approval of chairperson after consulting us.

The author fees of such paper may be waived off up to 40%.

The Global Journals Incorporation (USA) at its discretion can also refer double blind peer reviewed paper at their end to the board for the verification and to get recommendation for final stage of acceptance of publication.

The IBOARS can organize symposium/seminar/conference in their country on behalf of Global Journals Incorporation (USA)-OARS (USA). The terms and conditions can be discussed separately.

The Board can also play vital role by exploring and giving valuable suggestions regarding the Standards of "Open Association of Research Society, U.S.A (OARS)" so that proper amendment can take place for the benefit of entire research community. We shall provide details of particular standard only on receipt of request from the Board.

The board members can also join us as Individual Fellow with 40% discount on total fees applicable to Individual Fellow. They will be entitled to avail all the benefits as declared. Please visit Individual Fellow-sub menu of GlobalJournals.org to have more relevant details.

We shall provide you intimation regarding launching of e-version of journal of your stream time to time. This may be utilized in your library for the enrichment of knowledge of your students as well as it can also be helpful for the concerned faculty members.

After nomination of your institution as "Institutional Fellow" and constantly functioning successfully for one year, we can consider giving recognition to your institute to function as Regional/Zonal office on our behalf.
The board can also take up the additional allied activities for betterment after our consultation.

The following entitlements are applicable to individual Fellows:

Open Association of Research Society, U.S.A (OARS) By-laws states that an individual Fellow may use the designations as applicable, or the corresponding initials. The Credentials of individual Fellow and Associate designations signify that the individual has gained knowledge of the fundamental concepts. One is magnanimous and proficient in an expertise course covering the professional code of conduct, and follows recognized standards of practice.

Open Association of Research Society (US)/ Global Journals Incorporation (USA), as described in Corporate Statements, are educational, research publishing and professional membership organizations. Achieving our individual Fellow or Associate status is based mainly on meeting stated educational research requirements.

Disbursement of 40% Royalty earned through Global Journals : Researcher = 50%, Peer Reviewer = 37.50%, Institution = 12.50% E.g. Out of 40%, the 20% benefit should be passed on to researcher, 15 % benefit towards remuneration should be given to a reviewer and remaining 5% is to be retained by the institution.

We shall provide print version of 12 issues of any three journals [as per your requirement] out of our 38 journals worth $ 2376 USD.

Other:

The individual Fellow and Associate designations accredited by Open Association of Research Society (US) credentials signify guarantees following achievements:

- The professional accredited with Fellow honor, is entitled to various benefits viz. name, fame, honor, regular flow of income, secured bright future, social status etc.

- In addition to above, if one is single author, then entitled to 40% discount on publishing research paper and can get 10%discount if one is co-author or main author among group of authors.
- The Fellow can organize symposium/seminar/conference on behalf of Global Journals Incorporation (USA) and he/she can also attend the same organized by other institutes on behalf of Global Journals.
- The Fellow can become member of Editorial Board Member after completing 3yrs.
- The Fellow can earn 60% of sales proceeds from the sale of reference/review books/literature/publishing of research paper.
- Fellow can also join as paid peer reviewer and earn 15% remuneration of author charges and can also get an opportunity to join as member of the Editorial Board of Global Journals Incorporation (USA)
- • This individual has learned the basic methods of applying those concepts and techniques to common challenging situations. This individual has further demonstrated an in–depth understanding of the application of suitable techniques to a particular area of research practice.

Note :

"

- In future, if the board feels the necessity to change any board member, the same can be done with the consent of the chairperson along with anyone board member without our approval.
- In case, the chairperson needs to be replaced then consent of 2/3rd board members are required and they are also required to jointly pass the resolution copy of which should be sent to us. In such case, it will be compulsory to obtain our approval before replacement.
- In case of "Difference of Opinion [if any]" among the Board members, our decision will be final and binding to everyone.

"

Preferred Author Guidelines

We accept the manuscript submissions in any standard (generic) format.

We typeset manuscripts using advanced typesetting tools like Adobe In Design, CorelDraw, TeXnicCenter, and TeXStudio. We usually recommend authors submit their research using any standard format they are comfortable with, and let Global Journals do the rest.

Alternatively, you can download our basic template from https://globaljournals.org/Template.zip

Authors should submit their complete paper/article, including text illustrations, graphics, conclusions, artwork, and tables. Authors who are not able to submit manuscript using the form above can email the manuscript department at submit@globaljournals.org or get in touch with chiefeditor@globaljournals.org if they wish to send the abstract before submission.

Before and during Submission

Authors must ensure the information provided during the submission of a paper is authentic. Please go through the following checklist before submitting:

1. Authors must go through the complete author guideline and understand and *agree to Global Journals' ethics and code of conduct,* along with author responsibilities.
2. Authors must accept the privacy policy, terms, and conditions of Global Journals.
3. Ensure corresponding author's email address and postal address are accurate and reachable.
4. Manuscript to be submitted must include keywords, an abstract, a paper title, co-author(s') names and details (email address, name, phone number, and institution), figures and illustrations in vector format including appropriate captions, tables, including titles and footnotes, a conclusion, results, acknowledgments and references.
5. Authors should submit paper in a ZIP archive if any supplementary files are required along with the paper.
6. Proper permissions must be acquired for the use of any copyrighted material.
7. Manuscript submitted *must not have been submitted or published elsewhere* and all authors must be aware of the submission.

Declaration of Conflicts of Interest

It is required for authors to declare all financial, institutional, and personal relationships with other individuals and organizations that could influence (bias) their research.

Policy on Plagiarism

Plagiarism is not acceptable in Global Journals submissions at all.

Plagiarized content will not be considered for publication. We reserve the right to inform authors' institutions about plagiarism detected either before or after publication. If plagiarism is identified, we will follow COPE guidelines:

Authors are solely responsible for all the plagiarism that is found. The author must not fabricate, falsify or plagiarize existing research data. The following, if copied, will be considered plagiarism:

- Words (language)
- Ideas
- Findings
- Writings
- Diagrams
- Graphs
- Illustrations
- Lectures

- Printed material
- Graphic representations
- Computer programs
- Electronic material
- Any other original work

Authorship Policies

Global Journals follows the definition of authorship set up by the Open Association of Research Society, USA. According to its guidelines, authorship criteria must be based on:

1. Substantial contributions to the conception and acquisition of data, analysis, and interpretation of findings.
2. Drafting the paper and revising it critically regarding important academic content.
3. Final approval of the version of the paper to be published.

Changes in Authorship

The corresponding author should mention the name and complete details of all co-authors during submission and in manuscript. We support addition, rearrangement, manipulation, and deletions in authors list till the early view publication of the journal. We expect that corresponding author will notify all co-authors of submission. We follow COPE guidelines for changes in authorship.

Copyright

During submission of the manuscript, the author is confirming an exclusive license agreement with Global Journals which gives Global Journals the authority to reproduce, reuse, and republish authors' research. We also believe in flexible copyright terms where copyright may remain with authors/employers/institutions as well. Contact your editor after acceptance to choose your copyright policy. You may follow this form for copyright transfers.

Appealing Decisions

Unless specified in the notification, the Editorial Board's decision on publication of the paper is final and cannot be appealed before making the major change in the manuscript.

Acknowledgments

Contributors to the research other than authors credited should be mentioned in Acknowledgments. The source of funding for the research can be included. Suppliers of resources may be mentioned along with their addresses.

Declaration of funding sources

Global Journals is in partnership with various universities, laboratories, and other institutions worldwide in the research domain. Authors are requested to disclose their source of funding during every stage of their research, such as making analysis, performing laboratory operations, computing data, and using institutional resources, from writing an article to its submission. This will also help authors to get reimbursements by requesting an open access publication letter from Global Journals and submitting to the respective funding source.

Preparing your Manuscript

Authors can submit papers and articles in an acceptable file format: MS Word (doc, docx), LaTeX (.tex, .zip or .rar including all of your files), Adobe PDF (.pdf), rich text format (.rtf), simple text document (.txt), Open Document Text (.odt), and Apple Pages (.pages). Our professional layout editors will format the entire paper according to our official guidelines. This is one of the highlights of publishing with Global Journals—authors should not be concerned about the formatting of their paper. Global Journals accepts articles and manuscripts in every major language, be it Spanish, Chinese, Japanese, Portuguese, Russian, French, German, Dutch, Italian, Greek, or any other national language, but the title, subtitle, and abstract should be in English. This will facilitate indexing and the pre-peer review process.

The following is the official style and template developed for publication of a research paper. Authors are not required to follow this style during the submission of the paper. It is just for reference purposes.

Manuscript Style Instruction (Optional)

- Microsoft Word Document Setting Instructions.
- Font type of all text should be Swis721 Lt BT.
- Page size: 8.27" x 11'", left margin: 0.65, right margin: 0.65, bottom margin: 0.75.
- Paper title should be in one column of font size 24.
- Author name in font size of 11 in one column.
- Abstract: font size 9 with the word "Abstract" in bold italics.
- Main text: font size 10 with two justified columns.
- Two columns with equal column width of 3.38 and spacing of 0.2.
- First character must be three lines drop-capped.
- The paragraph before spacing of 1 pt and after of 0 pt.
- Line spacing of 1 pt.
- Large images must be in one column.
- The names of first main headings (Heading 1) must be in Roman font, capital letters, and font size of 10.
- The names of second main headings (Heading 2) must not include numbers and must be in italics with a font size of 10.

Structure and Format of Manuscript

The recommended size of an original research paper is under 15,000 words and review papers under 7,000 words. Research articles should be less than 10,000 words. Research papers are usually longer than review papers. Review papers are reports of significant research (typically less than 7,000 words, including tables, figures, and references)

A research paper must include:

a) A title which should be relevant to the theme of the paper.
b) A summary, known as an abstract (less than 150 words), containing the major results and conclusions.
c) Up to 10 keywords that precisely identify the paper's subject, purpose, and focus.
d) An introduction, giving fundamental background objectives.
e) Resources and techniques with sufficient complete experimental details (wherever possible by reference) to permit repetition, sources of information must be given, and numerical methods must be specified by reference.
f) Results which should be presented concisely by well-designed tables and figures.
g) Suitable statistical data should also be given.
h) All data must have been gathered with attention to numerical detail in the planning stage.

Design has been recognized to be essential to experiments for a considerable time, and the editor has decided that any paper that appears not to have adequate numerical treatments of the data will be returned unrefereed.

i) Discussion should cover implications and consequences and not just recapitulate the results; conclusions should also be summarized.
j) There should be brief acknowledgments.
k) There ought to be references in the conventional format. Global Journals recommends APA format.

Authors should carefully consider the preparation of papers to ensure that they communicate effectively. Papers are much more likely to be accepted if they are carefully designed and laid out, contain few or no errors, are summarizing, and follow instructions. They will also be published with much fewer delays than those that require much technical and editorial correction.

The Editorial Board reserves the right to make literary corrections and suggestions to improve brevity.

Format Structure

It is necessary that authors take care in submitting a manuscript that is written in simple language and adheres to published guidelines.

All manuscripts submitted to Global Journals should include:

Title

The title page must carry an informative title that reflects the content, a running title (less than 45 characters together with spaces), names of the authors and co-authors, and the place(s) where the work was carried out.

Author details

The full postal address of any related author(s) must be specified.

Abstract

The abstract is the foundation of the research paper. It should be clear and concise and must contain the objective of the paper and inferences drawn. It is advised to not include big mathematical equations or complicated jargon.

Many researchers searching for information online will use search engines such as Google, Yahoo or others. By optimizing your paper for search engines, you will amplify the chance of someone finding it. In turn, this will make it more likely to be viewed and cited in further works. Global Journals has compiled these guidelines to facilitate you to maximize the web-friendliness of the most public part of your paper.

Keywords

A major lynchpin of research work for the writing of research papers is the keyword search, which one will employ to find both library and internet resources. Up to eleven keywords or very brief phrases have to be given to help data retrieval, mining, and indexing.

One must be persistent and creative in using keywords. An effective keyword search requires a strategy: planning of a list of possible keywords and phrases to try.

Choice of the main keywords is the first tool of writing a research paper. Research paper writing is an art. Keyword search should be as strategic as possible.

One should start brainstorming lists of potential keywords before even beginning searching. Think about the most important concepts related to research work. Ask, “What words would a source have to include to be truly valuable in a research paper?” Then consider synonyms for the important words.

It may take the discovery of only one important paper to steer in the right keyword direction because, in most databases, the keywords under which a research paper is abstracted are listed with the paper.

Numerical Methods

Numerical methods used should be transparent and, where appropriate, supported by references.

Abbreviations

Authors must list all the abbreviations used in the paper at the end of the paper or in a separate table before using them.

Formulas and equations

Authors are advised to submit any mathematical equation using either MathJax, KaTeX, or LaTeX, or in a very high-quality image.

Tables, Figures, and Figure Legends

Tables: Tables should be cautiously designed, uncrowned, and include only essential data. Each must have an Arabic number, e.g., Table 4, a self-explanatory caption, and be on a separate sheet. Authors must submit tables in an editable format and not as images. References to these tables (if any) must be mentioned accurately.

Figures

Figures are supposed to be submitted as separate files. Always include a citation in the text for each figure using Arabic numbers, e.g., Fig. 4. Artwork must be submitted online in vector electronic form or by emailing it.

Preparation of Eletronic Figures for Publication

Although low-quality images are sufficient for review purposes, print publication requires high-quality images to prevent the final product being blurred or fuzzy. Submit (possibly by e-mail) EPS (line art) or TIFF (halftone/ photographs) files only. MS PowerPoint and Word Graphics are unsuitable for printed pictures. Avoid using pixel-oriented software. Scans (TIFF only) should have a resolution of at least 350 dpi (halftone) or 700 to 1100 dpi (line drawings). Please give the data for figures in black and white or submit a Color Work Agreement form. EPS files must be saved with fonts embedded (and with a TIFF preview, if possible).

For scanned images, the scanning resolution at final image size ought to be as follows to ensure good reproduction: line art: >650 dpi; halftones (including gel photographs): >350 dpi; figures containing both halftone and line images: >650 dpi.

Color charges: Authors are advised to pay the full cost for the reproduction of their color artwork. Hence, please note that if there is color artwork in your manuscript when it is accepted for publication, we would require you to complete and return a Color Work Agreement form before your paper can be published. Also, you can email your editor to remove the color fee after acceptance of the paper.

Tips for writing a good quality Computer Science Research Paper

Techniques for writing a good quality computer science research paper:

1. Choosing the topic: In most cases, the topic is selected by the interests of the author, but it can also be suggested by the guides. You can have several topics, and then judge which you are most comfortable with. This may be done by asking several questions of yourself, like "Will I be able to carry out a search in this area? Will I find all necessary resources to accomplish the search? Will I be able to find all information in this field area?" If the answer to this type of question is "yes," then you ought to choose that topic. In most cases, you may have to conduct surveys and visit several places. Also, you might have to do a lot of work to find all the rises and falls of the various data on that subject. Sometimes, detailed information plays a vital role, instead of short information. Evaluators are human: The first thing to remember is that evaluators are also human beings. They are not only meant for rejecting a paper. They are here to evaluate your paper. So present your best aspect.

2. Think like evaluators: If you are in confusion or getting demotivated because your paper may not be accepted by the evaluators, then think, and try to evaluate your paper like an evaluator. Try to understand what an evaluator wants in your research paper, and you will automatically have your answer. Make blueprints of paper: The outline is the plan or framework that will help you to arrange your thoughts. It will make your paper logical. But remember that all points of your outline must be related to the topic you have chosen.

3. Ask your guides: If you are having any difficulty with your research, then do not hesitate to share your difficulty with your guide (if you have one). They will surely help you out and resolve your doubts. If you can't clarify what exactly you require for your work, then ask your supervisor to help you with an alternative. He or she might also provide you with a list of essential readings.

4. Use of computer is recommended: As you are doing research in the field of computer science then this point is quite obvious. Use right software: Always use good quality software packages. If you are not capable of judging good software, then you can lose the quality of your paper unknowingly. There are various programs available to help you which you can get through the internet.

5. Use the internet for help: An excellent start for your paper is using Google. It is a wondrous search engine, where you can have your doubts resolved. You may also read some answers for the frequent question of how to write your research paper or find a model research paper. You can download books from the internet. If you have all the required books, place importance on reading, selecting, and analyzing the specified information. Then sketch out your research paper. Use big pictures: You may use encyclopedias like Wikipedia to get pictures with the best resolution. At Global Journals, you should strictly follow here.

6. Bookmarks are useful: When you read any book or magazine, you generally use bookmarks, right? It is a good habit which helps to not lose your continuity. You should always use bookmarks while searching on the internet also, which will make your search easier.

7. Revise what you wrote: When you write anything, always read it, summarize it, and then finalize it.

8. Make every effort: Make every effort to mention what you are going to write in your paper. That means always have a good start. Try to mention everything in the introduction—what is the need for a particular research paper. Polish your work with good writing skills and always give an evaluator what he wants. Make backups: When you are going to do any important thing like making a research paper, you should always have backup copies of it either on your computer or on paper. This protects you from losing any portion of your important data.

9. Produce good diagrams of your own: Always try to include good charts or diagrams in your paper to improve quality. Using several unnecessary diagrams will degrade the quality of your paper by creating a hodgepodge. So always try to include diagrams which were made by you to improve the readability of your paper. Use of direct quotes: When you do research relevant to literature, history, or current affairs, then use of quotes becomes essential, but if the study is relevant to science, use of quotes is not preferable.

10.Use proper verb tense: Use proper verb tenses in your paper. Use past tense to present those events that have happened. Use present tense to indicate events that are going on. Use future tense to indicate events that will happen in the future. Use of wrong tenses will confuse the evaluator. Avoid sentences that are incomplete.

11. Pick a good study spot: Always try to pick a spot for your research which is quiet. Not every spot is good for studying.

12. Know what you know: Always try to know what you know by making objectives, otherwise you will be confused and unable to achieve your target.

13. Use good grammar: Always use good grammar and words that will have a positive impact on the evaluator; use of good vocabulary does not mean using tough words which the evaluator has to find in a dictionary. Do not fragment sentences. Eliminate one-word sentences. Do not ever use a big word when a smaller one would suffice.

Verbs have to be in agreement with their subjects. In a research paper, do not start sentences with conjunctions or finish them with prepositions. When writing formally, it is advisable to never split an infinitive because someone will (wrongly) complain. Avoid clichés like a disease. Always shun irritating alliteration. Use language which is simple and straightforward. Put together a neat summary.

14. Arrangement of information: Each section of the main body should start with an opening sentence, and there should be a changeover at the end of the section. Give only valid and powerful arguments for your topic. You may also maintain your arguments with records.

15. Never start at the last minute: Always allow enough time for research work. Leaving everything to the last minute will degrade your paper and spoil your work.

16. Multitasking in research is not good: Doing several things at the same time is a bad habit in the case of research activity. Research is an area where everything has a particular time slot. Divide your research work into parts, and do a particular part in a particular time slot.

17. Never copy others' work: Never copy others' work and give it your name because if the evaluator has seen it anywhere, you will be in trouble. Take proper rest and food: No matter how many hours you spend on your research activity, if you are not taking care of your health, then all your efforts will have been in vain. For quality research, take proper rest and food.

18. Go to seminars: Attend seminars if the topic is relevant to your research area. Utilize all your resources.

19. Refresh your mind after intervals: Try to give your mind a rest by listening to soft music or sleeping in intervals. This will also improve your memory. Acquire colleagues: Always try to acquire colleagues. No matter how sharp you are, if you acquire colleagues, they can give you ideas which will be helpful to your research.

20. Think technically: Always think technically. If anything happens, search for its reasons, benefits, and demerits. Think and then print: When you go to print your paper, check that tables are not split, headings are not detached from their descriptions, and page sequence is maintained.

21. Adding unnecessary information: Do not add unnecessary information like "I have used MS Excel to draw graphs." Irrelevant and inappropriate material is superfluous. Foreign terminology and phrases are not apropos. One should never take a broad view. Analogy is like feathers on a snake. Use words properly, regardless of how others use them. Remove quotations. Puns are for kids, not grunt readers. Never oversimplify: When adding material to your research paper, never go for oversimplification; this will definitely irritate the evaluator. Be specific. Never use rhythmic redundancies. Contractions shouldn't be used in a research paper. Comparisons are as terrible as clichés. Give up ampersands, abbreviations, and so on. Remove commas that are not necessary. Parenthetical words should be between brackets or commas. Understatement is always the best way to put forward earth-shaking thoughts. Give a detailed literary review.

22. Report concluded results: Use concluded results. From raw data, filter the results, and then conclude your studies based on measurements and observations taken. An appropriate number of decimal places should be used. Parenthetical remarks are prohibited here. Proofread carefully at the final stage. At the end, give an outline to your arguments. Spot perspectives of further study of the subject. Justify your conclusion at the bottom sufficiently, which will probably include examples.

23. Upon conclusion: Once you have concluded your research, the next most important step is to present your findings. Presentation is extremely important as it is the definite medium though which your research is going to be in print for the rest of the crowd. Care should be taken to categorize your thoughts well and present them in a logical and neat manner. A good quality research paper format is essential because it serves to highlight your research paper and bring to light all necessary aspects of your research.

Informal Guidelines of Research Paper Writing

Key points to remember:

- Submit all work in its final form.
- Write your paper in the form which is presented in the guidelines using the template.
- Please note the criteria peer reviewers will use for grading the final paper.

Final points:

One purpose of organizing a research paper is to let people interpret your efforts selectively. The journal requires the following sections, submitted in the order listed, with each section starting on a new page:

The introduction: This will be compiled from reference matter and reflect the design processes or outline of basis that directed you to make a study. As you carry out the process of study, the method and process section will be constructed like that. The results segment will show related statistics in nearly sequential order and direct reviewers to similar intellectual paths throughout the data that you gathered to carry out your study.

The discussion section:

This will provide understanding of the data and projections as to the implications of the results. The use of good quality references throughout the paper will give the effort trustworthiness by representing an alertness to prior workings.

Writing a research paper is not an easy job, no matter how trouble-free the actual research or concept. Practice, excellent preparation, and controlled record-keeping are the only means to make straightforward progression.

General style:

Specific editorial column necessities for compliance of a manuscript will always take over from directions in these general guidelines.

To make a paper clear: Adhere to recommended page limits.

Mistakes to avoid:

- Insertion of a title at the foot of a page with subsequent text on the next page.
- Separating a table, chart, or figure—confine each to a single page.
- Submitting a manuscript with pages out of sequence.
- In every section of your document, use standard writing style, including articles ("a" and "the").
- Keep paying attention to the topic of the paper.
- Use paragraphs to split each significant point (excluding the abstract).
- Align the primary line of each section.
- Present your points in sound order.
- Use present tense to report well-accepted matters.
- Use past tense to describe specific results.
- Do not use familiar wording; don't address the reviewer directly. Don't use slang or superlatives.
- Avoid use of extra pictures—include only those figures essential to presenting results.

Title page:

Choose a revealing title. It should be short and include the name(s) and address(es) of all authors. It should not have acronyms or abbreviations or exceed two printed lines.

Abstract: This summary should be two hundred words or less. It should clearly and briefly explain the key findings reported in the manuscript and must have precise statistics. It should not have acronyms or abbreviations. It should be logical in itself. Do not cite references at this point.

An abstract is a brief, distinct paragraph summary of finished work or work in development. In a minute or less, a reviewer can be taught the foundation behind the study, common approaches to the problem, relevant results, and significant conclusions or new questions.

Write your summary when your paper is completed because how can you write the summary of anything which is not yet written? Wealth of terminology is very essential in abstract. Use comprehensive sentences, and do not sacrifice readability for brevity; you can maintain it succinctly by phrasing sentences so that they provide more than a lone rationale. The author can at this moment go straight to shortening the outcome. Sum up the study with the subsequent elements in any summary. Try to limit the initial two items to no more than one line each.

Reason for writing the article—theory, overall issue, purpose.

- Fundamental goal.
- To-the-point depiction of the research.
- Consequences, including definite statistics—if the consequences are quantitative in nature, account for this; results of any numerical analysis should be reported. Significant conclusions or questions that emerge from the research.

Approach:

- Single section and succinct.
- An outline of the job done is always written in past tense.
- Concentrate on shortening results—limit background information to a verdict or two.
- Exact spelling, clarity of sentences and phrases, and appropriate reporting of quantities (proper units, important statistics) are just as significant in an abstract as they are anywhere else.

Introduction:

The introduction should "introduce" the manuscript. The reviewer should be presented with sufficient background information to be capable of comprehending and calculating the purpose of your study without having to refer to other works. The basis for the study should be offered. Give the most important references, but avoid making a comprehensive appraisal of the topic. Describe the problem visibly. If the problem is not acknowledged in a logical, reasonable way, the reviewer will give no attention to your results. Speak in common terms about techniques used to explain the problem, if needed, but do not present any particulars about the protocols here.

The following approach can create a valuable beginning:

- Explain the value (significance) of the study.
- Defend the model—why did you employ this particular system or method? What is its compensation? Remark upon its appropriateness from an abstract point of view as well as pointing out sensible reasons for using it.
- Present a justification. State your particular theory(-ies) or aim(s), and describe the logic that led you to choose them.
- Briefly explain the study's tentative purpose and how it meets the declared objectives.

Approach:

Use past tense except for when referring to recognized facts. After all, the manuscript will be submitted after the entire job is done. Sort out your thoughts; manufacture one key point for every section. If you make the four points listed above, you will need at least four paragraphs. Present surrounding information only when it is necessary to support a situation. The reviewer does not desire to read everything you know about a topic. Shape the theory specifically—do not take a broad view.

As always, give awareness to spelling, simplicity, and correctness of sentences and phrases.

Procedures (methods and materials):

This part is supposed to be the easiest to carve if you have good skills. A soundly written procedures segment allows a capable scientist to replicate your results. Present precise information about your supplies. The suppliers and clarity of reagents can be helpful bits of information. Present methods in sequential order, but linked methodologies can be grouped as a segment. Be concise when relating the protocols. Attempt to give the least amount of information that would permit another capable scientist to replicate your outcome, but be cautious that vital information is integrated. The use of subheadings is suggested and ought to be synchronized with the results section.

When a technique is used that has been well-described in another section, mention the specific item describing the way, but draw the basic principle while stating the situation. The purpose is to show all particular resources and broad procedures so that another person may use some or all of the methods in one more study or referee the scientific value of your work. It is not to be a step-by-step report of the whole thing you did, nor is a methods section a set of orders.

Materials:

Materials may be reported in part of a section or else they may be recognized along with your measures.

Methods:

- Report the method and not the particulars of each process that engaged the same methodology.
- Describe the method entirely.
- To be succinct, present methods under headings dedicated to specific dealings or groups of measures.
- Simplify—detail how procedures were completed, not how they were performed on a particular day.
- If well-known procedures were used, account for the procedure by name, possibly with a reference, and that's all.

Approach:

It is embarrassing to use vigorous voice when documenting methods without using first person, which would focus the reviewer's interest on the researcher rather than the job. As a result, when writing up the methods, most authors use third person passive voice.

Use standard style in this and every other part of the paper—avoid familiar lists, and use full sentences.

What to keep away from:

- Resources and methods are not a set of information.
- Skip all descriptive information and surroundings—save it for the argument.
- Leave out information that is immaterial to a third party.

Results:

The principle of a results segment is to present and demonstrate your conclusion. Create this part as entirely objective details of the outcome, and save all understanding for the discussion.

The page length of this segment is set by the sum and types of data to be reported. Use statistics and tables, if suitable, to present consequences most efficiently.

You must clearly differentiate material which would usually be incorporated in a study editorial from any unprocessed data or additional appendix matter that would not be available. In fact, such matters should not be submitted at all except if requested by the instructor.

Content:

- Sum up your conclusions in text and demonstrate them, if suitable, with figures and tables.
- In the manuscript, explain each of your consequences, and point the reader to remarks that are most appropriate.
- Present a background, such as by describing the question that was addressed by creation of an exacting study.
- Explain results of control experiments and give remarks that are not accessible in a prescribed figure or table, if appropriate.
- Examine your data, then prepare the analyzed (transformed) data in the form of a figure (graph), table, or manuscript.

What to stay away from:

- Do not discuss or infer your outcome, report surrounding information, or try to explain anything.
- Do not include raw data or intermediate calculations in a research manuscript.
- Do not present similar data more than once.
- A manuscript should complement any figures or tables, not duplicate information.
- Never confuse figures with tables—there is a difference.

Approach:

As always, use past tense when you submit your results, and put the whole thing in a reasonable order.

Put figures and tables, appropriately numbered, in order at the end of the report.

If you desire, you may place your figures and tables properly within the text of your results section.

Figures and tables:

If you put figures and tables at the end of some details, make certain that they are visibly distinguished from any attached appendix materials, such as raw facts. Whatever the position, each table must be titled, numbered one after the other, and include a heading. All figures and tables must be divided from the text.

Discussion:

The discussion is expected to be the trickiest segment to write. A lot of papers submitted to the journal are discarded based on problems with the discussion. There is no rule for how long an argument should be.

Position your understanding of the outcome visibly to lead the reviewer through your conclusions, and then finish the paper with a summing up of the implications of the study. The purpose here is to offer an understanding of your results and support all of your conclusions, using facts from your research and generally accepted information, if suitable. The implication of results should be fully described.

Infer your data in the conversation in suitable depth. This means that when you clarify an observable fact, you must explain mechanisms that may account for the observation. If your results vary from your prospect, make clear why that may have happened. If your results agree, then explain the theory that the proof supported. It is never suitable to just state that the data approved the prospect, and let it drop at that. Make a decision as to whether each premise is supported or discarded or if you cannot make a conclusion with assurance. Do not just dismiss a study or part of a study as "uncertain."

Research papers are not acknowledged if the work is imperfect. Draw what conclusions you can based upon the results that you have, and take care of the study as a finished work.

- You may propose future guidelines, such as how an experiment might be personalized to accomplish a new idea.
- Give details of all of your remarks as much as possible, focusing on mechanisms.
- Make a decision as to whether the tentative design sufficiently addressed the theory and whether or not it was correctly restricted. Try to present substitute explanations if they are sensible alternatives.
- One piece of research will not counter an overall question, so maintain the large picture in mind. Where do you go next? The best studies unlock new avenues of study. What questions remain?
- Recommendations for detailed papers will offer supplementary suggestions.

Approach:

When you refer to information, differentiate data generated by your own studies from other available information. Present work done by specific persons (including you) in past tense.

Describe generally acknowledged facts and main beliefs in present tense.

The Administration Rules

Administration Rules to Be Strictly Followed before Submitting Your Research Paper to Global Journals Inc.

Please read the following rules and regulations carefully before submitting your research paper to Global Journals Inc. to avoid rejection.

Segment draft and final research paper: You have to strictly follow the template of a research paper, failing which your paper may get rejected. You are expected to write each part of the paper wholly on your own. The peer reviewers need to identify your own perspective of the concepts in your own terms. Please do not extract straight from any other source, and do not rephrase someone else's analysis. Do not allow anyone else to proofread your manuscript.

Written material: You may discuss this with your guides and key sources. Do not copy anyone else's paper, even if this is only imitation, otherwise it will be rejected on the grounds of plagiarism, which is illegal. Various methods to avoid plagiarism are strictly applied by us to every paper, and, if found guilty, you may be blacklisted, which could affect your career adversely. To guard yourself and others from possible illegal use, please do not permit anyone to use or even read your paper and file.

Criterion for Grading a Research Paper (Compilation) by Global Journals Inc. (US)

Please note that following table is only a Grading of "Paper Compilation" and not on "Performed/Stated Research" whose grading solely depends on Individual Assigned Peer Reviewer and Editorial Board Member. These can be available only on request and after decision of Paper. This report will be the property of Global Journals Inc. (US).

Topics	Grades		
	A-B	**C-D**	**E-F**
Abstract	Clear and concise with appropriate content, Correct format. 200 words or below	Unclear summary and no specific data, Incorrect form Above 200 words	No specific data with ambiguous information Above 250 words
Introduction	Containing all background details with clear goal and appropriate details, flow specification, no grammar and spelling mistake, well organized sentence and paragraph, reference cited	Unclear and confusing data, appropriate format, grammar and spelling errors with unorganized matter	Out of place depth and content, hazy format
Methods and Procedures	Clear and to the point with well arranged paragraph, precision and accuracy of facts and figures, well organized subheads	Difficult to comprehend with embarrassed text, too much explanation but completed	Incorrect and unorganized structure with hazy meaning
Result	Well organized, Clear and specific, Correct units with precision, correct data, well structuring of paragraph, no grammar and spelling mistake	Complete and embarrassed text, difficult to comprehend	Irregular format with wrong facts and figures
Discussion	Well organized, meaningful specification, sound conclusion, logical and concise explanation, highly structured paragraph reference cited	Wordy, unclear conclusion, spurious	Conclusion is not cited, unorganized, difficult to comprehend
References	Complete and correct format, well organized	Beside the point, Incomplete	Wrong format and structuring

Index

A

Abstraction · 3, 69
Actualize · 25
Arsenal · 26
Asymmetric · 20

C

Cambridge · 65
Circumstance · 23
Contingent · 11
Convenient · 3, 7, 8
Cryptography · 16, 20

D

Decrypted · 17
Decryption · 16, 17, 19
Deemed · 63

E

Embedded · 24
Empting · 61
Encyclopaedias · 7
Entangles · 23
Essence · 16

H

Hierarchical · 27, 42, 46, 57

I

Immerse · 48
Inherent · 1
Intertwined · 54, 62, 65

J

Jurisdiction · 14

M

Masonry · 27
Monolithic · 49, 52

N

Natively · 45
Nullified · 53

P

Partitioning · 51
Pioneering · 46
Premises · 5

Q

Quotas · 59

S

Scenario · 1, 46, 47
Serializers · 57
Shattered · 12
Signifies · 54
Sophisticated · 49
Surveillance · 40

T

Tierney · 8

save our planet

Global Journal of Computer Science and Technology

Visit us on the Web at www.GlobalJournals.org | www.ComputerResearch.org
or email us at helpdesk@globaljournals.org

ISSN 9754350