International Journal of Electrical and Computer Engineering (IJECE)
Vol. 6, No. 4, August 2016, pp. 1828~1838
ISSN: 2088-8708, DOI: 10.11591/ijece.v6i4.9956

Disaster Recovery Services in Intercloud Using Genetic Algorithm Load Balancer

Tamanna Jena[1], J. R. Mohanty[2]
[1] School of Computer Engineering, KIIT University
[2] School of Computer Application, KIIT University

Article Info

Article history:

Received Jan 19, 2016
Revised Jun 23, 2016
Accepted Jul 10, 2016

Keyword:

Cloud computing
Genetic algorithm
Green cloud computing
Intercloud
Load balancing

ABSTRACT

Paradigm need to shifts from cloud computing to intercloud for disaster recoveries, which can outbreak anytime and anywhere. Natural disaster treatment includes radically high voluminous impatient job request which demands immediate attention. Under the disequilibrium circumstance, intercloud is more practical and functional option. There are need of set of protocols like quality of services, service level agreement and disaster recovery pacts to be discussed and clarified during the initial setup to fast track the distress scenario. Orchestration of resources in large scale distributed system having muli-objective optimization of resources, minimum energy consumption, maximum throughput, load balancing, minimum carbon footprint altogether is quite challenging. Intercloud where resources of different clouds are in align and plays crucial role in mapping capability. The objective of this paper is to improvise and fast track the mapping procedures in cloud platform which can address impatient job requests in balanced and efficient manner. Genetic algorithm based resource allocation is proposed using pareto optimal mapping of resources to keep high utilization rate of processors, high througput and low carbon footprint. Decision variables include utilization of processors, throughput, locality cost and real time deadline. Simulation results of load balancer using first in first out and genetic algorithm are compared under similar circumstances.

Corresponding Author:

Tamanna Jena,
School of Computer Engineering,
KIIT University,
Bhubaneswar, Odisha, India.
Email: tamannasinghdeo@gmail.com

1. INTRODUCTION

Cloud Computing is a highly integrated and flexible computing service where user need not own infrastructure, server, datacenter etc. Using a device like smart phone, tablet, PC etc. enabled with internet, provide access to computability of cloud provider while paying for its usage. Cloud computing gave us the room for mobility to access data from anywhere unlike traditional setups where we need to be in the same location for data availability. Datacenters basically use hypervisor based virtualization to execute service. Cloud Computing is basically a new metaphor for subscription based utility service like electricity, telecom, internet etc. Its effect on human life and environment is discussed in many contradictory ways. There are three models of cloud services: Infrastructure as a service (IaaS), Software as a service (SaaS), and Platform as a service (PaaS). Google Apps and Salesforce.com are of type SaaS, where user uses the applications available without any control over the host. In PaaS, platform is provided which is more or less an application framework. Examples using PaaS are Google App Engine, Amazon Web Services (AWS) etc. User uses computability of resources, storage, processing power etc. in IaaS, example are Eucalyptus. Precisely user/clients can lease computing resources from IaaS, use applications from SaaS and can use as well as develop applications on the platform provided at PaaS cloud. Cloud has attracted huge volume of users because of its elastic, flexible, connectionless, high computability, investment free, user friendly, and secured storage capacity services. Users do not get the transparency of its execution until and unless it is explicitly demanded as per service level agreement (SLA). Performance of cloud depends highly on the availability and strength of internet. Slow communication increases waiting, thus increasing makespan of application altogether. Like all technology "cloud" has its share of pros and cons. It is noticed that because of huge commercial aspect, large number of public and private clouds are created in last half decade, but in most of the cases utilization is less than 50%. Leap in performance is often proclaimed whereas rise in carbon footprint is ignored. State of art of energy efficiency on large scale distributed system and its causes are in literature of many research papers [1],[2]. Carbon emission caused due to over provisioning, server spooling and cooling datacenters is immensely contributing towards global warming. Its networking infrastructure is quite strong, with high potential for a better environment with lesser carbon emission, if done right from its infancy or adoption of newer techniques like server consolidation, thermal management, optimal load balancing etc.

Nowadays almost all businesses are moving to cloud to reduce their cost of infrastructure, maintenance, carbon footprint, staffing [3]. In order to keep the carbon emission low, trade off between crucial features like privacy, performance and extent of virtualization are evaluated iteratively. Cloud federation is an integrated cloud structure which comes into existence with better fault tolerance and cleaner world concept. It potential to reduce digital waste and carbon emission is huge. Cloud computing model, where numerous user utilizes single cloud datacentre struggle several challenges. In order to incorporate ECC (Elastic Cloud Computing), most cloud provider establish large datacenter. They sometime reserve large capacities to incorporate sudden voluminous job requests, which result underutilized resources in most cases. Carbon emissions of large setup contribute towards technical waste and energy waste. These above factors combinedly contribute towards global warming. It is an open question for all humanities. Evolution of mechanics and technology has taken a big toll on nature as a result we are encountering natural disaster more frequently than ever. Abrupt number of job requests having different job locality can be raised at the same time which may or may not be interdependent (complexity increases more than two folds in case of interdependent impatient job requests). Agencies like government, military, health, security can be impatient at times. From user perspective single cloud datacenter reliability is inadequate. For a reliable, flexible and greener approach integrated clouds is a smarter energy efficient option.

1.1. Intercloud and its components:

Intercloud is an e-integrated infrastructure having extended computational capability for utilization of resources and storage capacity. Its model is like a seamless functionality mesh where each node is a cloud or its component. In other words, zoomed out multiple cloud net is intercloud. The interoperating between multiple clouds need to be in compliance among its member. It is a powerful platform, can be real rewarding when its strength are exercised efficiently. Here job request is a service to be executed by cloud for any user/enterprise. Job requests are mainly divided into small tasks. Each task can be executed by one or multiple cloud. Objective is optimum utilization of resources in accordance with SLA, which apparently lessens carbon emission. Job requests are divided into many categories like nature of service (IaaS, PaaS and SaaS), size, complexity, location etc. Job-requests are sometimes divided into CPU intensive tasks and input/output intensive tasks. Research on intercloud architecture and agreements is a virgin topic, neither well defined nor understood enough. In our paper architecture of intercloud is expressed in figure1. It even suffers from terminology ambiguity. A state of art of inter-cloud is done to an extent in [4]-[7].

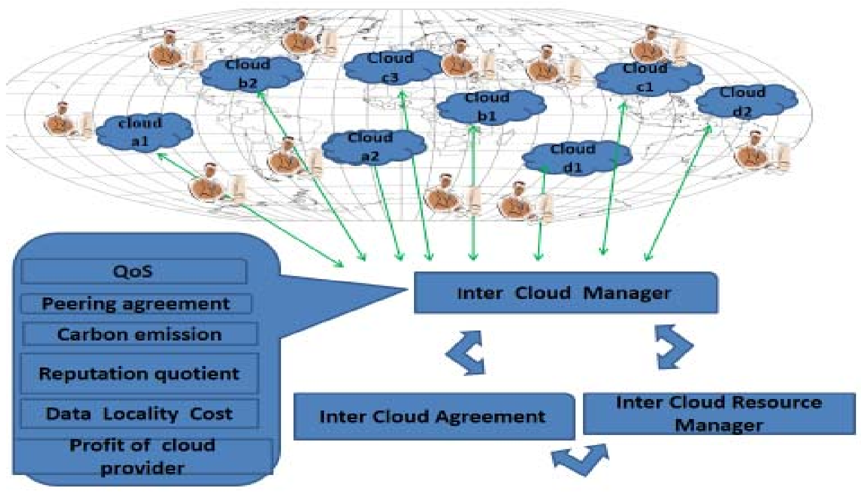

Figure 1. Architecture of Intercloud

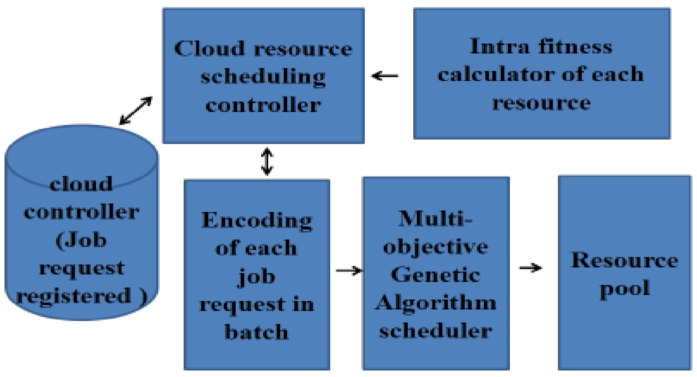
Cloud resource scheduling controller
Intra fitness calculator of each resource
cloud controller (Job request registered)
Encoding of each job request in batch
Multi-objective Genetic Algorithm scheduler
Resource pool

Figure 2. GA Based load balancer

Components of Intercloud:

- Intercloud Manager
- Intercloud Agreement (SLA agreement between peer cloud providers)
- Inter Cloud Resource Manager (health, availability and usage of each resource is audited periodically)
- Reputation Manager (performance of each component and feedback from user is documented, which reflects in its reputation in market)

Motivation for Intercloud**:**

- It is futile and incompetent to have a single cloud provider datacenter worldwide to combat disaster recovery. Peer providers need to be well integrated for common goal, providing QoS and worldwide compliance with distributed resources. Moving to intercloud is the next best logical option for betterment of both providers and users perspective inorder to avoid servers pooling, reduce carbon-footprint etc. To put check on carbon foot-print and energy waste by optimal utilization of each component of intercloud.

It is high time to incorporate green networking, computation and communication. Contribution of wireless technology and its impact on environment is detailed in literature [8]-[10]. The International Telecommunication Union Telecommunications Operations (ITU-T) operates on monitoring standard on an international basis for fair and competitive market. There is alignment between telecommunication department and environmental issues caused by them. It is found that networking waste is substantially high at end user site. This can be addressed by opting wired internet at user level or directional antennas. When equilibrium is attained between job-request and resources then pressure shifts to networking. Faster networking protocol which is introduced in 2000, called “InfiniBand (IB)” is considered as an advanced energy efficient alternative to traditional TCP/IP. It allows reading and writing data from remote computer’s memory by effectively by passing the operating system [11]. Apart from its huge potential it is not used much in Hadoop or cloud platform as it suffers from the misconception of being expensive. Intercloud services are divided into three complimentary components discussing the integration and inter-operability of multiple layered Cloud Model Device: 1] Intercloud federation 2] Intercloud control and compliance management 3] Intercloud multi layer interface among IaaS, PaaS and SaaS [12].

Intercloud should satisfy following goals:

- Components of intercloud should support communication between different service layer (vertical integration), different domains and applications and services (horizontal integration).
- QoS (Quality of Services) should be maintained across both horizontal as well as vertical integration along with optimization of computing and storage.
- Rule out vendor lock-in: Vendor lock-in is an unfortunate situation, in which user cannot transit its service from one provider to its competitor. It is found out to be the major impediment for enterprises to adopt cloud. Incompatible business proprietary technologies are the hurdles, as a result adaptability of provider and of user requirement are not in compliance.
- EC2 (Elastic Cloud Computing) is limited when restricted to single provider. Sudden spikes of user requirement cannot be accommodated by any single cloud provider in all times without causing perilous power consumption and exponential high cost.

In order to wrap all the attractive features of cloud, it is in the best interest of both provider and customer to have a high standard legislation of cloud federation. Protocol regarding QoS, deployment, maintenance, resource allocation, arbitrage, reputation quotient, peering agreement etc. need to be thought through minutely. Intercloud being novel, it suffers from ambiguous terminology. Artificial intelligence tools have been used as optimization scheduler in cloud platform. When search space is huge, constraints are multi objective and deadline is strict, CI (Computational Intelligence) tools are highly productive. In our paper we are using the pareto-optimal solutions concept for resource allocation technique in which, each solution is evaluated under multiple criteria. Subset of options is determined, such that there is no room of improvement in the provided solutions. Health of each component is evaluated iteratively. When one or multiple component (basically server, datacenter, internet, network) of cloud crosses lower threshold of fitness by natural/accidental/technical disaster, then re-routing of resource allocation is needed. These scenarios are recurrent due to many reasons such as over provisioning of resources, sudden spike of impatient job requests, low internet bandwidth, high internet traffic, high carbon footprint, high electricity consumption for cooling datacenters, power failure and natural calamity.

Load balancing is one of the most influential aspects of cloud computing, it can be broadly divided into centralized or decentralized and dynamic or static. Another concept used in this paper is Genetic Algorithm (GA), a stochastic optimization tool using Darwin theory which gives better result when search space is very high. All the possible solutions are termed chromosomes; fitness value of each chromosome is evaluated. Selection procedure is used to filter some solutions to the next generation pool. Operators of genetic algorithm are used on the chromosomes to obtain optimized schedule. Each scheduling strategy has its strength and weaknesses. Implementation time is very fast and gives better result whereas premature convergence is its down side. Consequently load balance of each server and virtual machine is maintained to avoid work overload. Virtualization of processor is used in simulation to encourage multitenancy. Optimum utilization of resources and equilibrium between demand and supply can make intercloud as the most beneficial utility service of the era.This paper is organized as follows section 2 provides related research on cloud computing, intercloud and scheduling used in cloud platform. Section 3 analyzes our model using optimization techniques called GA based load balancer. Section 4 describes result and discussion on simulation. Section 5 discusses conclusion.

2. RELATED RESEARCH

Architecture, techniques and role of components of cloud computing is mentioned in the literature of many research papers [2],[10],[12],[13]. Power management in cloud platform is one of the impactful topics. Large chunk of energy consumption is used in cooling datacenters. To lessen the power usage in cooling datacenters, load balancer plays vital role. Nature of job-requests are either interdependent or independent. Interdependency between jobs/tasks is represented by TIG (Task interaction Graph) to show mapping of different applications [14]. Complexity of execution of interdependent tasks becomes more than two folds in case of dependent job-requests. They have considered heterogeneous job-requests which are executed concurrently and communicating with each other at the same time. Job-requests faces queuing in getting the required resources [15]. Have used "Request dependent strategy" technique by scheduler to map job-requests to resources. They have tried to establish the relationship between job arrival rate, rate of execution on waiting–time of job-requests and rate of emptiness of queue. The drawback of request dependent strategy is if number of incoming job-request decreases then its impact on utilization rate of servers is more severe than other schedulers. Since specific servers are allotted to certain type of jobs sometimes causes server sprawl. Job-request undergo queuing once registered with cloud provider oe intercloud. Types and trends found in queuing cannot be overlooked. Different features of queuing are described spanning balking, reneging and jockeying in cloud [16].

Balking is a situation where customer decides not to join the queue if it is too long. In Reneging, customer leaves the queue after having waited for too long for service. Jockeying describes the extent of impatient users, where customer shifts between queues to be served fast. They have tried to establish the relationship between user balking/reneging and system blocking on the throughput rates. Nature of digital users is analyzed which impact throughput and usage of resources. In [17], they proposed an algorithm where bandwidth are shared between VMs dynamically, in order to map impatient job requests by Minimum Completion Time (MCT) scheduling algorithm. However precise analysis on performance are found missing. Clustering job-requests in accordance to resource processing capability is done in [18]. They have suggested job-grouping algorithm in which user tasks are teamed according to resource processing capability and grouped tasks are assigned to different resources in cloud computing. Since real life experiments of intercloud are very expensive and difficult to implement, often simulations are wiser option to enhance the performance.

Optimum task consolidation using GA for both long term and short term is proposed in [19], she suggested that resource utilization directly relates to energy consumption. Process of intercloud is mimicked in [20] using SimIC tool kit and CloudSim simulator. It is comparatively easy to setup multiple datacenter, networking aids, storage in simulators. It is found that results of SimIC compliments CloudSim. Scenarios which shows failure basically needs effective dynamic mapping technique. A mathematical model is presented, where virtual machine placement is optimally allocated whenever bandwidth capacity is available and network latency is low [21]. They have used performance-oriented VM allocation approach in cloud computing platform. Similar paper based on performance where focus is on latency are [22], they found that transferring data between intra-cloud providers are more cost effective than inter-cloud provides (datacenters of same cloud provider but different geographically distributed location). They suggested larger RI purchase than required is better in most cases. Although buying larger capacity can contribute towards server spooling. Many papers have used CI tools to map job-request to resources. Genetic algorithm is used as load balancer in [23], where size of nodes required are decided depending on the workload. Simulation results towards cloud node minimization shows number of nodes available is twice the workload.

Advantages of load balancing is outlined whereas scenario mapped is not appropriate. Similarly [24] have used fuzzyNN scheduler and Berger Model scheduler to compare the performances of different schedulers. Constraints considered in simulation are completion time and bandwidth utilization. Cloudsim simulator is used to map tasks to resources; mapping time and makespan time are the performance metrics in [25]. They have included time taken to stage in the input files and stage out the output files to desired location. Load balancing in cloud computing done by various algorithm, tabular representation of research done on load balancing using various evolutionary and swarm based algorithm is summarized in [26]. Cloud users are categorized on the basis of job-length, nature of user (domestic or commercial) and time of execution (peak or off-peak). Different pricing strategy is used to promote off-peak job and penalize voluminous job-request at peak hours [27]. Conscious contribution from both i.e, user and provider can make difference in green cloud computing.

VM migration is suggested in [28] to reduce workload of overloaded node. Deployment of VM migration causes data overhead and secutity system is attack prone. A two layered cluster as a service is proposed as disaster recovery services. First layer manages the operating system of the physical nodes forming the clusters. Second layer, deploys the software components to install on the nodes Disaster recovery in intercloud system by integrating cluster as a service and is done by using continous-time Markov chains [29]. Backup of all data of each node of clusters are saved in encrypted form with different organization placed in distant geographical location. Single backup seems inadequate and different provider may cause confidentiality issues. Load balancer based on GA is used in [30], where cloudanalyst simulator is used. They have considered single centralized cloud data center which is still a far-away dream. Different time-zone and different data-locality cost are not taken into consideration.

3. MODEL DESCRIPTION

Each job request comprises of group of input files, group of output files, QoS requirement, job-length size (expressed in MIPS), and arrival time and data location. In our simulation each job-request is divided into tasks like compute intensive and I/O intensive tasks. Each task is allocated to a virtual machine. Virtual machines are the basic computational units of datacenters. We have considered 10-15 physical machines, and each physical machine is capable of extending 10 virtual machines. Let D be the set of cloud providers/datacenters in which D= $\{d_1, d_2,\dots, d_{|D|}\}$. Let J be the set of job-requests which are registered with the cloud controller, either waiting in queue or being executed, such that J= $\{j_1, j_2,\dots, j_{|n|}\}$. Let T be set of tasks such that, T= $\{(t_{11}, t_{12,\dots,}t_{1m}), (t_{21}, t_{22,\dots,} t_{2m}),\dots,(t_{n1}, t_{n2,\dots,}t_{nm})\}$, where m represents number of tasks a job-request is divided. In our simulation we have taken m ranges from 0 to 5. Let QoS (t) be the set of QoS requirement for task, Fin_t is the set of input files for task t and $Fout_t$ is the set of output files for task t. Each task has two deadlines, namely sdt (start deadline) and fdt (final deadline). The major objective of this work is to optimize the mapping process in such a way that maximum number of job requests gets executed with least possible resources without compromising SLA constraints. Objective function of GA is maximizing the throughput, minimizing the number of physical resources and minimizing the makespan time. Throughput=ΣT_i, T is number of job requests whose execution is completed before deadline.

Objective function= max_throughput, min_datalocality cost, min_makespan, min_physical_resources

Since MOGA (Multi Objective Genetic Algorithm) is used based on the process of natural evolution, time spent on live migration of tasks is not considered within the scope of the paper. Energy loss is identified as energy which is not consumed by any sub system and overhead of the supporting subsystems [2]. Components of our model is illustrated in Figure 2.

Model constitutes of following components:
- Cloud resource scheduling controller
- Cloud Federation Controller
- Intra fitness calculator of each resource
- Multi-objective Genetic Algorithm load balancer
- Cloud controller (which receives the job request)
- Virtual resource pool

Constraints of model:
- Each job request need to be executed by deadline, otherwise it is supposed to be failed.
- Each job request can be allocated to more than one cloud.
- Important constraints are throughput, makespan of job requests, deadline, datalocality cost and QoS.

Encoding Rule: Encoding is the designing chromosome process in such a manner that tasks and resources are encoded in one chromosome. Each chromosome is represented as a 2 **M* matrix, where, *M* is the chromosome length. The first row of the matrix represents the requested applications, and second row of the matrix corresponds to the cloud where the application is executed. Here job requests are registered, assigned unique id, and then controller maps the ready job request to available resources using GA. Basically job requests are divided into multiple tasks. Tasks accumulated in every 5secs form a batch. Length of chromosome depends on the length of tasks in the batch. If 25 tasks are collected in a batch then the chromosome is 2*25. We have randomly generated 300-400 job-requests from 20 users. Length of job-request, task, and server capacity are expressed in MIPS. Number of tasks in a batch is equal to number of virtual machines. Out of 10 servers, we have randomly categorized them into 3 different data locations, having different data locality cost.

Table 1. Allocation of tasks to different servers

Job Request	A1	A2	A3	A4	A5
Tasks	CA1, IA1	CA2, IA2	CA3, IA3	CA4, IA4	CA5, IA5
Cloud Provider	(2,7)	(8,9)	(2|4, 6|3)	(1|9, 4|2)	(3,4)

Set of cloud providers having same data locality cost are grouped together, eg. {1,2,3}, {4,5,6} and {7,8,9,10 } are clouds having same data locality. Although current scenario is cloud providers only charges for outbound storage whereas inbound storage is free. Both inbound and outbound incurs high carbon footprint in both inbound and outbound of large amount of data.

In Table 1, job request A1 is divided into CPU intensive and I/O intensive job tasks i.e, CA_1 and IA_1. Cloud providers 2 and 7 are the options available for execution of the job requestA1. Similarly in case of job request A4, cloud provider 1 and 9 both are available for CPU intensive whereas cloud providers 4 and 2 for I/O intensive. Charges of processing rate depend on the data locality of the datacenter, bandwidth rate, CPU utilization rate and traffic of job requests.

Though computing is dynamic but at any instance of time the load balancing problem is formulated as “P number of jobs submitted by cloud users to Q number of processing units in the cloud”. Computability Vector (CV) of each processing unit indicates status of rate of utilization, expressed in MIPS. β is execution cost of the task and L is the delay cost / penalty (to be paid by provider in case deliverables fail to meet deadlines. Current status of the processing unit is expressed in equation (1) as follows. Equation (2) shows the status of job request represented by a Job Status Vector (JSV).

$$CV = f\,(MIPS, \beta, L) \tag{1}$$

$$JSV = f\,(t, MIPS_j, AT, d) \tag{2}$$

where”t” describes type of service required by the job, spanning IaaS, SaaS or PaaS. $MIPS_j$ represents the number of instructions present in the job determined by the processor; AT indicates the arrival time of job in the system and d is the minimum time required to complete the job by processor. ζ is the cost function which is minimized.

$$\zeta = w1 * \beta\,(MIPS_j / MIPS) + w2 * L \tag{3}$$

where w1 and w2 are predefined weights, depending upon the user preference or reputation value of the user.

3.1. Proposed method using Genetic Algorithm:

The strengths of Genetic Algorithm technique is that it works well when search space is vast, premature convergence can be avoided by dynamically changing the operators of natural selection. Steps includes as follows:

1. **Initialization**: Fixed length of chromosomes is generated for each batch. Initialization of chromosome is randomly generated. Here each job request is considered to be the set of sub-divided tasks called genome. Fitness values are calculated using equation (2). Elitism selection is used i.e., the chromosome having highest fitness value is selected and moved to the next generation mating pool. If the best fit chromosome of the current generation fitness value is lesser than the most fit chromosome of the previous generation, the later would oust the chromosome having least fitness value of the current population. Some pre-requisite data are index of datacenters, the list of VM for each server/datacenter,

Requirements of each VMs are:

$$TCT_{tvd} = VMC_{vd} + Sin_{tvd} + E_{tvd} + Sout_{tvd} \tag{4}$$

TCT_{tvd}=Task completion time, when job request t on virtual machine v on data center d completed its execution.
VMC_{vd} is the time to create virtual machines in the physical machine.
E_{tvd} is the execution time the task t runs on the virtual machine v of datacenter d.
Sin_{tvd} is the process of getting all the requisite input files and Sout is the process of transfering the executed result to the predefined destinations. Real number encoding is used in our simulation.

2. **Crossover**: The objective of this step is to produce new individuals by reforming parts of selected two chromosomes. Two chromosomes from the mating pool are selected and exchanged of genome parts at one or multiple point. In this paper one-point crossover is implemented which is selected randomly. Probablility of crossover (p_c) and probability of mutation (p_m) influences the degree of solution accuracy and rate of convergence in implementing GA.

3. **Mutation**: As we have used real number encoding, mutation is different than binary encoding. Unlike binary bits of chromosomes are not toggled from 0/1 or 1/0.

Selection, crossover and mutation are used iteratively till the optimal solution is obtained or maximum number of iteration is reached. The final chromosome obtained is the best possible mapping solution for the batch of tasks collected per that instance of time. A hypothetical configuration has been developed. We have considered 3 regions. Servers/datacenters are distributed in these regions having respective data locality cost. Single time zone has been considered for all regions.

Length of each job request varies between 5 to 20 MIPS. Number of datacenters considered ranges from 2 to 20. Each physical machine can produce VM upto 10. We have randomly assigned data location cost to the geographically distributed data centers.

4. **Selection**: Most fit chromosome is selected and taken to the next generation mating pool. Ellitism selection was used. The above processes are iteratively executed till number of generation is attained or

convergence of output incur. It is tested and proved theoretically that genetic algorithm with ellitism selection operator have great global convergence.

4. RESULTS AND DISCUSSION

Testing the proposed model on real system being in-executable, simulation is done using MATLAB. Load balancing in cloud platform is a NP complete problem in real time. Techniques proposed till date own its share of strength and weaknesses. Our work accomodates optimal trade off between: arrival rate of job-requests, number of active physical machines and real time load balancing using GA along with high server utilization rate without compromising on QoS.

First generation influences the quality of the final outcome produced in concluding generation. It is one of the vital step in the entire algorithm. In this paper, initialization is done randomly to avoid premature convergence and GA operators are selected by rigorous simulation, in order to obtain optimum mapping combination between number of active servers and impatient tasks keeping high throughput and minimum makespan time. Each possible mapping is assessed under multiple criteria. Set of allocations obtained in simulation shows pareto efficient. In accordance to Pareto set allocation of resources, it is impossible to make any option best without making atleast one option worst. Subsets of possible options are also identified. By gathering all of the concieveably optimal solutions, a designer can make focused and trialed tradeoffs within this constrained set of parameters, rather than needing to consider the full ranges of parameters.

We found that our model shows better load- balancing results and even suggests healthy balance between numbers of physical machines required with respect to arrival rate of job-requests. More number of active servers does not contribute any value addition only enhances the maintenance cost. Here we have taken job-requests of diverse length. We found that switch on/off of physical machine is a beneficial proposal only when it can remains idle for a long time(enough to compensiate the switch on/off overhead) and it also suffers long waiting of ready jobs in queue by the time physical machine is lined up for service. [31] have compared adaptive genetic algorithm (AGA) and adaptive algorithm-job spanning time and load balancing genetic algorithm (JLGA) to achieve load balancing with least makespan but have considered only 30 job-requests using comparatively higher number of nodes. Comparatively fewer number of job-requests (ranges from 10 to 100) with less diversity in job-requests size are used for mapping [26]. Mainly all job-requests are CPU intensive and bulky. We have done simulation on more than 300 job-requests. Job-requests are of various sizes and capacity. We have simulated with wide range of arrival rate, computability of servers and diversity in job sizes. Our model is quite flexible and adapted well with sudden change of workloads. Number of job-request produced ranges from 200-400 in each simulation which is further divided into numerous tasks.

Simulation result shown in Figure 3 shows, the makespan time of total jobs (here mainly high CPU intensive job-requests are considered) where workload is distributed in a balanced way among available multiple resources when rate of arrival of job-requests is 10. X-axis refers to the number of active servers and y-axis expresses total makespan time of job-requests held in the batch, which are mapped to resourses in a balanced manner using GA and FIFO. In Figure 4, where arrival rate is 20. Results suggests that the minimum makespan time can be achieved when number of active servers is 14. Our model helps to attain the optimum trade off between supply and demand while keeping QoS intact and completion of job execution in real time. In Figure 5, where α is 30 shows that minimum makespan time of job execution can be achieved using 12 servers. Distribution of job-requests are efficiently mapped to resources. Figure 6 shows the plot pattern when arrival rate is 40, here optimum distribution can be achieved when servers used are 10. Graph indicates considering 10 servers is the optimum solution whereas considering 16 servers when makespan time is least. Overhead of switch on/off can be estimated here by using simulation result. Our model evenly distributes the workload within real time in such a manner that utilization of processors are kept high. It helps in deciding the performance of the entire system. It suggests smart decision making regarding available resources and its usage. Depending on the dynamic parameters like availability of servers, arrival rate of job-requests and even computability of servers which keep on changing in real life, mainly in unconventional circumstances then the optimal scenario can be proactively expressed using our model. During disaster recovery sudden change of arrival rate of job-requests can put extra stress on available resources and completion of important job-requests within estimated deadline is challenging as well as crucial.

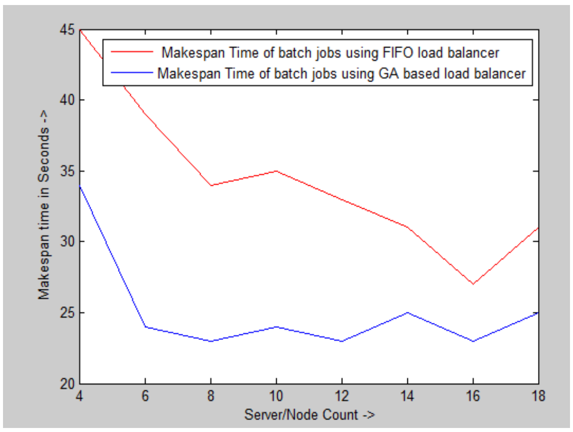

Figure 3. when α is 10

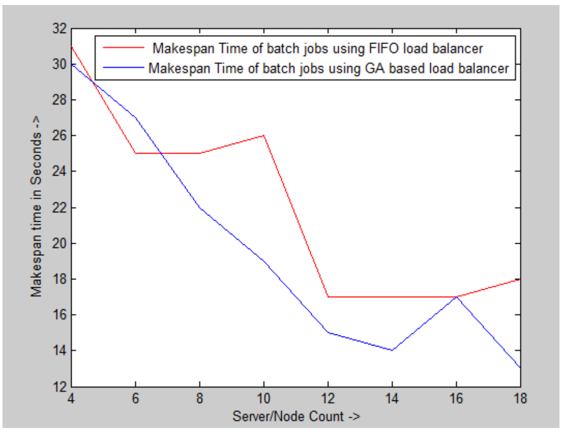

Figure 4. when α is 20

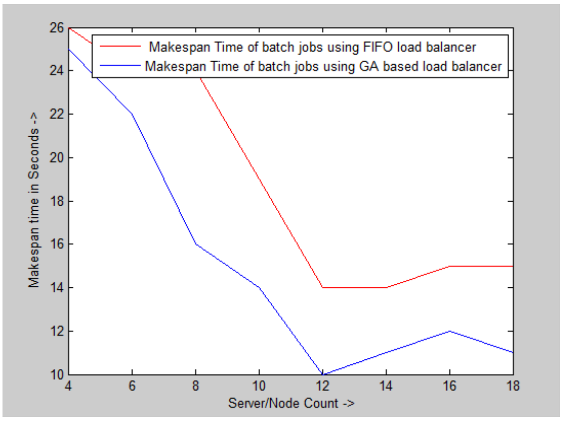

Figure 5. when α is 30

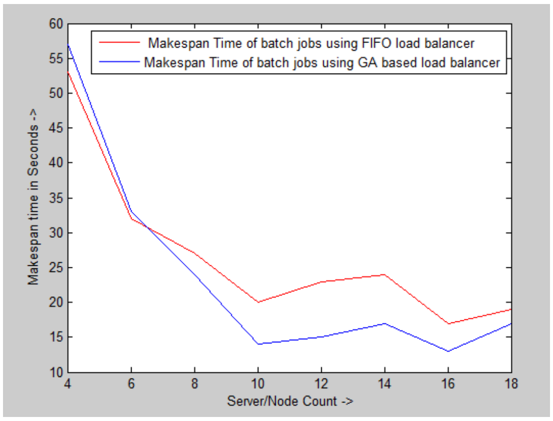

Figure 6. when α is 40

In all cases GA load balancer is more efficient than FIFO. Arrival rate ranges from 4 to 40 and diversity of job-requests exists, we tried to load balance the incoming job-requests using multiple but dynamic number of servers. We tried to establish a trade-off between arrival rate and number of active physical machines. Sometimes in disaster recovery switch on and switch off physical machines are less rewarding. Power management techniques that feat features of virtualization to save energy are highly appropriate with the impatient scenario. Special focus is given to power management techniques that exploit the virtualization technology to save energy. A similar work is done in [32] where hybrid genetic algorithm along with greedy algorithm is used for resource scheduling. Live migration of workload from heavily loaded virtual machines are shifted to idle ones. Their proposed model shows better elasticity but associated overhead and expenses of live migration as well as data locality cost which is possibility due to latency arbitrage is not considered. Strength of our model is negligible time spent for optimum resource mapping. Basically it improvises the resource allocation procedure in fast and efficient way. Rate of emptiness of queue is high (mapping technique in GA is better than FIFO, therefore same batch of job-request takes comparatively lesser time when GA based load balancer is used) in case of GA than FIFO.

5. CONCLUSION

Load balancer using GA plays crucial role in performance of intercloud platform. Combat process of disaster need accurate and thoughtful precision in performance, as incoming job requests could be impatient and erratic, resources will be limited and time constraint will be steep. Establishing trade off between demand which indicate impatient job-requests and supply specifies available resources here is tough, when rate of arrival of jobs is sporadic, deadlines are stern and carbon-foot-print is to be restricted. GA based load balancer shows better result than traditionally used FIFO regarding optimum allocation of job-requests to resources. Queuing used here helps in efficient task mapping among multiple homogeneous servers having different data locality cost and strict real time constraint. GA based mapping procedure is very fast and efficiently distribute workloads evenly among processors. Decision variables considered are CPU usage, throughput, data locality cost and makespan time. Throughput is higher in GA load balancer than FIFO. Server utilization rate is found to be high in all cases when GA based load balancer is used resulting lower carbon foot-print.

REFERENCES

[1] Orgerie A. C., *et al.*, "A survey on techniques for improving the energy efficiency of large-scale distributed systems," *ACM Computing Surveys (CSUR),* vol/issue: 46(4), pp. 47, 2014.

[2] Mastelic T., *et al.*, "Cloud Computing: Survey on Energy Efficiency," *ACM Computing Surveys (CSUR),* vol/issue: 47(2), pp. 33, 2015.

[3] "Accenture in collaboration with WSP Environment & Energy," *Cloud Computing Sustainability: The Environmental Benefits of Moving to the Cloud,* 2010. http://nstore.accenture.com/.../ccr/2010-2011/Accenture-Sustainability-Cloud-Computing-TheEnvironmentalBenefitsofMovingtotheCloud.pdf.

[4] Bernstein D., *et al.*, "Blueprint for the intercloud-protocols and formats for cloud computing interoperability," in *Internet and Web Applications and Services. ICIW'09. Fourth International Conference IEEE*, pp. 328-336, 2009.

[5] Abawajy J., "Determining service trustworthiness in intercloud computing environments," in *Pervasive Systems, Algorithms, and Networks (ISPAN), 10th International Symposium IEEE*, pp. 784-788, 2009.

[6] Bittencourt L. F., "Madeira ER, Da Fonseca NL. Scheduling in hybrid clouds," *Communications Magazine, IEEE*, vol/issue: 50(9), pp. 42-47, 2012.

[7] Toosi A. N., "Calheiros RN, Buyya R. Interconnected cloud computing environments: Challenges, taxonomy, and survey," *ACM Computing Surveys (CSUR)*, vol/issue: 47(1), pp. 7, 2014.

[8] J. F. Morais and G. L. Siqueira, "Wireless technologies environmental impacts," *Microwave and Optoelectronics Conference (IMOC) SBMO/IEEE MTT-S International*, pp. 523-527, 2009.

[9] Brooks T., *et al.*, "Conceptualizing a secure wireless cloud," *International Journal of Cloud Computing and Services Science (IJ-CLOSER)*, vol/issue: 1(3), pp. 89-114, 2012.

[10] Chauhan S., "Environmental and Health Hazards of Mobile Devices and Wireless Communication," 2002. http://crystal.uta.edu/~kumar/cse6392/.../Savita_paper.pdf.

[11] Bermúdez A, *et al.*, "Fast routing computation on InfiniBand networks," *Parallel and Distributed Systems, IEEE Transactions,* vol/issue: 17(3), pp. 215-226, 2006.

[12] Demchenko Y., *et al.*, "Defining Inter-Cloud architecture for Interoperability and Integration," *The 4th IEEE Conference on Cloud Computing, (CloudCom2012)*, Taipei, Taiwan, 2012.

[13] Rathor V. S., *et al.*, "Survey on Load Balancing through Virtual Machine Scheduling in cloud computing Environment," *International Journal of Cloud Computing and Services Science,* vol/issue: 3(1), pp. 37, 2014.

[14] De Falco I, *et al.*, "Two new fast heuristics for mapping parallel applications on cloud computing," *Future Generation Computer Systems 37*, vol. 37, pp. 1-13, 2014.

[15] S. Varma P., *et al.*, "Performance analysis of cloud computing using queuing models," in *Cloud Computing Technologies, Applications and Management (ICCCTAM), 2012 International Conference IEEE,* pp. 12-15, 2012.

[16] Chiang Y. J., *et al.*, "Performance and Cost-effectiveness Analyses for Cloud Services Based on Rejected and Impatient Users," *SeviceComputing, IEEE Transactions on Services Computing*, vol. 99, 2014.

[17] Mehdi N., *et al.*, "Virtual machines cooperation for impatient jobs under cloud Paradigm," *International Journal of Information and Communication Engineering,* vol/issue: 7(1), pp. 13-9, 2011.

[18] Selvarani S. and Sadhasivam G. S., "Improved cost-based algorithm for task scheduling in cloud computing," in *Computational intelligence and computing research (ICCIC), IEEE international conference on,* pp. 1-5, 2010.

[19] Yamini R., "Power management in cloud computing using green algorithm," *In Advances in Engineering, Science and Management (ICAESM), IEEE International Conference on,* pp. 128-133, 2012.

[20] Sotiriadis S., *et al.*, "Towards inter-cloud simulation performance analysis: Exploring service-oriented benchmarks of clouds in SimIC," *InAdvanced Information Networking and Applications Workshops (WAINA), IEEE 27th International Conference on,* pp. 765-771, 2013.

[21] Justafort V. D. and Pierre S., "Performance-aware virtual machine allocation approach in an intercloud environment," in *Electrical & Computer Engineering (CCECE), 2012 25th IEEE Canadian Conference on,* pp. 1-4, 2012.

[22] McGrath M. G., *et al.*, "Intercloud Networks Performance Analysis," *InCloud Engineering (IC2E), 2015 IEEE International Conference on*, pp. 487-492, 2015.

[23] Al Sallami N. and Al Aloussi S., "A genetic algorithm in green cloud computing," *British Journal of Applied Science & Technology*, vol/issue: 7(2), pp. 179, 2015.

[24] Kumar V. V. and Dinesh K., "Job scheduling using fuzzy neural network algorithm in cloud environment," *Bonfring International Journal of Man Machine Interface*, vol/issue: 2(1), pp. 1-6, 2012.

[25] Mehdi N. A., *et al.*, "Impatient task mapping in elastic cloud using genetic algorithm," *Journal of Computer Science*, vol/issue: 7(6), pp. 877, 2011.

[26] Rana M. M., *et al.*, "A study on load balancing in cloud computing environment using evolutionary and swarm based algorithms," in *Control, Instrumentation, Communication and Computational Technologies (ICCICCT), IEEE International Conference on,* pp. 245-250, 2014.

[27] Jena T., *et al.*, “Paradigm shift to green cloud computing,” *Journal of Theoretical and Applied Information Technology*, vol/issue: 77(3), pp. 394-402, 2015.
[28] Kanakala V. R. and Reddy V. K., “Performance analysis of load balancing techniques in cloud computing environment,” *TELKOMNIKA Indonesian Journal of Electrical Engineering,* vol/issue: 13(3), pp. 568-573, 2015.
[29] Khoshkholghi M. A., “Cluster as a Service for Disaster Recovery in Intercloud Systems: Design and Modeling,” *International Journal of Cloud Computing and Services Science (IJ-CLOSER*), vol/issue: 3(3), pp. 163-171, 2014.
[30] Dasgupta K., *et al.*, “A genetic algorithm (GA) based load balancing strategy for cloud computing,” *Procedia Technology*, vol. 10, pp. 340-347, 2013.
[31] Wang T., *et al*., “Load balancing task scheduling based on genetic algorithm in cloud computing,” in *Dependable, Autonomic and Secure Computing (DASC), 2014 IEEE 12th International Conference on,* pp. 146-152, 2014.
[32] Chen S., *et al.*, “A cloud computing resource scheduling policy based on genetic algorithm with multiple fitness,” in *Computer and Information Technology (CIT) IEEE 12th International Conference on,* pp. 177-184, 2012.

BIOGRAPHIES OF AUTHORS

Tamanna Jena is currently pursuing her PhD in School of Computer Science Engineering, KIIT University, Bhubaneswar, Odisha (India). She received her B.E. degree in Computer Science and Engineering from Biju Pattanaik University of Technology, M.Tech in Computer Science from Biju Pattanaik University of Technology in 2012. Her current research interests include Cloud Computing, Computational Intelligence and Data Mining.

Dr. J. R. Mohanty is an Associate Professor in the School of Computer Application, KIIT University, Bhubaneswar, Odisha (India). He is in this field since 19 years. He earned his Ph.D. Degree in Computer Science from Utkal University, India. He has published eight number of research papers in peer reviewed International Journals and thirteen number of research papers in International Conferences. He is a reviewer for several International Conferences. Edited one book published by Springer and two IJCA volume as International Conference Proceedings. He has written two chapters and reviewed two books of TMH. His research interest includes Queuing Theory, Computational Intelligence, and Cloud Computing.