Applied Mathematics & Information Sciences
An International Journal

http://dx.doi.org/10.18576/amis/13S131

A New Optimized Distributed Scheduling Service using Genetic Computing

A. Sandana Karuppan[1,*], *N. Bhalaji*[1] *and N. Ramaraj*[2]

[1]Department of Information Technology, SSN College of Engineering, Rajiv Gandhi Salai, OMR, Chennai-603110, India
[2]Department of ECE, Vignan's University, Guntur-522213, India

Received: 2 Mar. 2019, Revised: 2 May 2019, Accepted: 11 May 2019
Published online: 1 Aug. 2019

Abstract: Cloud computing is a paradigm to perform distributed processing and open computing environment. It hides the need of high-cost dedicated expensive hardware, space and dedicated software to minimal interaction and management. Enormous growth of data or big-data generated by cloud system has been recognized. Recently cloud computing is delivering on-demand services like software, memory, data, network-bandwidth and IT related services on Internet. The reliable performance of cloud-services can be related to various key-factors like task-scheduling. Scheduling can be done in different levels like job level or infrastructure level or process level. In this research work focus mainly on task scheduling method. End-user sends the request to the main data-center for jobs to be computing, tasks are named. A task means piece of work or process that can be executed within the deadline. Tasks are divided into critical and non-critical where critical tasks are executed with in the given time period, and non-critical consider as low priority work in the cloud system. Scheduling dispatches the all user tasks provided by the users of cloud system to the cloud service provider for available resources in the cloud.

Keywords: Cloud Computing, Resource Allocation, Utilization, Latency, Cloudsim

1 Introduction

Cloud computing is a paradigm to perform distributed processing and open computing environment. It hides the need of high-cost dedicated expensive hardware, space and dedicated software to minimal interaction and management. Enormous growth of data or big-data generated by cloud system has been recognized. Recently cloud computing is delivering on-demand services like software, memory, data, network-bandwidth and IT related services on Internet. The reliable performance of cloud-services can be related to various ey-factors like task-scheduling. Scheduling can be done in different levels like job level or infrastructure level or process level. In this research work focus mainly on task scheduling method. End-user sends the request to the main data-center for jobs to be computing, tasks are named. A task means piece of work or process that can be executed within the deadline. Tasks are divided into critical and non-critical where critical tasks are executed with in the given time period, and non-critical consider as low priority work in the cloud system. Scheduling dispatches the all user tasks provided by the users of cloud system to the cloud service provider for available resources in the cloud[1][2].

There are three types of scheduling approach in cloud distributed environment scheduling method based on resources, scheduling based work-process and scheduling method based on tasks. In our research, we focused Task-Scheduling approach (TSA) that allocates right tasks to right resource always through genetic algorithm by checking the feasibility of the tasks. Task scheduling methods only can be distributed. It performed in distributed environment on any tasks (sequential or atomic). The scheduling in centralized environment is less complexity, but, it eliminates the flexibility and decrease the success rate. Distributed Scheduling divided into two types such ad hybrid approach and heuristic method. Heuristic techniques are categorized into Static-Heuristics-Scheduling (SHS) and Dynamic-Heuristic-Scheduling (DHS). The DHS carried out in online mode. In SHS all the input of the tasks and VMs are known before scheduling. In DHS all the input

* Corresponding author e-mail: sandanakaruppana@ssn.edu.in

of the tasks are scheduled instantly, as it arrive in. DHS is better t an the SHS in all the way.

2 Related Works

A proposed task scheduling algorithm based on Heuristic Approach (TSA-HA) (Li et al 2010)[3], has been introduced for allocating and executing any type (both critical and non-critical) of tasks through implemented in cloud environments. The objective of this scheduling strategy targets to minimize overall execution time and cost for the tasks and finally achieved maximum utilization of VMs in the provided cloud setup. The scheduling algorithm has implemented through CloudSim. The completion time for the given TSA-HA is reduced by (41.83%) and (39.26%) about the default FCFS and RR algorithms along with general genetic approach. The future work is extended to add possibility dynamic characteristic of VMs through run GA. Moreover, more parameters can be added based on the user requirements. Now focuses on some of the important resource management techniques (RMT) (Manvi & Shyam 2014), such as resource-provisioning (RP), resource-allocation (RA), resource-mapping (RM) and resource-adaptation (RAD). This paper emphasize the comprehensive survey of RMT for IaaS in cloud computing, and also put forth the open challenges for further research. An optimized version of the FCFS scheduling algorithm to investigate the challenges of task-scheduling in cloud system. The all incoming jobs are grouped on the basis of task demand like minimal time of execution or minimize cost and prearranged (like FCFS manner). Task selection and VM selection is done on the basis of task parameters using a greedy method. This paper proposed model has been implemented and evaluated on CloudSim open source tool. We try to create a module depicting the normal FCFS algorithm in comparison to our other good strategy for resource provisioning scheme in the cloud. Demonstrated the usefulness of CloudSim by a case study involving dynamic provisioning of application services in the hybrid federated clouds environment (Singhal et al 2013). The outcome of this case study confirm that the aligned Cloud system model somewhat improves the QoS requirements of the applications under alternate resource and service demand model. The further work may extended to absorb new pricing and provisioning policies to Cloud system, in order to pass a built-in support to simulate the recently available public clouds. Other future directions of this work include absorbing: (i) workload pattern with load balancing; (ii) Patterns for DBMS services such as blob, SQL etc.; (iii) QoS observing capability of VM at Cloud level and (iv) Cost model for standard clouds to support economy-oriented resource provisioning studies.

An investigated resource scheduling service (Chen et al 2011) or strategy used by various researchers and classifies these services on the based on problems investigated, Services and the arguments used in evaluating various methodologies. The other aspect of modeling, worrying about in the area of Quality-of-Service (QoS)(Ackermann et al 2011), resource-optimality, green-computing VM-migration and task-scheduling. These open issues and further works will be extended to play a critical role in categorizing the technological road map for rising the future IaaS computing.

3 System Description

A task set consists of m independent periodic tasks$\{\tau_1....\tau_2\}$, of computation times $\{C_1;...;C_m\}$, periods $\{T_1;...;T_m\}$, and hard deadlines equal to the task periods. The utilization factor ui of any task i, defined as ui Ci=Ti, is assumed to be , where is the maximum reachable utilization factor for any task. Thus, is a parameter of the task set which takes the ?task sizes? into account. The total utilization of the task set, denoted by U is the sum of the utilization factors of the tasks of which it is composed.

Tasks are allocated to an array of n identical processors $\{P_1;...;P_n\}$. Once a task is allocated to a processor, it is executed exclusively on that processor. Within each processor, tasks are preemptively scheduled using fixed priorities assigned according to the RM criterion. Allocation is carried out using reasonable allocation (RA) algorithms. A reasonable allocation algorithm is one which fails to allocate a task only when there is no processor in the system which can hold the task. Whether a task fits into a processor depends on the single scheduling algorithm, the single cloud schedulability condition and the tasks previously allocated to the processor.

The following algorithm designed for multi-cloud systems from Min-min existing algorithm only for big tasks is extend to periodic and aperiodic with minimum complexity. In this system implemented in Hadoop-system with MapReduce model that is cloud application. The following procedure don?t miss its tasks deadline at any time, that is always. The designed system displayed below.

We consider two type tasks, there are critical and non-critical. For a critical problems, it is quite challenging process that how to keep QoS to each tasks at a maximum level and increase the complete fairness of the scheduling. The second option, in a structure of heterogeneous, all task?s execution is determined and that is made-up of multiple small-jobs corresponding to heterogeneous-services, and also mapped with a not consistent budget to refuse to give its overall payment.

In our system description, we consider different types of priorities to tasks in the state of scheduling and in there source allocation stages that brings-out importantly different outputs on the overall results and fairness.

Loop: foreach τ_i do

$$t = \sum_{j=1}^{i} c_i \tag{1}$$

continue = TRUE
while [continue] do

$$if \left[\frac{I_i^t}{t} + \frac{c_i}{t} \leq 1 \right] \tag{2}$$

continue = FALSE $/* \tau$ *is schedulable*/*
else

$$t = I_i^t + C_i \tag{3}$$

endif

$$if [t > D_i] \tag{4}$$

$c_i = c_i - [c_{i-1} + c_{i-2}, ... + c_{i-n}]$
goto Loop
else
exit
endif
endwhile
endfor
Where C_j - Worst case execution time.
I_t^i - Interference

However, we done the literature survey the best-fit queuing algorithms for increasing the overall goodness and results based on QoS. As for the requests or jobs in sequential policy, the candidate- queuing-model include shortest-optimal-length-first,lightest-workload-first,first-come-first-serve, shortest-subtask-first (a.k.a., min-min algorithms), and slowest progress-first. A shortest-optimal-length-first assigns higher priorities to the tasks with minimal-theoretically-optimal process the length calculated based on our convex-optimization model, it is similar to the heterogeneous earliest-finish-time(HEFT)[4]. The LWF and SSTF can be considered shortest-job-first (SJF) and min-min policy [5] respectively. An interested to apply idea of SPF is similar to earliest-deadline-first (EDF)[6], where as we accommodate two states to evaluate the entire progress of execution of jobs. Where, as we also study the best-fit scheduling policy for the jobs in parallel process model. The other best-fit scheduling policy include LSTF (longest-subtask-first), shortest-subtask-first, first-come-first-serve and the hybrid strategy queuing policy, e.g.,LWF+EDF.We apply the best-fit resource allocation model for critical jobs. Specifically, we show how tasks are distributed among the pool servers wherein maintaining the end-to-end delivered in the distributed cloud system. Based on the hybrid-cloud-service strategy, we demonstrate a distributed model that is ability to solve and determined NP-problems supplied by cloud requesters.

We also implement the best-fit option for selecting the arguments to our policy using genetic process. By executing our set-up performed on areal-cloud infrastructure with 56 VMs and 9 services with different execution instants. Our experiments is to demonstrating for the user jobs in parallel mode with dynamic way, the best-case performance under LWF+EDF is highest than that under other policies by at least 43 percent when overall resource requested amount is about twice as the actual resource that can be allocated. Another key point is in a critical situation, sub tasks (with the short-execution-length) are better to be assigned with more suitable resource amounts than the theoretically optimal values derived from the theory of optimization.As per our hybrid model (LWF+EDF) is produce best result in parallel operation for heterogeneous tasks. Which receive better solutions by 5.2 percent-60.3 percent than other.

4 Resource Allocation Scheme

4.1 Allocation of resources

Allocation of resources is activities of connecting all services together for utilization of the entire cloud platforms so as to meet the requirement of the all users of cloud. It required the type and quantity of cloud resources required by every user in order to finish their tasks. There are two parameters for optimization of the resource allocation, one is time and another one is order. Another important is how resources are organized for given user job. There are various views of abstraction of the cloud services in terms of developers, and various arguments can be optimized during allocation of resources. The construction of the resource model should consider these requirements so that resource allocation done properly.A cloud resource construction completed only the model can contain any resources for users and developers and cloud people request the anything at any time.

Allocation of resource is finished within VM allocated to each cloudlet with set of tasks. Scheduling process is shown in Figure 1, they mainly processed with four main modules as follows,

1.Models
2.Virtual machine
3.Tasks
4.Datacenter Brokerage

Our Hybrid cloud service system architecture is shown in above Figure 1. A tasks set consists of multiple tasks and subtasks are connected in two way either sequential or parallels.Each sub-task is an instance of Commercial-off-the-shelf (COTS) that has a suitable API to be called. Every task-set is expected to finish within the deadline (constraint of its budget). Task-scheduling layer apply the task priorities for scheduling each task-set. The

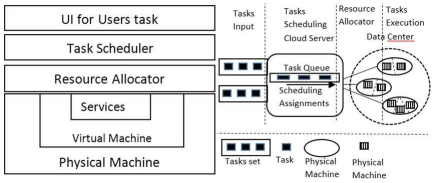

Fig. 1: System Framework

allocation of resource layer is responsible for allocating suitable cloud resource to internet-cloud-task and determines the best-fit-cloud resource in the cloud centric. All local machine runs more than one virtualized instance,on each of which are deployed with all of Commercial-off-the-shelf (COTS) (e.g., the libraries that do the computation).All sub-task or small task will be executed on aVM, then, reallocate the task-setto particular VM by VMM hypervisor. In a data center task failure to be calculated by VMM [7].

In Fig. 1. Shows, each task is processed using centralized random queue. All jobs are supplied contiguously over the time, and every jobs supplied to be investigated by GA analyzer. In order to predict the subtask workloads based on specified parameters. Sub-tasks workload can be characterized using queue profiler (separate queue for maintaining the executed task) wherein profiler is text file available in data center for providing information about resource availability [3]. Our novel method will determine the best-fit resource vector of all sub-tasks for the task-set. And then, the unexecuted sub-tasks with no-dependent preceding unprocessed sub-tasks will be put in a profiler, and waiting to be rescheduling. Notified later, the hypervisor of the selected local node will invoke a virtual machine and perform the cloud resource selection to match optimized demand. We adopt XEN?s credit scheduler [8] to perform ther esource isolation among virtual machine pool on the same local machine. With XEN [8], we do dynamically separate the suitable resources (like CPU rate and network bandwidth) to fit to specific usage demands of different types cloud-tasks-set. We adopt XEN?s credit scheduler to perform the relocation selection among virtual pool on the same local machine. With XEN,we can dynamically separate the suitable resources (like rate of CPU and bandwidth of the network) to fit with the specific usage demands of various tasks.

4.2 Genetic Based Scheduling Algorithm

Schedulability can be expressed by

$$\forall_i : 1 \le i \le 2 : \frac{C_i}{D_i} + \frac{I_i}{D_i} \le 1 \qquad (5)$$

Where $I_1 = 0$

$$I_2 = \left\lceil \frac{D_2}{T_1} \right\rceil c_1 \qquad (6)$$

Schedulability can also be expressed by

$$\forall : 1 \le i \le\le 2 : \frac{C_i}{Di} + \frac{I_i}{Di} \le 1 \qquad (7)$$

Checking feasibility of all given process

$\forall: 1 \le i \le\le n: \frac{C_i}{Di} + \frac{I_i}{Di} \le 1$

Where $I_1 = 0$

$I_2 = \left[\left[\frac{D_2 - D_1}{T_1}\right] + 1\right] C_1 +$
$\left[\left[\frac{D_2}{T_1}\right] - \left(\left[\frac{D_2 - D_1}{T_1}\right] + 1\right)\right] xmin\left(C_1, C_2 - \left[\frac{D_2}{T_1}\right] T_1\right)$

$$I_i = \sum_{j-1}^{i-I} \left\lceil \frac{D_i}{T_j} \right\rceil c_j \qquad (8)$$

Equation (6) is like Equation (7), is sufficient but not necessary.

For an Example:
Consider the following process system

$\tau: C_1 = 2\ D_1 = 3\ T_1 = 5$

$\tau: C_2 = 2\ D_2 = 6\ T_1 = 5$

τ: $c_3 = 4\ D_3 = 10\ T_3 = 20$

The schedulability of the process system can be determined by Equation (7).
(a) check process τ_1

$\frac{C_1}{D_1} + \frac{I_1}{D_1} \le 1$

$\frac{2}{3} < 1$

Hence τ_2 is schedulable. (c) checking process τ_3

$\frac{C_3}{D_3} + \frac{I_3}{D_j} \le 1$

$\frac{4}{10} + \frac{I_3}{10} \le 1$

Where $I_3 = \left[\frac{D_3}{T_1}\right] c_1 + \left[\frac{D_3}{T_1}\right] C_2$

Where

$I_3 = \left[\frac{10}{5}\right] 2 + \left[\frac{10}{15}\right] 2 = 6$

Substituting $\frac{4}{10} + \frac{0}{10} = 1$

Hence τ_3 is schedulable. Consider D3=11, this should not affect the schedulability of the system (C)recheck process

$\frac{C_3}{D_3} + \frac{I_3}{D_3} \le 1$

$\frac{4}{11} + \frac{I_3}{11} \le 1$ Where

$$I_3 = \left\lceil \frac{D_3}{T_1} \right\rceil C_1 + \left\lceil \frac{D_j}{T_2} \right\rceil C_2$$

$$I_3 = \left\lceil \frac{11}{5} \right\rceil 2 + \left\lceil \frac{11}{15} \right\rceil 2 = 8$$

Substituting $\left\lceil \frac{4}{11} \right\rceil + \left\lceil \frac{8}{11} \right\rceil > 1$

Hence τ_3 is unschedulable by equation (7).

4.3 Experimental Studies

The algorithm starts by ordering the tasks in the task queue in non-decreasing order of their deadlines. The algorithm then selects a set of tasks from the sorted list and generates an initial population. Each chromosome in the initial population is generated by assigning each task in the task set to a randomly selected processor and inserting the pair (task, processor) in a randomly selected unoccupied locus of the chromosome. If the number of tasks is less than the chromosome size, then the first n loci of the chromosomes are used in solution encoding and active chromosome size is set to n. This ensures that all the genetic operators are applied to only the active part of the chromosome. The first loop determines the best schedule by applying genetic operators, crossover and mutation. The second loop terminates if the best chromosome has a fitness value in two subsequent generations or a maximum number of iterations have been completed. Once best schedule for a set of tasks has been found, the dispatch tasks in the queues. The general steps are

1.Randomly generate
2.Eliminate all deadlock whose SL is ¡SL of Machine
3.Schedule based on criticality
4.Schedule based on Laxity
5.Functionality ? Population
6.Generate for max no = number of jobs *2

For all calculation?. Finish Time = Σ ei .

4.3.1 Consideration

–Dynamic Updating - Security Level
–Update a table based on Profiling - Load Balance
–Statically measure transfer rate and store in a table - Transfer Rate (Distance)
–Check Criticality based on levels

4.4 Results and analysis.

Obtain the outputs of the GSA the experimental was done by using real hadoop private cloud on windows operating system with Latest configuration. Latest NetBeans IDE is used to run Cloud service lab setup scenario, the GSA is compared to the existing recourse allocation schemes for this reason selecting standard examples data is taken.We have created various types of VMs and jobs with different size. Our lab setup considers the two datacenters and 500 to 5000 jobs for the scenario. The arguments values on the cloud given in Table 1.

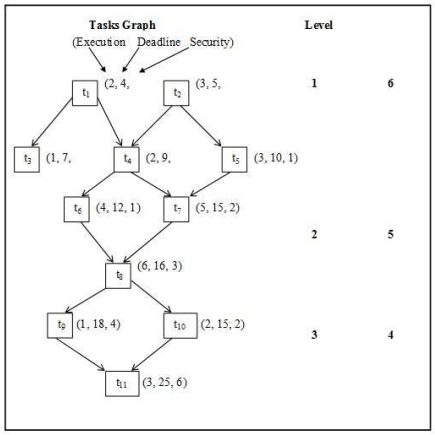

Fig. 2: Task graph

Table 1: Final Scheduling

Tasks	Laxity	Criticality	Security Level	Processes
t1	2	6	1	P1 P2 P3
t2	2	6	2	P2 P3
t3	6	5	3	P2 P3
t4	7	5	4	P2 P3
t5	7	5	1	P1 P2 P3
t6	8	4	1	P1 P2 P3
t7	10	4	2	P2 P3
t8	10	3	3	P2 P3
t9	17	2	4	P2 P3
t10	18	2	5	P3
t11	22	1	6	P3

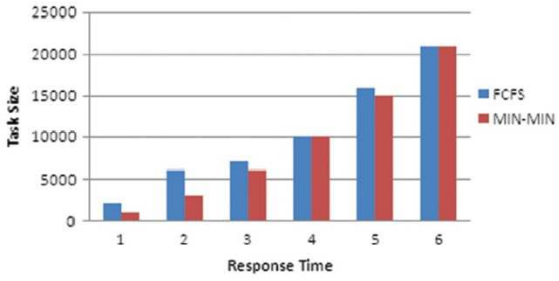

Fig. 3: Makespan for Average waiting time

4.5 Results Comparisons

Figure 3, shows the overall makespan for average waiting time of six cloudlets on two data centers in space shared

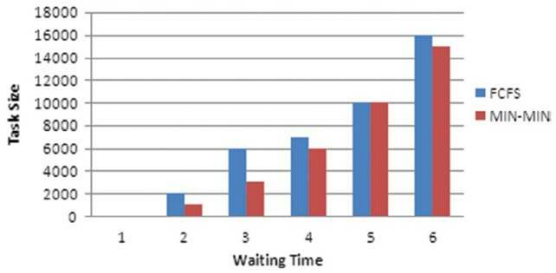

Fig. 4: Makespan for Average response time

```
========== OUTPUT ==========
Cloudlet ID    STATUS    Data center ID    VM ID    Time    Start Time    Finish Time
     0         SUCCESS         2             0        2        0.1          2.1
     1         SUCCESS         2             1        4        0.1          4.1
     3         SUCCESS         2             0        5        0.1          5.1
     2         SUCCESS         2             2        6        0.1          6.1
     4         SUCCESS         2             1        7        0.1          7.1
     5         SUCCESS         2             2        9        0.1          9.1
MIN_MIN finished!
BUILD SUCCESSFUL (total time: 0 seconds)
```

Fig. 5: Simulation Output

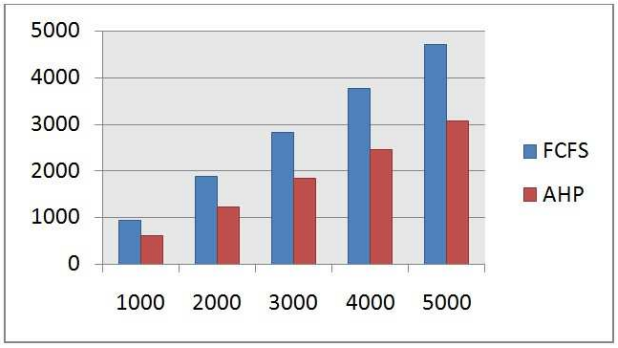

Fig. 6: AHP and FCFS with Space shared Resource allocation

manner where GSA resource allocation policy is used. It requires minimum waiting time than FCFS algorithm. Overall finishing time of tasks graph on GSA is less as compared to Min-Min, Min-Min allocation policy does not utilize the processing capabilities of data centers. In Figure 4, which uses average response time for resource allocation both the data centers are equally utilized, the load is balanced and minimized the overall makespan. So, in this scenario, GSA provides better performance than FCFS. Thus Simulation output is shown in Figure 5, which create the Data-center, Cloudlet with each Virtual machine is allocated to it. Thus start time and finish time is calculated using Min-Min algorithm with minimum completion time.

Figure 5 which create the own Datacentre, Cloudlet for tasks with each Virtual machine is allocated to it. Thus departure time and end time is measured by using GSA

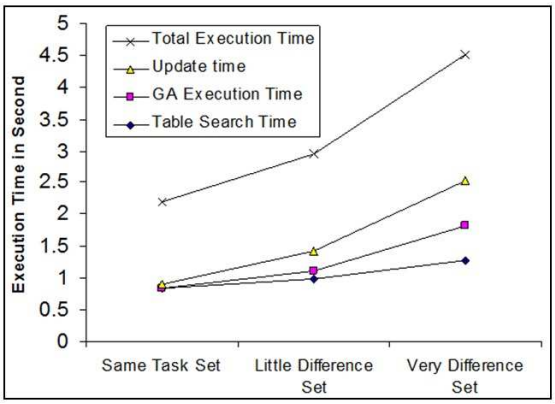

Fig. 7: Scheduling with different task set.

service gets the minimum completion time. An output is run for various scheme and the outputs are recorded. Similarly, the same set of story is also performed for the default FCFS scheduling algorithm class. Using these outputs, performance of AHP scheduler is analyzed.

5 Conclusion

Resource allocation technique is very much necessary to improve the data center of the cloud as well as VMs utilization, also to increases the entire cloud system performance. So our work is new resource provisioning technique or novel scheduling method is used to allocate the tasks in the cloud. This GSA algorithm is a somehow simple and efficient technique to allocate and schedule the tasks in cloud system. This allocation and scheduling method is more beneficial as compared to other existing mechanism.In the experimental result shows the comparison of different algorithms that have maximum resource utilization. Thus, the various results in our experiments in which, the information about the cloudsim parameters.

Extent our work further fast genetic based allocation can be done in real time scenario like hadoop environment using mapreduce programming skill set that further reduces the makespan and system utilization in cloud environment.

References

[1] A. Saraswathi, Y. Kalaashri, and S. Padmavathi, "Dynamic resource allocation scheme in cloud computing," *Procedia Computer Science*, vol. 47, pp. 30 – 36, 2015. Graph Algorithms, High Performance Implementations and Its Applications (ICGHIA 2014).

[2] J. O. Kephart, "Research challenges of autonomic computing," in *Proceedings of the 27th International Conference on Software Engineering*, ICSE '05, (New York, NY, USA), pp. 15–22, ACM, 2005.

[3] H. Izakian, A. Abraham, and V. Snel, "Metaheuristic based scheduling meta-tasks in distributed heterogeneous computing systems," *Sensors*, vol. 9, no. 7, pp. 5339–5350, 2009.
[4] Y. B. Ma, S. H. Jang, and J. S. Lee, "Ontology-based resource management for cloud computing," in *Proceedings of the Third International Conference on Intelligent Information and Database Systems - Volume Part II*, ACIIDS'11, (Berlin, Heidelberg), pp. 343–352, Springer-Verlag, 2011.
[5] S. Abrishami, M. Naghibzadeh, and D. H. Epema, "Deadline-constrained workflow scheduling algorithms for infrastructure as a service clouds," *Future Gener. Comput. Syst.*, vol. 29, pp. 158–169, Jan. 2013.
[6] L. Shakkeera, L. Tamilselvan, and M. Imran, "Improving resource utilization using qos based load balancing algorithm for multiple workflows in iaas cloud computing environment," *ICTACT Journal On Communication Technology*, vol. 4, no. 02, pp. 750–757, 2013.
[7] K. Dinesh, P. Scholar, G. Poornima, K. Kiruthika, and U. Scholar, "Efficient resources allocation for different jobs in cloud."
[8] V. Valtchanov and J. A. Brown, "Evolving dungeon crawler levels with relative placement," in *Proceedings of the Fifth International C* Conference on Computer Science and Software Engineering*, C3S2E '12, (New York, NY, USA), pp. 27–35, ACM, 2012.

A. Sandana Karuppan was born in Madurai, Tamilnadu, India, in 1980. He received the B.E. degree in Computer Science and Engineering from the Kamaraj University, India, in 2003, the M.E, in Computer Science & Engineering from Anna University in 2006 and Ph.D. degrees in Computer Science and Engineering pursuing from the Anna University, India. He is currently working as Assistant professor in the department of Information Technology of SSN College of Engineering, OMR, Kalavakkam, Tamilnadu, India. He has presented and published 5 technical papers in international and national conferences and 5 technical paper in international journal.

N. Bhalaji , Associate Professor of Information Technology has over 14 years of teaching experience. He received his B.E. & M.E. degree both in the discipline of Computer Science and Engineering and Ph.D. specializing in Trust Based Routing approach for MANET?s from Anna University, Chennai. His current research interests Application of Trust over information and communication domains namely Internet of Things and exploring Machine learning algorithms to improve quality of societal outlook. He is also a recognised supervisor of Anna university and guiding 2 full time and 8 part time scholars. He is also serving as a Doctoral Committee member for VIT, SRM and Satyabama University and as a Board of studies member in VIT Bhopal and Vels University, Chennai.

N. Ramaraj, joined as Professor of EEE of Vignan"s University, Andra Pradesh recently and guiding research scholars and M.Tech. students in the areas of Economic Scheduling, Load Dispatch, Optimal placement of PMUs, Power System Wheeling pertaining to EEE and Inter-operability, Big Data Analysis, data Mining, Component Technology and Networking pertaining to CSE.