Performance Analysis of Heterogeneous Earliest Time First Algorithm: A Cloud Computing Prospective

[1]Er. Pallavi Narang, [2]Dr. I.S. Hudiara, [3]Dr. S.S. Khurmi
[1]Research Scholar, Desh Bhagat University, Mandi Gobindgarh, Fatehgarh Sahib, Punjab, India
[2]Director Research, Chitkara University, India
[3]Professor, Dept. of Computer Sciences, Desh Bhagat University, Mandi Gobindgarh, Punjab, India

Abstract
Cloud Computing has become means to deploy large scale systems involving tasks prioritizing and processor selection phases in a robust and cost effective manner. The maintenance of large scale systems involves monitoring which allows for the detection of faults, errors and anomalies and the enacting of optimization and corrective measures. Monitoring large scale systems is significant challenge requiring accessing of large volumes of data and near real time analysis. In this paper we propose various software metrics that will allow evaluation of Heterogeneous earliest time first algorithm on the basis of computation and communication cost. We describe in detail the mechanisms which enable HEFT to function effectively and explore the performance of HEFT through a detailed evaluation.

Keywords
Cloud Computing, Software Metrics, Heterogeneous Earliest Time First(HEFT), Cloud Models.

I. Introduction
Earlier the term "cloud" was used for computing networking diagrams for hiding the complexity of the infrastructure concerned. A cloud is a combination of hardware and software that runs in those data centers that helps to enable the computing model. Now, the cloud services providers gives more of the space and particular parallel computing resources at low cost.

The main challenge in cloud services is the management of data in bulkiness. The cloud computing has become prevalent and stores the large amount of data in the cloud which is being shared by the users with the privileges later.

Cloud computing is a service distributor over the internet for computing, data access and cloud storage by create scalability, elasticity and less cost. Cloud computing [1] has become the growing technology for computing these days. A cloud computing is normally includes models like Software as a Service, Infrastructure-as-a-service and Platform-as-a-service.

Cloud computing mainly focuses on allocating data and computations over a scalable information centers of network. It is basically a type of computing that rely on allocation of the computing resources rather than local servers and applications [2]. This technology is for the cost estimation, application hosting, and substance storage. Cloud computing system ought to have the following properties:

- To supply the service to end users quickly via interface on Internet.
- To assign and organize physical resources accurately and rapidly.
- To acquire the characteristics of scalability.
- To conform the rule of on demand function and distribution.
- To deal with huge quantity of requests and examine needs gracefully.

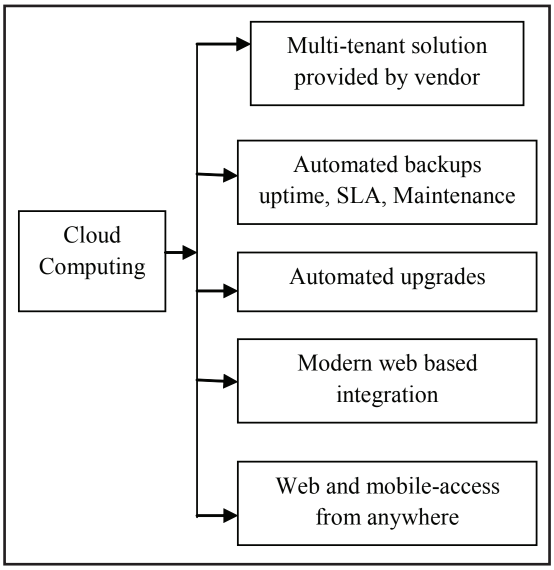

Fig. 1: Cloud Computing

The pattern of the model carry out the relation ↔1↔n .

The above fig. shows the cloud computing system arrangement.

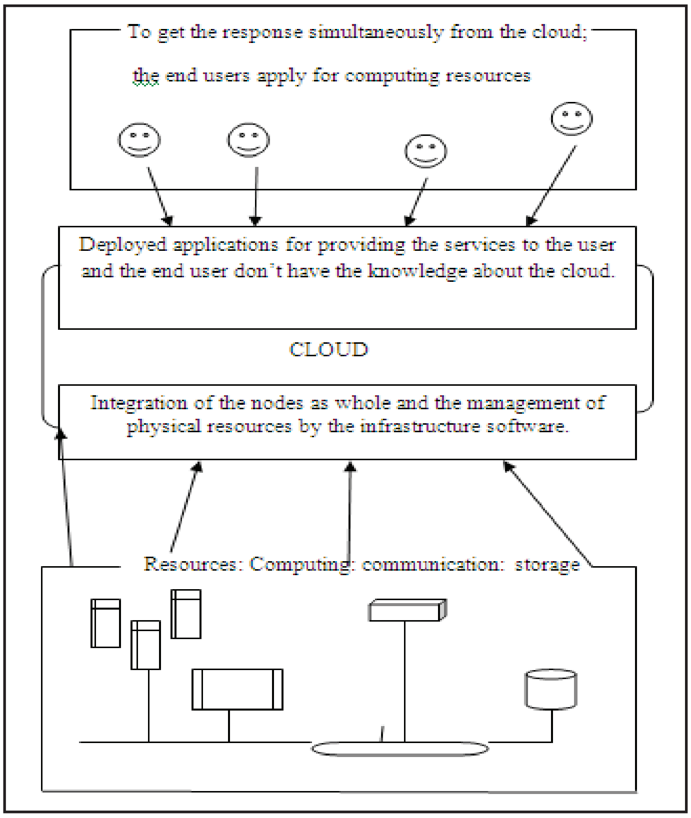

Fig. 2: Cloud Computing Pattern

ISSN : 0976-8491 (Online) | ISSN : 2229-4333 (Print)

A. Cloud Models

1. Deployment Model
Cloud deployment models correspond to a specific type of cloudy environment, primarily divided by size and access. Deployment model are three types:

(i). Public Cloud Computing
It is mainly depends on third person to suggest services by paying them on monthly basis according to the procedure. Public Cloud environment is made accessible to all unrestricted users who can subscribe the needed services [3]. The security issues will be decided by the service supplier and so it is very vital to choose the provider.

(ii). Private Cloud Computing
The organization itself has control over the services. Usually organizations leave for private cloud when the need of sensible data occurs. Scaling can be done very efficiently by adding hardware and thus the environment can be expanded.

(iii). Hybrid Cloud Computing
It is the combination of both public and private cloud computing. A less sensible data will be stored in public and others in Private Cloud.

Table 1: Comparison of Public and Private Cloud Computing

Public Cloud Computing	Private Cloud Computing
Can be used by more customers	Only a single customer
Suitable for no sensible information	Suitable for sensible information
Less security	Highly Secure
Utilizes shared infrastructure.	Utilize shared infrastructure.

2. Service Models
A cloud is a computing process in which services are dispersed above network by computing processes. Service models consist of three main categories:

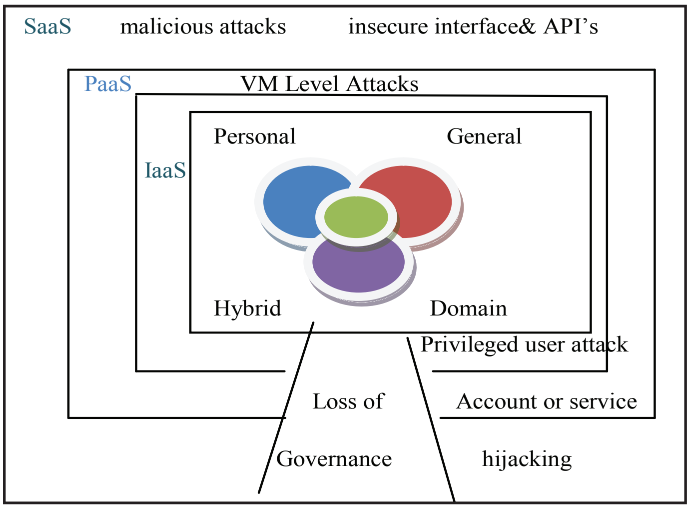

Fig. 3: Cloud Service Models

(i). SaaS (Software as a Service)
- The web access is given to commercial software.
- From a middle location, the software is managed.
- One –to-many is the way for delivering the software.
- The users don't need to manage software improvements and patches.
- Among number of software's, Application Programming Interfaces (APIs) allows the integration.

(ii). PaaS (Platform as a Service)
- To allow the services to expand, experiment, organize, host and protect the application in the same integrated improved atmosphere and the equivalent services desired to accomplish the application development procedure.
- The web build user interface formation tools assists to make, adapt, test and organize dissimilar UI framework.

	SaaS	PaaS	IaaS
	Consume	Build	Host
Consumer	End User	Application user	Application Owner
Service Provided Type	Completed Application	Runtime scenario Cloud Storage Integration, etc	Cloud Storage Virtual Server
Service Level Coverage	Application Uptime Application Performance	Environment Availability Environment performance No application Coverage	Virtual Server Availability Time to provision No platform or application coverage
Service provided examples	CRM E-mails Collaboratives ERP	Application Development Decision Support Web Streaming	Caching Security Legacy System Management

- Multi-tenant plan that has numerous simultaneous users use the similar growth application.
- Constructed in scalability of deployed software counting load balancing and failover.
- Addition with the web services and databases of frequent standards.
- Sustain for growth team collaboration – some PaaS solutions comprises of project planning and communication tools.
- Tools to handle billing and subscription management.

(iii). IaaS (Infrastructure as a Service)
- The resources are dispersed as a service.
- It permits for effectual scaling.
- It has a patchy cost, usefulness pricing model.
- Usually it has multiple users on a solitary piece of hardware.

II. Heterogenous Earliset Time First (HEFT).
It is one of the best and acceptable list based heuristics. Twenty scheduling heuristics has been compared by Canon and concluded HEFT gives the best schedule for the random graphs. The work was being compared by means of robustness and schedule length. HEFT algorithm has a complexity of $O(v^2 p)$. v shows the number

of tasks and p shows the number of processors. Priority rank is being assigned to every task and the insertion based framework is used for assigning the tasks to a specific processor for minimizing the overall execution time.

HEFT is the application for the bounded number of heterogeneous processors consisting Task prioritizing phase and the Processor Selection phase.

The Task Prioritizing phase needs the priority for each task for the upward rank value i.e. $rank_u$. The upward rank value depends on the mean computation and on the mean communication costs. With the decreasing order for $rank_u$, the task list is sorted. This phase is basically used for computing the priority of every task.

The upward rank of the task m_i is recursively defined as:

$$rank_u(m_j) = \overline{v_i} + max_{m_j \in succ(m_j)}(\overline{b_{i,j}} + rank_u(m_j)) \quad (1)$$

$succ(m_j)$ are the immediate successors of the task m_i.

$$rank_u(C) = \overline{v_{exit}} \quad (2)$$

Above equation is for the exit task $rank_u\ (m_j)$ with the upward rank value.

The Process Selection phase is used for selecting the tasks by means of their priorities by scheduling every task by the best processor by minimizing the finishing time.
For the task prioritization phase and the process selection phase, the algorithm of HEFT is defined as:

1. The computation cost and the communication cost of the edges to be set with the mean values.
2. Calculate the $rank_u$ for the tasks started with the exit task by the traversing graph.
3. Tasks should be arranged in non-increasing order for $rank_u$ values.
4. While the presence of unscheduled tasks in the list do
5. To select the first task, m_i, from the scheduling list.
6. For the processors in queue, the processor- set is ($q_k \in P$)do
7. Calculate EFT (m_i, q_k)value by insertion based scheduling policy.
8. To assign the task m_i to q_k processors.
9. End while

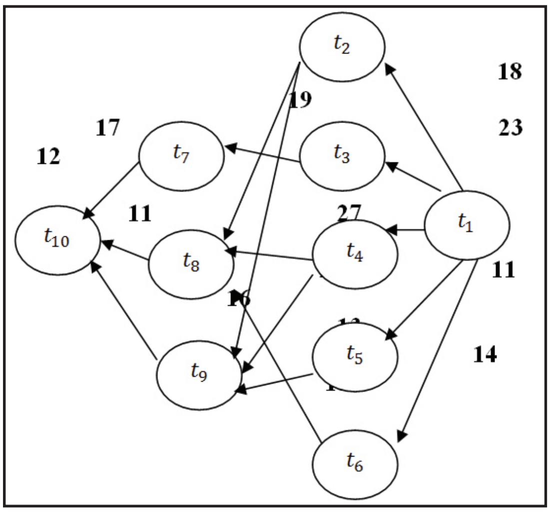

Fig. 4: Sample Task Graph

Table 2: Computation Cost Matrix

Task	P_1	P_2	P_1
t_1	14	16	9
t_2	13	19	18
t_3	11	13	19
t_4	13	8	17
t_5	12	13	10
t_6	13	16	9
t_7	7	15	11
t_8	5	11	15
t_9	18	12	20
t_{10}	21	7	16

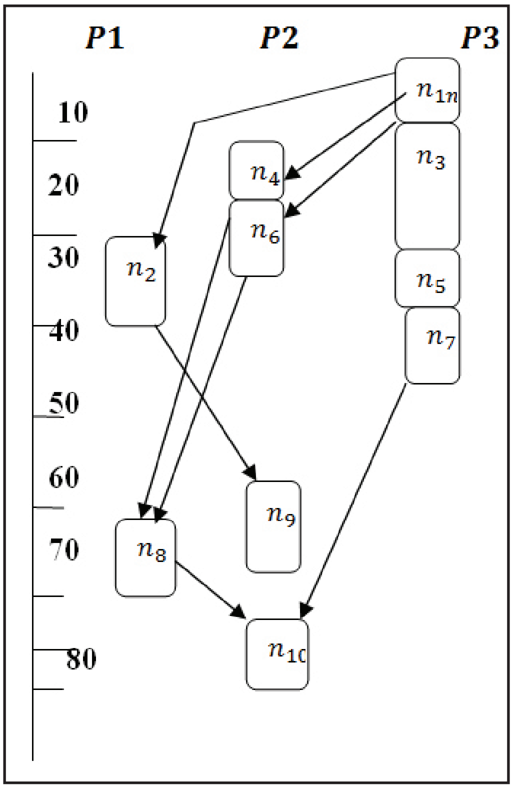

Fig. 5: Task Graph Scheduling With HEFT Algorithm

III. Software Metrices

A. Throughput

The network throughput is the total number of bytes successfully received, divided by the total time over which the transmission is being attempted. In other words, it is the size of all packet headers and trailers in the number of bytes successfully received.

Throughput = packet_sent/total_data With total_data = packet_sent+packet_lost

B. Entropy

It calculates a network performance with given targets and outputs, with optional performance weights and other parameters.

The formula for entropy in the case of a two-valued variable is as follows:

entropy = -(p * log(p) + (1-p) * log(1-p))

C. BER

A bit error rate is defined as the rate at which errors occur in a transmission system. This can be directly translated into the number of errors that occur in a string of a stated number of bits. The definition of bit error rate can be translated into a simple formula:

BER = No. of errors/ total bits sent

D. Elapse Time

Elapsed time is simply the amount of time that passes from the beginning of an event to its end. The first thing always needs to do when solving for elapsed time is to figure out your start time and end time.

Toc reads the elapsed time from the stopwatch timer started by the tic function. The function reads the internal time at the execution of the toc command, and displays the elapsed time since the most recent call to the tic function that had no output, in seconds.

E. Jitter

Jitter is the variation in the latency on a packet flow between two systems, when some packets take longer to travel from one system to the other. Jitter results from network congestion, timing drift and route changes. Jitter is especially problematic in real-time communications like IP telephony and video conferencing.

$J(i) = J(i–1) + (|D(i–1, i)| – J(i–1)) / 16$

Where the value D(i–1, i) is the difference of relative transit times for the two packets.

F. DIT

Depth of inheritance, also called depth of inheritance tree (DIT), is defined as "the maximum length from the node to the root of the tree". Depth of inheritance is predicated on three fundamental assumptions [CK]:

1. The deeper a class in the hierarchy, the greater the number of methods it will probably inherit which makes it harder to predict its behavior.
2. Deeper trees involve greater design complexity since more classes and methods are involved.
3. Deeper classes in the tree have a greater potential for reusing inherited methods.

Assumptions 1 and 2 tell us that having a higher number for depth is bad. If that is where it ended we would be in good shape; however assumption 3 indicates that a higher number for depth is good for potential code reuse.

Inheritance tree depth = max (inheritance tree path length)

G. CCR

High CCR indicates high variance of the computation costs of a task, with respect to the processors in the system, and vice versa. As the value of CCR increases, interprocessor communication overhead dominates computation and hence, the performance of all three scheduling algorithms tends to degrade.

C = S + R * u

S= cost of communication during startup time
R= average communication cost over all processors
U = units of data from tasks

H. SLR

It is the normalized schedule length to the lower bound of the schedule length. It is calculated using the following formula.

$SLR = makespan / \sum C_{min}$

I. Efficiency

As value of CCR increases over thereby lowering the efficiency of the heterogeneous architecture.
Efficiency = 2*(Vertex Count – Processor Count)/ Overall Communication Cost

J. Speedup

The speedup value is given by dividing the sequential executing time by parallel execution time. The sequential time is computed by assigning all tasks to single processor that minimize the Computation cost.

$Speedup = \min \sum C_{omputation\ cost} / makespan$

K. LOC

Lines of Code or LOC (also known as Source Lines of Code - SLOC) is a quantitative measurement in computer programming for files that contains code from a computer programming language, in text form. The number of lines indicates the size of a given file and gives some indication of the work involved.

LOC= ∑ Total no. of source lines

References

[1] Cong Wang et al,"Ensuring Data Storage Security in Cloud Computing," IEEE, 2009.
[2] Balachandra Reddy Kandukuri, et al,"Cloud security issues," In Services, Computing, 2009. IEEE International Conference on, page 517520, 2009.
[3] Guan, Qiang, et al, "Cda: A cloud dependability analysis framework for characterizing system dependability in cloud computing infrastructures," Dependable Computing (PRDC), 2012 IEEE 18th Pacific Rim International Symposium on. IEEE, 2012.
[4] Amanatullah, et al., "Toward cloud computing reference architecture: Cloud service management perspective," ICT for Smart Society (ICISS), 2013 International Conference on. IEEE, 2013.
[5] Luo, Shengmei, et al, "Virtualization security for cloud computing service," Cloud and Service Computing (CSC), 2011 International Conference on. IEEE, 2011.
[6] Sabahi, Farzad, "Virtualization-level security in cloud computing, "Communication Software and Networks (ICCSN), 2011 IEEE 3rd International Conference on. IEEE, 2011.
[7] Duan, Qiang, Yuhong Yan, and Athanasios V. Vasilakos. "A survey on service-oriented network virtualization toward convergence of networking and cloud computing." Network

and Service Management, IEEE Transactions on 9.4 (2012): 373-392.
[8] Tan, Yuesheng, Dengliang Luo, Jingyu Wang. "Cc-vit: Virtualization intrusion tolerance based on cloud computing." Information Engineering and Computer Science (ICIECS), 2010 2nd International Conference on. IEEE, 2010.
[9] Zhong, Liang, et al.,"A Virtualization-based SaaS Enabling Architecture for Cloud Computing." Autonomic and Autonomous Systems (ICAS), 2010 Sixth International Conference on. IEEE, 2010.
[10] Dong, Hanfei, et al.,"Formal discussion on relationship between virtualization and cloud computing." Parallel and Distributed Computing, Applications and Technologies (PDCAT), 2010 International Conference on. IEEE, 2010.
[11] Zhu, Qixuan, Xi Zhang.,"Game-theory based power and spectrum virtualization for maximizing spectrum efficiency over mobile cloud-computing wireless networks." Information Sciences and Systems (CISS), 2015 49th Annual Conference on. IEEE, 2015.

Er. Pallavi Narang recieved her Bachelor's degree in Computer Science and Engineering from Kurukshetra University, Haryana in 2010 and received the Master's Degree in Computer Science and Engineering from Punjab Technical University, Jallandhar in 2012. She is currently pursuing Ph.D. in Computer Science and Engineering, at Desh Bhagat University, Mandi Gobindgarh, Punjab, India. She is Presently working as Assistant Professor in Chandigarh Group of Colleges, Landran, Mohali since June 2015. She has worked as Assistant Professor in Computer Science & Engineering Department at Seth Jai Prakash Institute of Engineering & Technology, Radaur, Haryana in 2012-2014. Her research interests include Software Engineering, Web Technologies, Cloud computing.