ISSN (Print) : 0974-6846
Indian Journal of Science and Technology, *Vol 9(11), DOI: 10.17485/ijst/2016/v9i11/82635, March 2016* ISSN (Online) : 0974-5645

Comparative Analysis on Load Balancing Techniques in Cloud Computing

Amit Verma[1*], Sanjay Kumar Sharma[2] and Bhavneet Kaur[1]

[1]Department of Computer Science, Chandigarh Engineering College, Landran, Mohali – 140307, Punjab, India; dramitverma.cu@gmail.com, bhavneet14kaur@gmail.com
[2]Department of Computer Science, Chandigarh University, Gharuan (Mohali) - 160036, Punjab, India; talk4sanjaysharma@gmail.com

Abstract

Background/Objectives: Cloud computing is an arena that is ruling the world of information technology. Every user has its own definition for this technology as per their use. This paper is properly discussed document that describes the complete evolution of cloud computing from its beginning. **Findings:** With the presence of vast literature in field of load balancing, it was found confusion for the new scholars to find the startup point for their research in this field. Therefore, an exhaustive comparison has been made for the superior understanding of cloud evolution through various proposed algorithms from the past many decades, which will make the researchers possible to analyze the existing scenarios and a better way out to overcome the unsolved queries. **Application/Improvements:** The assessments between the algorithms will help the new researchers to analyze and opt for the parameters those need much more concentration to meet the required targets for better outcomes in the field.

Keywords: Cloud Computing, Information Extraction and Performance Measure, Load Balancing

1. Introduction

Computing, a term originated from the term compute which describes a way to attain the results from corresponding task a user had performed. Therefore, we can define computing as an attaining of results from evaluating the problem. Civilization of computer society is always on its peak in terms of the development. The most emerging technology of current era on which massive investigations and drastic progressions are coming is cloud computing. It is a drastic development in field of computing which has been emerged by witnessing the development that followed the pathway from distributing computing[1] to parallel computing[2], followed by cluster computing[3,4] and grid computing[5,6]. Therefore, cloud computing is considered as a striking computing model which allows for the provisioning of resources on-demand[7], which in a layman language can be described as a virtual space in which users are allotted some space for storage which is accessible by the user ID and password that is actually their private space but along with this there is another kind of space, where data present on it has open access to people for its usage. The live example of who is Google. Table 1 is presented that describes the tabular representation of the evolution in technology and network environment in computing. This paper we are going to study the algorithmic evolution of load balancing in computing.

Table 1. Evolution of Technology

S No.	Era	Technology	Network Environment
1	From 1950	ECL	SNA
2	1950 - 1970	Custom Bipolar	SNA, DEC
3	1970 - 1980	CMOS Micro	TCP/IP and NFS
4	1980 - 1990	CMOS Micro	TCP/IP and NFS
5	1990 - till date	X86 Micro	Netware

* *Author for correspondence*

2. Evolution Study via Algorithms

One of the fruitful ways to not only improve the throughput of systems but also enables the effective usage of resources and reduction in response time can be done by the load balancing[8]. Many algorithms were designed to solve various issues related to computing but here we are going to discuss the evolution of the load balancing for various computing environments. Firstly, Table 2 presented here describes the comparisons between the few selected algorithms from different years of development.

Table 2. Comparison between existing scheduling algorithm

Title	Parameters	Findings	Environment	Tools
LBDMC	$C_{pattern}$, W_{load}, A_{gnt}, H_{st}	Th_{hold}, L_{red}, S_{cb}	D_{Evnt}	JDK
MQCSS	SS_{rate}, $T_{m,}$ C_{t}, $M_{span,}$	D_{wrkflw}, C_{exec} T_{exec}	C_{Envt}	C_{Sim}

DFD Level 0 is used to represent the basic overview of the system similarly, the DFD level 0 for evolution of technology describes that the information will be extracted from cloud by using various techniques like clustering, gridding, scheduling and many more.

Table 3 represents the DFD level 0 of technology evolution for load balancing in cloud computing.

Level 1 DFD for evolution of technology describes that to extract the information for load balancing in cloud would only be possible by using any sort of technique like clustering, scheduling, process migration, mark span, gridding on the data and along with the use of techniques the tool is required for the data extraction accordingly.

Table 4 shows the DFD Level 1 respectively.

DFD Level 2 is that level of data flow diagram that describes the detail prototype of complete system. Like as in Table 5 that displays the information extraction from cloud for load balancing. Here it has been described that the techniques used for load balancing are clustering, scheduling, process migration, mark span and gridding. And correspondingly the tools used on these techniques are ND tool, JDK, OpenVZ, CloudSim and GridSim respectively.

In[9] author has presented an idea of scheduling parallel applications for multiuser, non-homogenous and large scale distributed systems. Target here is to design a system which has ability to handle idle cycles in network along with ability to handle clustered system and parallel processors. The algorithm proposed here is a better combination of three already existing algorithms that can handle multiple processors capabilities, various types of architecture and the variations in the processes. Here the decisions for scheduling will be done with the aim to reduce the turnaround time. At last the virtual processors for each application are gang scheduled to increase the efficiency of the system.

In[10] author has presented an algorithm that runs with an aim to minimize response time and total time. These algorithms are used to develop an algorithm for query processing. These algorithms have result in more effectiveness cost and execution.

In[11] author has presented a survey on distributed system design for load sharing. The paper is proposed with the source and server initiative approaches. Performance evaluation is done between the ten selected algorithms. Here a QFactor has been defined that ranked the various selected algorithms on the basis of their efficiency and fairness. After that the evaluation is done by using various mathematical and simulation techniques, it has been concluded that design decision is a critical issue and existing algorithms are providing effective solutions.

Table 6 describes the complete review of the papers from year 1900-2000.

In[12] author has developed a successful algorithm that can handle the load of the system through cluster nodes. When they switch on the cluster handles the node dynamically and according turn off to save the energy. Algorithm works at three level firstly at cluster based, secondly at OS level and lastly by application negotiation.

In[13] author has discussed the issues related to the agent properties and load balancing. An algorithm of load balancing has been proposed that is based on communication in multiagent computing field. That has been implemented and correspondingly results have been evaluated. In algorithm, credit value is associated with every agent. And this credit value depends on the machine`s affinity, workload, mobility etc. Here the credit of every agent is investigated in the scenario of load imbalancement and then the lowest credit value agent will be drifted to the machine with the light load in system. Here experiment is held to analyze the comparison of improvement in both load balancing and its schemes oriented with workload.

In[14] author has developed an idea of GHS i.e. Grid Harvest Service that gives dynamic and self-adaptive scheduling which can works in large applications in non-homogenous environment.

Table 3. DFD Level-0

Here it has been described that the techniques used for load balancing are clustering, scheduling, process migration, mark span and gridding. And correspondingly the tools used on these techniques are ND tool, JDK, OpenVZ, CloudSim and GridSim respectively.

In[9] author has presented an idea of scheduling parallel applications for multiuser, non-homogenous and large scale distributed systems. Target here is to design a system which has ability to handle idle cycles in network along with ability to handle clustered system and parallel processors. The algorithm proposed here is a better combination of three already existing algorithms that can handle multiple processors capabilities, various types of architecture and the variations in the processes. Here the decisions for scheduling will be done with the aim to reduce the turnaround time. At last the virtual processors for each application are gang scheduled to increase the efficiency of the system.

In[10] author has presented an algorithm that runs with an aim to minimize response time and total time. These algorithms are used to develop an algorithm for query processing. These algorithms have result in more effectiveness cost and execution.

In[11] author has presented a survey on distributed system design for load sharing. The paper is proposed with the source and server initiative approaches. Performance evaluation is done between the ten selected algorithms. Here a QFactor has been defined that ranked the various selected algorithms on the basis of their efficiency and fairness. After that the evaluation is done by using various mathematical and simulation techniques, it has been concluded that design decision is a critical issue and existing algorithms are providing effective solutions.

Table 6 describes the complete review of the papers from year 1900-2000.

In[12] author has developed a successful algorithm that can handle the load of the system through cluster nodes. When they switch on the cluster handles the node dynamically and according turn off to save the energy. Algorithm works at three level firstly at cluster based, secondly at OS level and lastly by application negotiation.

In[13] author has discussed the issues related to the agent properties and load balancing. An algorithm of load balancing has been proposed that is based on communication in multiagent computing field. That has been implemented and correspondingly results have been evaluated. In algorithm, credit value is associated with every agent. And this credit value depends on the machine`s affinity, workload, mobility etc. Here the credit of every agent is investigated in the scenario of load imbalancement and then the lowest credit value agent will be drifted to the machine with the light load in system. Here experiment is held to analyze the comparison of improvement in both load balancing and its schemes oriented with workload.

In[14] author has developed an idea of GHS i.e. Grid Harvest Service that gives dynamic and self-adaptive scheduling which can works in large applications in non-homogenous environment.

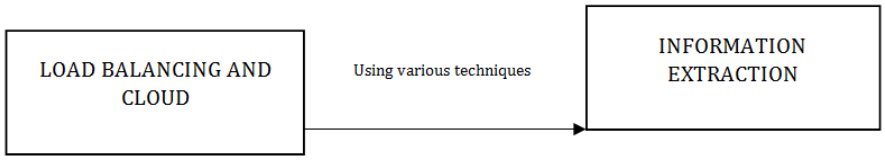

Table 4. DFD Level-1

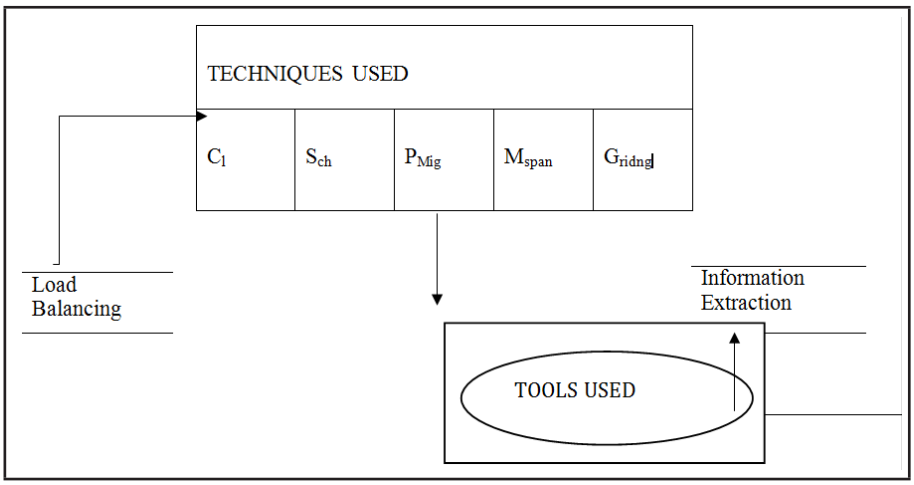

Table 5. DFD Level-2

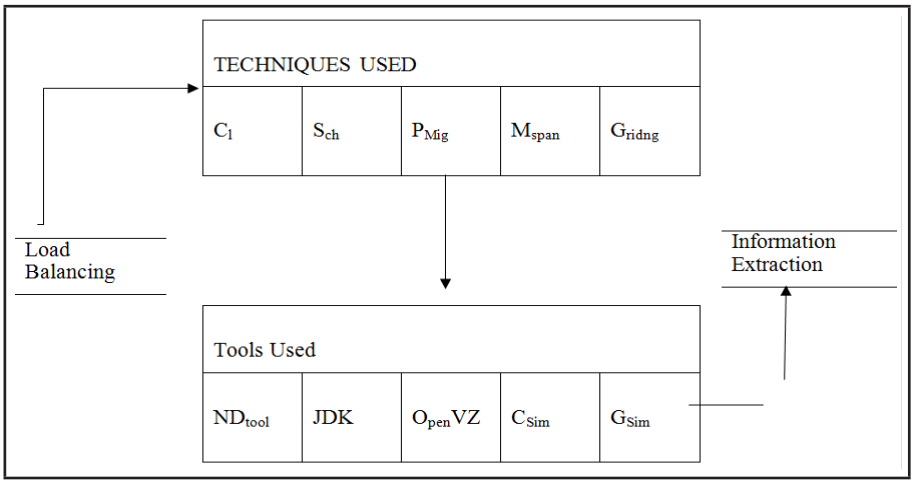

Table 6. Comparison between algorithms form 1900-2000

Author	Title	Year	Parameters				Algorithms Used			Pros			Cons		Tools / Tech used	
S.A Khaled et al. [9]	GSHDS	1900	Flx	E_{ff}	N_{ap}	N_{ap}	MTAT	$C_{omp}A$	$E_{xp}A$	E_{ff}	N_{ap}	N_{ap}	L_{d-}	S_{cb}	N_{ap}	N_{ap}
Alan R. Hevner et al.[10]	QPDDS	1979	QPDDS	N_{ap}	N_{ap}	N_{ap}	G_{based}	N_{ap}	N_{ap}	$C_{e,}$	E_{qpss}	E_{e}	stribtn I_{con-}	C_{fail-}	S_{e}	C_{l}
Yung-Terng Wang et al. [11]	LSDS	1985	LSDS	S_{vce}	P_{mnc}	F_{tol}	N_{ap}	N_{ap}	N_{ap}	N_{ap}	N_{ap}	N_{ap}	cern C_{S-} Class	ure S_{Sub-} sets	N_{ap}	N_{ap}

In[15] author has proposed an algorithm that carries a capability to organize sensors present in WSN into clusters. Algorithms have ability to attain cluster head hierarchy in which energy saving increase with its each level.

In[16] author has proposed a strategy for job grouping at runtime which in together have ability to perform simulation analysis. Quantity of jobs that can be processed in certain time can be determined by the processing of granularity size.

In[17] author has developed an algorithm which is layered approach for load balancing that too in grid computing. Algorithm supports heterogeneity along with scalability. It also made concern for adaptability. The algorithm is on top is a tree like structure model.

In[18] author has proposed a work for cloud workflow scheduling. It is made in concern to have pay as per use for execution and execution time. It was focused that algorithms must carry facilities for execution cost and time according to user`s input.

Table 7. Comparison between algorithms form 2001-2010

Author	Title	Year	Parameters				Algorithms Used			Pros			Cons		Tools / Tech used	
Eduardo Pinheiro et al. [12]	LBP_2CBS	2001	LBCBS	P_{wer}	E_{ngy}	N_{ap}	C_{confi}	$L_{dstribtn}$	N_{ap}	$E_{consrve,}$	$P_{con-srve}$	N_{ap}	N_{ap}	P_{et}	C_l	ND-tool
Ka-Po Chow et al.[13]	LBDMC	2002	$C_{pattern}$	W_{load}	A_{gnt}	H_{st}	C_{omet}	N_{ap}	N_{ap}	Th_{hold}	L_{red}	S_{cb}	N_{ap}	P_{loss}	JDK	N_{ap}
Ming Wu, et al. [14]	$SATSH_c$	2003	SATS	$A_{l-loctn}$	P_{mnce}	T_m	$M_{min}TA$	$M_{min}GA$	$S_{adap}TS$	$T_{relloat}$	P_{loss}	C_{time}	C_{cost}	$,M_{cost}$	GHS	N_{ap}
Seema B. et al. [15]	EHCAWN	2003	E_{ngy}	T_m	N_{ap}	N_{ap}	EE_{Algo}	N_{ap}	N_{ap}	L_{engy}	T_{cxity}	N_{ap}	C_{cost}	MSP	BUA	N_{ap}
Nithiapidary M et al. [16]	$D_JGB_{SA}F_{GT}$	2005	T_m	N_{ap}	N_{ap}	N_{ap}	$J_{gp}A$	$S_{chdule}A$	N_{ap}	Com_{time}	P_{otime}	N_{ap}	N_{ap}	J_{gp}	G_{Sim}	N_{ap}
Belabbas Yagoubi et al. [17]	$DL_{bal}SGC$	2006	DLBS	L_d	C_t	N_{ap}	I_{site}	$I_{cluster}$	I_{grid}	L_{bal}	C_{cost}	N_{ap}	G_s	O_{rge}	C_l	N_{ap}
Yun Yang et al. [18]	SSTCCW	2008	$C_{exec},$	T_m	N_{ap}	N_{ap}	WSA	N_{ap}	N_{ap}	C_{Min}	C_{exec}	N_{ap}	N_{ap}	R_{App}	SDW-C	N_{ap}
Yi Zhao et al. [19]	$AD_{LBA}M$-VM	2009	C_{nvrge}	T_m	N_{ap}	N_{ap}	CB	N_{ap}	N_{ap}	F_{Conv}	N_{ap}	N_{ap}	$M_{Slow,}$	$AA_{N-Soppt}$	OpenVZ	N_{ap}
Saeed Parsa et al.[20]	RASA	2009	M_{span}	N_{ap}	N_{ap}	N_{ap}	M_{in}-M_{in}	RASA	N_{ap}	R_{MS}	N_{ap}	N_{ap}	D_{task}	C_{exec}	G_{Sim}	N_{ap}
Meng Xu et al. [21]	MQCSS	2009	SS_{rate}	T_m	$M_{span,}$	C_t	$S_{chdule}A$	T_{Sort}	N_{ap}	D_{wrkflw}	C_{exec}	T_{exec}	A_{vlble}	R_{elable}	C_{Sim}	N_{ap}
Mrs.S.Selvarani et al. [22]	ICBATS	2010	P_{mnce}	T_m	N_{ap}	N_{ap}	T_{gpscd}	ABC	I_{scdule}	$Comp_{Cost}$	N_{ap}	N_{ap}	$SS_{chdule,}$	$D_{Factors}$	C_{Sim}	N_{ap}
Ke Liu et al. [23]	CTCSAS-DC	2010	C_t	T_m	N_{ap}	N_{ap}	CTC	N_{ap}	N_{ap}	$C_{exec},$	T_{exec}	N_{ap}	N_{ap}	C_{Envrt}	SDW-C	N_{ap}
Suraj Pandey et al .[24]	PSOHSW	2010	R_{utilze}	T_m	N_{ap}	N_{ap}	$S_{chdule}A$	PSO	N_{ap}	C_{Min}	W_{dis}	N_{ap}	N_{ap}	R_{App}	AEC_2	N_{ap}

Table 8. Comparison between algorithms from 2011-2015

Author	Title	Year	Parameter				Algorithms Used				Benefit (pro)				Cons			Tool used (tech)		
T.Kokilavani et al. [25]	LBMSMTS	2011	M_{span}	N_{ap}	N_{ap}	N_{ap}	LBMM	Min-Min	N_{ap}	N_{ap}	$R_{mrk-spn}$	N_{ap}	$R_{Utilize}$	T_{Hete}	M_{Hete}	A_{Cost}	C_{pp}	N_{ap}	N_{ap}
Cui Lin, et al. [26]	SSWEC	2011	T_{exec}	S_{cb}	N_{ap}	N_{ap}	SHEFT	N_{ap}	N_{ap}	N_{ap}	T_{exec}	N_{ap}	R_{Scale}	N_{ap}	N_{ap}	N_{ap}	C_{Sim}	N_{ap}	N_{ap}
Subasish Mohapatra et al.[27]	$CHLBVM_{Cc}$	2013	LBVM	T_m	L_d	R_{utilze}	ESCE	Thro	RR	FCFS	L_{bal}	R_{alloat}	$P_{I/Otime}$	S_{mb}	N_{ap}	N_{ap}	C_l	GSM	CA
P. Keerthika, et al. [28]	HSALB	2013	F_{tol}	M_{span}	L_d	N_{ap}	MCSA	N_{ap}	N_{ap}	N_{ap}	R_{MS}	F_{tol},	L_{Bal}	C_{ovrhed}	N_{ap}	N_{ap}	G_{Sim}	N_{ap}	N_{ap}
Priyanka Gautam, et al. [29]	ERRLB	2014	ERRLB	C_t,	T_m	N_{ap}	E_{rrlb}	N_{ap}	N_{ap}	N_{ap}	E_{time},	E_{cost}	N_{ap}	$M_{dis-cost}$,	$C_{resource}$	N_{ap}	SPP	N_b	N_{ap}
Santanu Dam et al. [30]	GAGEHL	2015	M_{span}	N_{ap}	N_{ap}	N_{ap}	GHLBA	N_{ap}	N_{ap}	N_{ap}	R_{MS}	R_{VM}	N_{ap}	F_{tol}	J_{Prior}	N_{ap}	C_{Ana}	N_{ap}	N_{ap}
Salawu Nathaniel at. [31]	$DPAT_d$ L_b PP	2015	RSS	C_l	N_{ap}	N_{ap}	DPAT-dA	N_{ap}	N_{ap}	N_{ap}	L_{Bal}	PP_h	N_{ap}	E_{op}	N_{ap}	N_{ap}	N_{ap}	N_{ap}	N_{ap}
M. Lavanya et al.[32]	EL_b S_a PC	2015	Tt	E_{ff}	M_s	N_{ap}	ELB	N_{ap}	N_{ap}	N_{ap}	R_t	E_{ff}	MS_{vm}	N_{ap}	N_{ap}	N_{ap}	N_{ap}	N_{ap}	N_{ap}
Moon Suk Yeon et al.[33]	MLL_b HWB_c	2015	S_{cb}	A_{vlble}	L_b	N_{ap}	$PPAL_b$	WAR_s	N_{ap}	N_{ap}	S_{cb}	A_{vlble}	L_b	N_{ap}	N_{ap}	N_{ap}	N_{ap}	N_{ap}	N_{ap}

In[19] author has proposed a Compare and Balance sampling based load balancing algorithm. In this paper virtual machine migration implementation has been worked on a simple and basic model that reduces the migration time of VM on the basis of storage. The results of above are successful.

In[20] author has proposed an algorithm RASA with features of scalability and distribution. It is based on the study of traditional scheduling algorithms. At top of all, the proposed algorithm carries merits and overcome the demerits of these studied algorithms. Like it execute non-large tasks first as compare to others and support concurrent execution of large and small task in spite of execution of just large tasks. Results show low mark span as compare to early.

In[21] author has presented a Multiple QoS Constrained Scheduling Strategy of Multi-Workflows. Here problems are handled by workflows that are related to requested services of different QoS requirements by multiple users at same time. This strategy works on the contribution of task sort algorithm and schedule algorithm. By this strategy the drawbacks of RASA dynamic workflow, issues related to execution cost and time had overcame.

In[22] author has proposed an algorithm that is concerned with the job grouping costs. Aim in this paper is to task group scheduling it is because of job grouping optimization increases. Here algorithm measures cost of resources and computation of performance.

In[23] has proposed an algorithm that is time cost algorithm for scheduling. This algorithm proves that cost of execution is decrease by around 15% whereas execution time is decrease by 20% within execution cost of user.

In[24] author has described a heuristic which is based on PSO i.e. Particle Swam Optimization that will schedule cloud resources to applications. Here experiment has been conducted on application of workflow by timely varying its communication and computational cost and it was conclude that PSO is three times saver in comparison to existing BRS i.e. it has best resource selection and is nice distributor of workload.

Table 7 is presented that describes the complete review of the papers from year 2001-2010.

In[25] author has studied that the basic min-min algorithm is the one that reduce the mark span as compare to others which somehow failed in load balancing scheduling but in this paper they proposed algorithm which on one hand decrease make span and in other hand increase the utilization of the resources. For this two phases are developed in first phase where basic min-min algorithm is executed in other hand non-utilized resources are made in use with maximum effectiveness.

In[26] author has proposed a SHEFT algorithm that schedule workflow. It's an algorithm that elastically schedule workflow at runtime.

In[27] author has discussed and presented the comparative study of various load balancing policies that are developed for effective and efficient service providing. But in our paper we have only used round robin algorithm for load balancing.

In[28] author has proposed an algorithm for load balancing, architecture based user satisfaction, fault tolerance, availability and heterogeneity of resources. Along with this using Grid Sim tool kit has reduced the make span. And it results in better hit rate of make span and utilization of resources.

In[29] author has observed that there a problematic issue of content storage which cause clustering and waiting queue for load balancing. Here the extended version of existing round robin algorithm has been proposed. This has resulted in reduced execution time and execution cost.

In[30] author has discussed a strategy for load balancing to balance overwhelmed node using cloud analyst simulation tool. Then the results attained from this proposed algorithm are made compared to the existing algorithms.

In[31] author proposed a dual-parameters adaptive timer decision algorithm for balancing the load and to simultaneously control the Ping-Pong handover in the network. And algorithm became successful to achieve 95% level for load balancing.

In[32] ELB algorithm has been proposed for scheduling of tasks to virtual machines that result in high throughput and low turnaround time. Here global queue has been implemented that results in increase of response time and efficiency.

In[33] Packet processing algorithm has been proposed that deals with bottom level dynamic weight based agents. Those later results increase in response time.

Table 8 is presented that describes the complete review of the papers from year 2011-2015.

Some of the algorithms are selected from all algorithms of evolution. Whose tabular comparison is shown in Table 2 and the algorithms are discussed below:

Algorithm: Complete_Comet

Begin
Initialize C_L and sleep T
For (C_L=1, C_L <L_{agent} , C_L++) // where C_L current location
{
For each Location (C_C=1, C_C< $C_{C(last)}$, C_C++)
{

Calculate the credits for each location

}
}

If
{
$C_{L(n)} = I_{REQUIRED}$
// Where $I_{REQUIRED}$ is information required to be calculated for each load.
Target meet
Then
Change T_H
// Where T_H is the threshold.
Exit(1)

Else
Migrate the agent with the smallest credit according to the selection and location policies
}
End

Comet algorithm is proposed in[13] that employs information collection and decision making policy. Here machines are synchronized and analyze load at periodic basis in contradiction with load threshold. This system has facility to report average. Here T is a set of all tasks and S is a set of available services. The major motive is to first submit the workflow of user with appropriate Quality of Services (QoS). After this the system will allocate the services for workflow and correspondingly will schedule the tasks according to QoS needs and environment load and variance at every phase of load balancing. After that selection decision is made accordingly. Corresponding algorithm is described above.

Input: Threshold load value TH (determined by profiling), information collection period T (in seconds), number of agent's n, hosts characteristics (address and other system information) and number of host' *sp*.

In[21] Multiple QoS Constrained Scheduling Strategy of Multi-Workflows has been introduced. Here problems are handled by the workflows that request the service of different QoS requirements by multiple users at same time. This strategy works on the contribution of task sort algorithm and schedule algorithm. By this strategy the drawbacks of RASA dynamic workflow, issues related to execution cost and time are overcame. Both the task sort algorithm[21] and schedule algorithm[21] are discussed above.

3. Conclusion

A complete evolution of computing from its preliminary stage to the current progressions has been discussed in this paper. Scheduling has been considered as an important issue for managing execution of applications in cloud. Therefore, progress that how load balancing in computing environment was done from early years to current year has been studied. Along with the assessments between the algorithms that have actually overcome the shortcomings of the existing algorithms, the parameters those were considered by various researchers which will help the new researchers to analyze and select the parameters those need much more concentration to meet the required functions are discussed in this paper.

Algorithm : Task Sort Algorithm

```
Begin

Initialize T, S

G_RT()                    // Where G_RT is a Get Ready Task
{
    while ( t is Ready )
    t=R_T                 // Where R_T is a ready task
}

ST(R_T, q)                // Where ST is Sort Task
{
    while (t. R_T)
    IT(t, q)
}

IT (t, q)                 // Where IT is Insert Task
 {
    Insert t into q according to strategy
 }
```

Algorithm: Schedule Algorithm

```
Begin
Initialize q, t

S_Ch (q,S)                // Where S_Ch is a schedule
{
   while (q!=0)
     {
       q= t               // first task in queue
       s=gets(t, S)
       schedule t on s
       q= q – t
       S= S – r
     }
}

Get_S (t,S)               // Where Get_S is a Get Service
  {
    Select s.S
    (E_time and C_t)!<= (T_time and T_cost)   // Where E_time
is Execution Time,
                          C_t is cost, T_time is Total Time and
T_cost is Total cost

return s
  }
End
```

4. References

1. Inderpal S. Review on parallel and distributed computing. Scholars Journal of Engineering and Technology, 2013. 1(4):218–25.
2. Satyanarayanan M. Pervasive computing: Vision and challenges. IEEE Personal Communications. 2001 Aug; 8(4):10–7.
3. Rajkumar B, Hai J, Toni C. Cluster computing. Future Generation Computer Systems. 2002 Jan; 18(3):5–8.
4. Suri PK, Sumit M. A comparative study of various computing processing environments: A review. International Journal of Computer Science and Informational Technology. 2012; 3(5):5215–8.
5. Ross JW, Westerman G. Preparing for utility computing: The role of IT architecture and relationship management. IEEE IBM Systems Journal. 2010 Jan; 43(1):5–19.
6. Joseph J, Ernest M, Fellenstein C. Evolution of grid computing architecture and grid adoption models. IBM System Journal. 2004; 43(4):624–45.
7. Shyamala K, Sunitha Rani T. An analysis on efficient resource allocation mechanisms in cloud computing. Indian Journal of Science and Technology. 2015 May; 8(9):814–21.
8. Balaji N, Umamakeshwari A. load balancing in virtualized environment - A survey. Indian Journal of Science and Technology. 2015 May; 8(S9):230–4.
9. Al-Saqabi K, Otto SW, Walpole J. Gang scheduling in heterogenous distributed systems. CSE_Tech; 1900 Jan. p. 1–25.
10. Henver AR, Yao SB. Query processing in distributed database system. IEEE Transactions on Software Engineering. 1979 May; 5(3):177–87.
11. Wang YT, Morris RJT. Load sharing in distributed systems. IEEE Transactions on Computers. 1985; 34(3):204–17.
12. Pinherio E, Bianchini R, Carrera EV, Heath T. Load balancing and unbalancing for power and performance in cluster-based systems. Technical Report DCS–TR–440, Workshop on Compilers and Operating Systems for Low Power. 2011; 180:1–11.
13. Chow K, Kwok YK. On load balancing for distributed multiagent computing. IEEE Tranactions on Parallel Distributed Systems. 2002 Aug; 13(8):787–801.
14. Wu M, Sun XH. A general self-adaptive task scheduling system for non-dedicated heterogeneous computing. Proceeding 2003. IEEE Conference on Cluster Computing; 2003 Dec 1-4. p. 354–61.
15. Bandyopadhyay S, Coyle EJ. An energy efficient hierarchical clustering algorithm for Wireless Sensor Networks. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. INFOCOM 2003; 2003; 3: p. 1713–23.
16. Muthuvelu N, Liu J, Soe NL. Venugopal S, Sulistio A, Buyya R. A dynamic job grouping-based scheduling for deploying applications with fine-grained tasks on global grids. Australasian Workshop on Grid Computing and e-Research, Newcastle. 2005; 44:41–8.

17. Yagoubi B, Slimani Y. Dynamic load balancing strategy for grid computing. World Academy of Science, Engineering and Technology. 2008; 2(7):2424–9.
18. Yang Y, Liu K, Chen J, Liu X. An algorithm in SwinDeW-C for scheduling transaction-intensive cost-constrained cloud workflows. IEEE 4th Conference on e-Science`08; Indianapolis, IN. 2008. p. 374–5.
19. Zhao Y, Huang W. Adaptive distributed load balancing algorithm based on live migration of virtual machines in cloud. IEEE 5th International Joint Conference on INC, IMS and IDC, NCM`09; Seoul. 2009 Aug 25-27. p. 170–5.
20. Parsa S, Entezari-Maleki R. RASA: A new task scheduling algorithm in grid environment. World Applied Sciences Journal. 2009; 7:152–60.
21. Xu M, Cui L, Wang H, Bi Y. A multiple QoS constrained scheduling strategy of multiple workflows for cloud computing. IEEE International Sympoium on Parallel and Distributed Processing with Applications; Chengdu. 2009 Aug 10-12. p. 629–34.
22. Selvarani S, Sadhasivam GS. Improved cost-based algorithm for task scheduling in cloud computing. IEEE International Conference on Computational Intelligence and Computing Research (ICCIC); Coimbatore. 2010 Dec 28-29. p. 1–5.
23. Liu K, Jin H, Chen J, Liu X, Yuvan D, Yang Y. A compromised-time-cost scheduling algorithm in SwinDeW-C for instance-intensive cost-constrained workflows on cloud computing platform. International Journal of High Performance Computing Applications. 2010 Nov; 24(4):445–56.
24. Pandey S, Wu L, Guru SM, Buyya R. A particle swarm optimization-based heuristic for scheduling workflow applications in cloud computing environments. IEEE International Conference on Advanced Information Network and Applications (AINA); Perth WA. 2010 Apr 20-23. p. 400–7.
25. Kokilavani T, Amalarethinam DI. Load balanced min-min algorithm for static meta-task schedulimg in grid computing. International Journal of Computer Applications. 2011 Apr; 20(2):43–9.
26. Lin C, Lu S. Scheduling scientific workflows elastically for cloud computing. 2011 IEEE International Conference on Cloud Computing (CLOUD); Washington DC. 2011 p. 746–7.
27. Mohapatara S, Smruti Rekha K, Mohanty S. A comparison of four popular heuristics for load balancing of virtual machines in cloud computing. International Journal of Computer Applications. 2013; 68(6):33–8.
28. Keerthika P, Kasthuri N. A hybrid scheduling algorithm with load balancing for computational grid. International Journal of Advanced Science Technology. 2013; 58:13–28.
29. Gautam P, Bansal R. Extended round robin load balancing in cloud computing. International Journal of Engineering Computer Science. 2014 Aug; 3(8):7926–31.
30. Dam S, Mandal G, Dasgupta K, Dutta P. Genetic algorithm and gravitational emulation based hybrid load balancing strategy in cloud computing. IEEE 3rd International Conference on Computer Communication Control and Information Technology; Hooghly. 2015 Feb 7-8. p. 1–7.
31. Nathaniel S, Ariffin SHS, Farzamnia A. A dual-parameter adaptive timer decision algorithm for load balancing and ping-pong handover control in LTE Networks. Indian Journal of Science and Technology. 2015 Dec; 8(33):1–12.
32. Lavanya M, Ravi A, Aditya A, Samyuktha R, Vaithiyanathan V, Saravanan S. An enhanced load balancing scheduling approach on private clouds. Indian Journal of Science and Technology. 2015 Dec; 8(35):1–4.
33. Yeon MS, Jeong BS. Multi-level load balancing methods for hierarchical web server clusters. Indian Journal of Science and Technology. 2015 Sep; 8(21):1–5.

Appendix

P_{Mig}- Process Migration, G_{ridng}- Gridding, GSHDS-Gang scheduling in heterogeneous distributed systems, Flx-Flexibility, E_{ff}-efficiency, MTAT- Minimum Turn Around Time, C_{omp}A- Compression Algorithm, E_{xp}A-Expansion Algorithm, $L_{dstribtn}$-Load Distribution, S_{cb}-Scalability, QPDDS-Query Processing in Distributed DatabaseSystem, G_{based}-Grammer Based Algorithm, C_{e}-Cost Effective, E_{qpss}-Effective query processing subsystem, E_{e}-Effective execution, $I_{concern}$-Integral part concern, $C_{failure}$-Component failure, S_{e}-Scheduling, C_{l}-Clustering. LSDS-Load Sharing in Distributed Systems, S_{vce}-Service, P_{mnce}-Performance, F_{tol}-Fault Tolerance, N_{a}- Not applicable, C_{SClass}-Simple class considerations, $S_{Subsets}$-Small subsets. $LBP_{2}CBS$- Load Balancing and Unbalancing for Power and Performance in Cluster-Based Systems, LBCBS-Load Balancing in Cloud Based System, P_{wer}-Power, E_{ngy}-Energy, C_{confi}- cluster configuration algorithm, $L_{dstribtn}$-load distribution algorithm, $E_{consrve}$-Energy conservation, $P_{consrve}$-Power conservation, P_{et}-Execution time prediction, ND_{tool}- ND Tool. LBDMC-Load Balancing for Distributed Multiagent Computing, $C_{pattern}$-Communication Pattern ,W_{load}-Workload, A_{gnt}-Agents, H_{st}-Hosts, C_{omet}-Comet, Th_{hold}-Threshold, L_{red}- Load Reduction, S_{cb}-Scalability, P_{loss}-Performance Loss. $SATSH_{C}$-Self-adaptive Task Scheduling System for Non-dedicated Heterogeneous Computing SATS- Self-adaptive Task Scheduling System, A_{lloctn}-Allocation, P_{mnce}-Performance, T_{m}-Time, M_{min}TA- Mean-time task allocation algorithm, M_{min}GA-Min-min task group allocation algorithm, S_{adap}TS-Self-adaptive task scheduling algorithm, $T_{relloat}$- Task reallocation, P_{loss}-Performance loss, C_{time}- Completion time, C_{cost}-Communication cost, M_{cost}-Migration cost, GHS- Grid Harvest Services. EHCAWN- Energy Efficient Hierarchical Clustering Algorithm for wireless Sensor Network, E_{ngy}-Energy, T_{m}-Time, EE_{Algo}-Energy Efficient Algorithm, L_{engy}- Less energy consumption, T_{cxity} -Time complexity , C_{cost}- Communication cost, MSP- Medium

access protocol, BUA- Bottom- up approach. $D_JGB_{SA}F_{GT}$- Dynamic Job Grouping-Based Scheduling for Deploying Applications with Fine-Grained Tasks on Global Grid, T_m-Time, $J_{gp}A$- Job Grouping Algorithm, $S_{chdule}A$- Scheduling Algorithm, Com_{time}-Communication time, P_{otime}- Processing overhead Time, J_{gp}-Job Grouping, G_{Sim}- Grid Sim. $DL_{bal}SGC$- Dynamic Load Balancing Strategy for Grid Computing DLBS- Dynamic Load Balancing Strategy, L_d-Load, C_t-Cost, I_{site}- intra-site algorithm,$I_{cluster}$- intra-cluster algorithm, I_{grid}-intra-grid algorithm, L_{bal}- Load balancing, C_{cost}-Communication cost, G_s- Grid simulators, O_{rge}-Real grid environment operations, C_l- Clustering. $AD_{LBA}MVM$- Adaptive Distributed Load Balancing Algorithm based on Live Migration of Virtual Machines in Cloud, T_m-Time, C_{nvrge}-Convergence, CB- Compare and balance, F_{Conv}- Fast Convergence, M_{Slow}- Slow Migration for virtual machine, AA_{NSoppt} –No Support for affinity and anti-affinity. RASA- RASA:A New Task Scheduling Algorithm in Grid Environment, M_{span}- Markspan, M_{in}-M_{in} – Min-min Algorithm, RASA- Resource-Aware-Scheduling algorithm, R_{MS}- reduce makespan, D_{task}- Tasks Deadlines, C_{exec}-Execution Cost ,G_{Sim}-Grid Sim. N_{ap}-Not applicable. MQCSS- A Multiple QoS Constrained Scheduling Strategy of Multiple Workflows for Cloud Computing, SS_{rate}-Schedule Success rate, T_m-Time, C_t-Cost, M_{span}-, Markspan ,T_{Sort}- Task Sort Algorithm, $S_{chdule}A$- Schedule Algorithm, D_{wrkflw}- Dynamic workflow, C_{exec}-Execution Cost, T_{exec}- Execution time, A_{vlble}- Availability, R_{elable}- Reliability, C_{Sim}-Cloud Sim. ICBATS- Improved Cost-Based Algorithm For Task Scheduling in Cloud Computing, T_m-Time, P_{mnce}-Performance, ABC- ABC Algorithm, T_{gpscd}- Task grouping and scheduling algorithm, I_{scdule} - Improved Scheduling algorithm, $Comp_{Cost}$- Compilation Cost, SS_{chdule}-Simultaneous Scheduling , $D_{Factors}$-Dynamic Factors, C_{Sim}-Cloud Sim. CTCSASDC- A Compromised-Time-Cost Scheduling Algorithm in SwinDeW-C for Instance-Intensive Cost-Constrained Workflows on Cloud Computing Platform, C_t-Cost, T_m-Time, CTC-Compromised-Time-Cost Scheduling Algorithm, C_{exec}-Execution Cost, T_{exec}- Execution time, C_{Envrt}- Multiple cloud environment, SDW-C- Swinburne Decentralised Workflow for Cloud. PSOHSW- A Particle, Swarm Optimization-based Heuristic for Scheduling Workflow Applications in Cloud Computing Environment, $R_{utilize}$-Resource Utilization, T_m-Time, $S_{chdule}A$- Schedule Algorithm , PSO- Particle Swarm Optimisation Algorithm, C_{Min}-Cost Minimum, W_{dis}- Workload Distribution, R_{App}- Real Applications, AEC_2 -Amazon EC2. LBMSMTS- Load balanced Min-Min algorithm for static meta-task scheduling in grid computing, M_{span}-, Markspan , LBMM- Load balanced Min-Min algorithm, M_{in}-M_{in}- Min-Min algorithm, R_{MS}- reduce makespan,

$R_{Utilize}$-Resource Utilization, M_{Hete}- machine hetrogeneity, T_{Hete}- task heterogeneity, A_{Cost}- Cost Analysis, C_{pp}-C++. SSWEC- Scheduling ScientificWorkflows Elastically for Cloud Computing, T_{exec}- Execution time, S_{cb}-Scheduling, SHEFT- Scalable-Heterogeneous-Earliest-Finish-Time algorithm T_{exec}- Execution time, R_{Scale}-Resource Scale, C_{Sim}- Cloud Sim. $CHLBVM_{Cc}$ - Comparison of heuristics for Load Balancing of Virtual Machines in Cloud Computing, LBVM- Load Balancing of Virtual Machines in Cloud Computing, $R_{utilize}$-Resource Utilization, T_m-Time, L_d-Load, RR-Round Robin Algorithm, Thro- Throttled algorithm, ESCE- Equally Spread Current Execution Load, FCFS- First Come First Serve algorithm, L_{bal}-Load Balancing, R_{alloat}-Resource allocation, $P_{I/Otime}$- Propagating Input/output time, S_{mb}-Same mb for each cloutlet, GSM- Load balancing Generic Model, CA- Cloud Analyst. HSALB- A Hybrid Scheduling Algorithm with Load Balancing for Computational Grid, F_{tol}-Fault Tolerance, M_{span}-, Markspan, L_d-Load MCSA- Multi Criteria scheduling Algorithm, R_{MS}- reduce makespan ,L_{Bal} –Load Balancing, C_{ovrhed}-Communication overhead, G_{Sim}-Grid Sim. ERRLB- Extended Round Robin Load Balancing in Cloud Computing, C_t-Cost, T_m-Time E_{rrlb}- Extended Round Robin Load Balancing in Cloud Computing, E_{time}-Execution time, E_{cost}-Execution cost, $M_{discost}$-Minimum distance cost, $C_{resource}$- Resource cost, SPP- Service Proximity Policy, N_b-Netbeans. GAGEHL- Genetic Algorithm and Gravitational Emulation Based Hybrid Load Balancing Strategy In Cloud Computing, M_{span}- Markspan, GHLBA- Gravitational Emulation Based Hybrid Load Balancing Algorithm, R_{MS}- reduce make span, R_{VM}-Response time of virtual machines, F_{tol}- Fault Tolerance, J_{Prior}-Job Priority, C_{Ana}- Cloud Analyst. D_{Evnt}-Distributed Environment, G_{Envt}- Grid environment, C_{Envt}- Cloud Environment, $DPAT_d$ L_b PP- Dual-Parameter Adaptive Timer DecisionAlgorithm for Load Balancing and Ping-Pong Handover Control in LTE Networks, DPATdA- dual-parameter adaptivetimer based decision algorithm, C_l- cell load, EL_b S_a PC- An Enhanced Load Balancing Scheduling Approachon Private Clouds, ELB- Enhanced Load Balancing algorithm, M_s- memory space, MLL_b HWB_c- Multi-Level Load Balancing Methods for HierarchicalWeb Server Clusters, $PPAL_b$- Packet processing algorithm for load balancer, WAR_s- Weight algorithm for real servers.