Proceedings of the International Conference on Intelligent Computing and Communication for Smart World (I2C2SW-2018)
14-15 December 2018

Task Based Resource Scheduling in Cloud

I. Devi
Department of CSE
PSG College of Technology
Coimbatore- 641 004, India
deviilangovan@yahoo.com

G.R. Karpagam
Department of CSE
PSG College of Technology
Coimbatore- 641 004, India
grkarpagam@gmail.com

***Abstract*— Natural calamity is considered as one of the most significant research subject worldwide. Subsequently after disasters, the need for resilient communication rises and constantly fluctuates due to altering environmental situations and demand from public. At the present time, Cloud has been appeared as the most extensively used environment for providing services to the users. Cloud offers services which are available on pay per use model and on-demand. One of the important processes of such systems is Scheduling. The present work focuses on task based resource scheduling and used Precedence Scheduling Algorithm (PSA) to get non-stop cloud services.**

Keywords— disaster, virtual machine, scheduling, cloud.

I. INTRODUCTION

During the large-scale disasters, offering services to victims in the calamity site is particularly important. The inspiration for our research originates from such experiences of the natural calamities. In preceding years, there has been a considerable rise in the number of natural calamities such as floods, earthquakes, forest fire and tsunamis across the globe. Recent examples consist of Malaysia flash flood 2017, Heavy rains and floods were reported across India (Assam, Manipur, Arunachal Pradesh, Gujarat, Bihar, Uttarkhand, Uttar Pradesh and Orissa states) since early July (UN Office for the Coordination of Humanitarian Affairs). Though India is providing necessary measures to prevent loss of life by means of pre-disaster alert/ warning system, the entire communication network has been severely smashed up. Complete disconnection of network leads to severe problems for evacuation and rescue activities like transportation of food, medicines, and rescue materials. Disasters are spontaneous events which cause severe damage or loss of life. There exists a necessity to swiftly set up a new infrastructure which offers abrupt services by using existing network resources and work in a heterogeneous environment. As a result, a numerous standardization activities and research practices have been carried out recently to manage disaster resilient communication.

II. RESEARCH BACKGROUND AND OBJECTIVES

During disaster conditions, communicating networks were diminished and restoration of network was a puzzling issue. IoT offers communication in everyplace, everywhere and anytime. Under this circumstance, IoT offered information about resources which can be scheduled for distinct tasks. The resource scheduling problem was analyzed using bankers algorithm and attained the optimum usage of resources [1]. In addition, the obtained results were assessed with execution time and fairness of resource allocation for utility. A cloud scheduling algorithm was developed and it considers the existing VM resource utilization by examining prior VM utilization levels. The obtained results were compared with traditional schedulers available in Open Stack. The results showed that there exists decreased performance degradation by 19% and increased CPU utilization by 2% [2].A unique framework for discovering best VM placement strategy in data centre based on Round Robin (RR) and First Come First Serve (FCFS) algorithms was developed [3]. They observed better VM placement with increased resource utilization and declined response time, power consumption. This framework offered better cloud services to users. In order to virtualize a server into multiple servers, Virtualization was considered as the most extensively used technology which provides an operating environment for VM based cloud platform with improved efficiency. Greedy Particle Swarm Optimization (G &PSO) was proposed to resolve Task based scheduling problem. Greedy algorithm exhibited better performance parameters like rapid convergence rate, robust local and global search capabilities, and balanced workload on each virtual machine. Comparing with traditional particle swarm optimization algorithm, the G&PSO algorithm experienced improved virtual machine efficiency and resource utilization [4].

A. Objective

The objectives of the present work are presented as follows:

Handle on-demand request: To provide unremitting, certainly accessible cloud services using scheduling algorithms based on the user requisitions.

Assessment: Comparison with existing algorithms based on parameters namely waiting time, response time, make span**,** execution time and resource utilization.

III. SYSTEM MODEL

In the present work, the disaster management system is a combination of conventional network standards with cloud computing and IoT technology [5] are being designed and built. In addition, scheduling of accessible resources for distinct user requests during disaster has been recognized as the significant issue. In general, disaster management was divided into four phases namely mitigation, preparation, response and recovery. Figure 1 represents the phases of disaster management. The mitigation and preparation phases were considered afore the occurrence of disaster [1]. During mitigation phase, providing public awareness and education, protecting and improving the existing infrastructure and information drives were offered. In the preparation phase, volunteer supervision, civic alertness and disaster response strategy were performed. Instantaneously after the disaster, the response phase came into existence. During the response

phase, saving lives, providing basic resources (food, cloth, shelter, health facility and security) resource scheduling, resource provisioning was deployed. Recovery phase is the reestablishment of all facets of disaster's effect on a society and restoration to normal economy. In response phase, the resource scheduling phase was considered as the acute process in post disaster management. In order to handle the disaster situation for real time response, the different tasks need distinct resources. Thus, the resources need to be scheduled in a precise way to complete the tasks in a certain time frame. The dynamic requirements and continual changing environment in the course of disaster encourages to adopt a novel technology to make an effective and accurate decision in least time. Nowadays, cloud computing has become the most extensively used systems for offering virtual resources to the end users of internet. Cloud computing provides "Everything as a Service (XaaS)" [6]. The most vital process of such system was termed as task based resource scheduling by using Precedence Scheduling algorithm (PSA). In case of disaster, the tasks were considered as initiating the network, offering medicinal facilities to the victims, evacuation rescue and restoration processes and the like. With the aim of accomplishing the aforesaid tasks, resources like food, volunteers, fire services, hospital facilities, rescue teams were required. The resources were provisioned based on the precedence. The precedence was determined based on the level of user interest and affinity towards particular resource. In general, cloud comprises of service providers and consumers. In the present work, the following cloud computational resources were considered namely host CPU architecture (Intel/AMD), CPU cores and memory. Storage resources were classified based on its type, capacity and speed. Similarly, the tasks were classified based on their precedence, arrival time, start time, waiting time and dependency among the tasks. The cloud comprises of one or more datacenters which were controlled by single or multiple cloud service providers. In the present work, only one datacenter managed by one cloud service provider was considered to obtain the cloud environment.

A. *Cloud Service Provider*

The Cloud Service Providers (CSP) was the objects which deliver its resources (R) or services (S). The cloud (C) comprises a set of resources. The resources were:

N Computational Resources R_C= {R_{C1}, R_{C2}, R_{C3}… R_{CN}}

Where, $R_{Ci=}$ {c_i, s_i, me_i, pa_i} for all i=1, 2, 3….N

c_i : Number of CPU cores of resource i

s_i :speed of resource i

me_i: memory of resource i

pa_i: Processor architecture of resource i

P number of Storage resources R_S= {R_{S1}, R_{S2}, R_{S3}…. R_{SP}}

Where, $R_{Sk=}$ {si_k, spe_k, sty_k} for all k=1, 2, 3…P

si_k: Size of storage resource k

spe_k: Speed of storage resource k

sty_k :Type of storage resource k

S number of Network resources R_N= {R_{N1}, R_{N2}, R_{N3}….. R_{NS}}

Where, $R_{Nj=}$ {b_j} for all j=1, 2, 3….S

b_j : Bandwidth of network resource j.

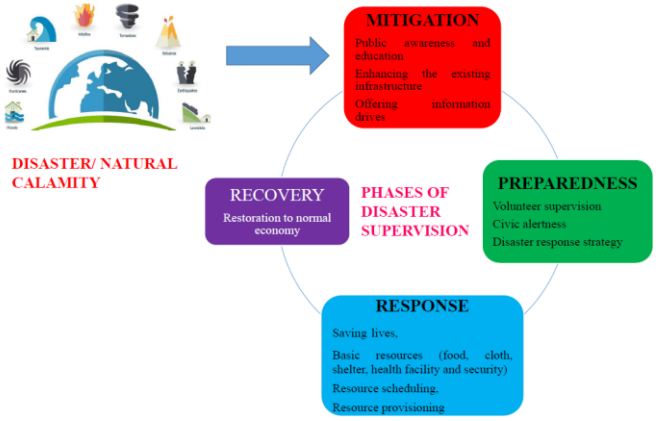

Fig. 1. Phases of disaster management

B. *Cloud Consumer*

The requests from the users can be depicted as a set of M tasks t_m= {t_1, t2, t3….t_M}. The distinct requirements of each tasks were showed as t_m= {R_C^{mC}, R_S^{mS}, R_N^{mN}, $t_{m,p}$}. Where,

R_C^{mC} = { $R_C^{mC,\,c}$, $R_C^{mC,\,s}$, $R_C^{mC,\,me}$, $R_C^{mC,\,pa}$ }

R_S^{mS}= {$R_S^{mS,si}$, $R_S^{mS,spe}$, $R_S^{mS,sty}$}

R_N^{mN}= {$R_N^{mN,b}$}

m^C = 1,2…….. M^C,is the user requesting compute resource.

m^S= 1,2…….. M^S,is the user requesting storage capacity.

m^N= 1,2…….. M^N,is the user requesting network capacity.

R_C^{mC} = required compute resource by task m^C

$R_C^{mC,\,c}$ = required number of CPU cores by task m^C

$R_C^{mC,\,s}$ = required speed by task m^C

$R_C^{mC,\,me}$ = required memory by task m^C

$R_C^{mC,\,pa}$ = required architecture by task m^C

R_S^{mS}= required storage resource by task m^S

$R_S^{mS,si}$ = required size of storage resource by task m^S

$R_S^{mS,spe}$ = required speed of storage resource by task m^S

$R_S^{mS,sty}$ = required type of storage resource by task m^S

R_N^{mN} = required network resource by task m^N

$R_N^{mN,b}$= required bandwidth of network resource by task m^N

$T_{m,\,at}$= arrival time of task m.

$T_{m,\,bt}$= burst time of task m.

$T_{m,\,pt}$=precedence of task m.

On the whole, available resource R_A= {R_C, R_S, R_N}. The required resource t_m={R_C^{mC}, R_S^{mS}, R_N^{mN}, $t_{m,p}$}. To describe the scheduling in the case of disasters, two matrices were used. The matrices were relevant to tasks and available resources. The method should find the requirements of jobs and substantiate if the required resources were available.

IV. PROPOSED PRECEDENCE SCHEDULING ALGORITHM

Cloud computing offers IT services as utilities, concerning the certain service model provided by the provider, they widely divided as Infrastructure as a Service (IaaS) where users rents VM, network and storage; Platform as a Service (PaaS) where users use the platforms and frameworks to host their application; Software as a Service (SaaS) where users are allowed to access the application hosted in remote systems rather than their own local systems. Table 1 shows the available services with their infrastructure requirements. Here, Urgent Bag of Tasks on cloud is bulk request at a time for acquiring the service.Table 2 shows the available resources in the cloud.

TABLE I. AVAILABLE SERVICE WITH THEIR INFRASTRUCTURE REQUIREMENTS

Services	Infrastructure
Rescue	16 CPU cores, Memory:56 GB, Local SSD:800 GB
Hospital	8 CPU cores, Memory:28 GB, Local SSD:400 GB
Food	4 CPU cores, Memory:16 GB, Local SSD:200 GB

TABLE II. AVAILABLE RESOURCES

INSTANCE	CPU (c_i)	RAM (me_i)	DISK (si_k)	ID
Miniscule	1	512 MB	1 GB	1
Minor	1	2048 MB	20 GB	2
Average	2	4096 MB	40 GB	3
Major	4	8192 MB	60 GB	4
Outsized	8	16384 MB	80 GB	5

A. Precedence scheduling algorithm

The proposed system architecture was implemented using Precedence Scheduling algorithm. The outcome obtained by using this algorithm is a composition of three important measures namely turn-around time and waiting time or response time. The process of obtaining these aforesaid criteria depends on the following constraints:

Arrival time (AT): time on which the task reaches in the queue.

Burst Time (BT): time need by a task for execution.

Completion Time (CT):time on which the task concludes its execution

Table 3 represents the description for computing the two major conditions using the aforementioned parameters to attain the optimal scheduling of resources.

TABLE III. PRECEDENCE SCHEDULING ALGORITHM CONSTRAINTS

S.NO	CONDITIONS	FORMULA
1.	Turn-around time	TT=CT-AT
2.	Waiting-time	WT =TT-BT

Algorithm: Precedence scheduling algorithm

Input: Service Request from the users

Output: {RCmC, RSmS, RNmN, tm,p}

Where,

RCmC = required compute resource by task mC

RSmS= required storage resource by task mS

RNmN = required network resource by task mN

1. Services are registered in a cloud by Cloud Service Providers (CSP).
2. Many users request many resources at a time.
3. To achieve scalability, cloud schedules the requests to Virtual machine in such a way to provide the proper resource provisioning. In order to determine the precedence, the following steps were carried out:

   3.1. The scheduling request was divided into a ladder of objective, measure, sub- measure and alternative.

   3.2. Creation of Pair-Wise evaluation matrices with all the measure and alternatives. The matrix was mathematically depicted as,

$$A=\begin{pmatrix} a11 & \cdots & a1n \\ \vdots & \ddots & \vdots \\ an1 & \cdots & ann \end{pmatrix}$$

   Where A= aij, aij>0 and 1/[aij]=aij[aij] where i, j=1,2,3,…….n

   If aij=1 then i=j;

   If aij=1/[aij] then i≠j.

   3.3. From the pair-wise comparison matrices, in order to decide the weights of the criteria and alternatives, each value in a column "j" was divided by the total number of the values in a column "j".

$$A_w=\begin{matrix} \frac{a11}{\Sigma ai1} & \frac{a12}{\Sigma ai2} & \cdots & \frac{a1n}{\Sigma ain} \\ \frac{a21}{\Sigma ai1} & \frac{a22}{\Sigma ai2} & \cdots & \frac{a2n}{\Sigma ain} \\ \frac{an1}{\Sigma ai1} & \frac{an2}{\Sigma ai2} & \cdots & \frac{ann}{\Sigma ain} \end{matrix}$$

   3.4. Eigenvector of matrix A was obtained by calculating the averaging the values in the row "i" of Aw matrix.

$$C=\begin{matrix} c1 \\ c2 \\ c3 \end{matrix}\begin{matrix} \frac{a11}{\Sigma ai1}+\frac{a11}{n} & \cdots & +\frac{a1n}{\Sigma ain}\frac{1}{n} \\ \cdots & \cdots & \cdots \\ \frac{an1}{\Sigma ai1}+\frac{an2}{n} & \cdots & +\frac{ann}{\Sigma ain}\frac{1}{n} \end{matrix}$$

The Eigen vector C is termed as a Precedence vectors.

4. From the obtained precedence, the tasks were scheduled in VM and resources are promptly provisioned.

Scheduling algorithm

Data: M tasks t_m= {t_1, t2, t3….t_M}.

Required: tm={R_C^{mC}, R_S^{mS}, R_N^{mN}, $t_{m,p}$}.

Sort M based on the priority value (p) of each job *j* and resource R;

For each *Job j: M* **do**

/* concentrate on high priority or emergency job first*/
Calculate Δcore =∑_(i=1)^k [δcore(R_C^{mC})]
//Available compute resource
Calculate Δsto =∑_(i=1)^k [δsto(R_S^{mS})]
//Available storage resource
Calculate Δnet =∑_(i=1)^k [δnet(R_N^{mN})]
//Available network resource
TS (V) =set of tasks in VM_i;
τ = |TS (V)|; /* total number of tasks in VM_i */
Total= $\sum_{i=1}^{\tau} \theta(t_M)$
For each task t: TS(V) do
ReqCompute= ceil((Δcore/Total)* θ(τ));
ReqStorage= ceil((Δsto/Total)* θ(τ));
ReqNetwork= ceil((Δnet/Total)* θ(τ));
Assign ReqCompute, ReqStorage and ReqNetwork resources to current task t;
End
end

V. EXPERIMENTAL RESULTS

In the present work, pi is the precedence of each user request or task. Tat= arrival time of task and Tbt= burst time of task. This subdivision focuses on the experimental results of the proposed work. Cloudsim is an open source simulating tool which has been extensively used in performing simulation of cloud applications and associated algorithms. In general, scheduling in cloudsim was performed in two distinct levels namely VM scheduling and Cloudlet scheduling. VM scheduling involves scheduling of Processing elements and Virtual Machine whereas Cloudlet scheduling includes scheduling between VM and cloudlets (applications to be executed over cloud). Cloudsim considers two types of scheduling procedures with both levels namely time shared scheduling and space shared scheduling[7]. Cloudsim was used for the implementation of the proposed Precedence Scheduling Algorithm (PSA) for better scheduling of resources.

A. Results and discussions

The implementation begins with default Cloudsim tool, where first VM accepts first task; next VM accepts the next task and so forth. From the obtained results, it was observed that the time required for successful build of cloudlets was 3 seconds. Table 4 shows the result obtained with default cloudsim.

Secondly, the experimentation was carried out by allowing the tasks to be consigned to any available VM (VMy or VMy+1) based on PSA algorithm. This type of suppleness decreases the running time to a great extent. Table 5 shows the results obtained with Precedence Scheduling Algorithm. From the table 6, it was observed that the time required for successful build of cloudlets was 2 seconds. On comparing with aforementioned results, PSA algorithm exhibited 25-30% more improvement ratio.

TABLE IV. ILLUSTRATION OF TOTAL EXECUTION TIME OF THE PROPOSED ALGORITHM COMPARED TO THE STATE OF THE ART ALGORITHMS

SNO	ALGORITHM	TOTAL EXECUTION TIME (ET)(SEC)
1.	Default cloudsim	3
2.	[8]	3.2
3.	[3]	2.5
4.	Proposed PSA	2

Fig. 2 depicts the comparison of execution time of the proposed algorithm compared to the state-of-the art works. The obtained results showed an enhancement ratio in the range of 55%-85% concerning to the execution time. The proposed algorithm exhibited a better performance than state of art works.

TABLE V. RESULTS WITH DEFAULT CLOUDSIM

Cloudlet ID	Status	Datacenter ID	VM ID	Time	Start time	Finish time
0	Success	2	0	800	0.1	800.1
2	Success	2	2	1200	0.1	800.1
1	Success	2	1	8000	0.1	800.1
3	Success	2	3	16000	0.1	800.1

TABLE VI. RESULTS PSA ALGORITHM ON CLOUDSIM

Cloudlet ID	Status	Datacenter ID	VMID	Time	Start time	Finish time
1	Success	2	1	1600	0.1	1600.1
0	Success	2	0	1600	0.1	1600.1
3	Success	2	3	1600	0.1	1600.1
2	Success	2	2	1600	0.1	1600.1

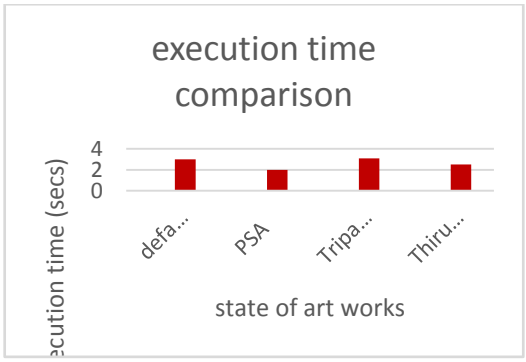

Fig. 2. Comparison of execution time of the proposed algorithm with the state-of-the art works

Make span is stated as the time needed to accomplish all the tasks. It the greatest completion time of tasks executed on cloud resources. It is the vital performance measure of scheduling of VM.

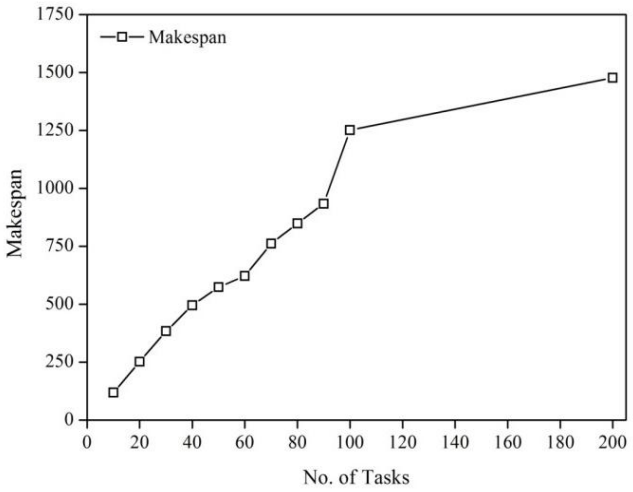

Fig. 3. Make span VS No of tasks

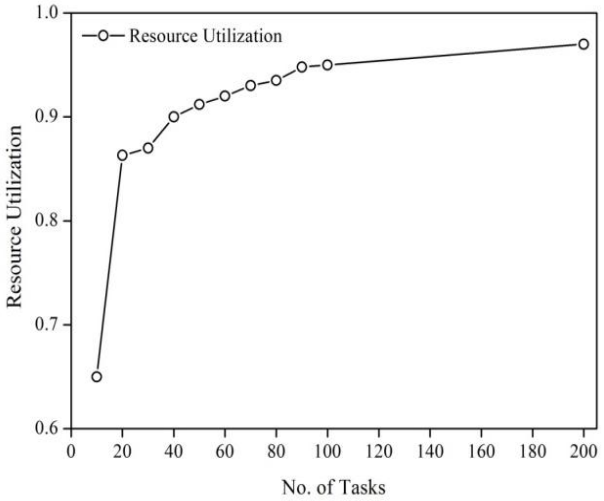

Fig. 4. Resource Utilization VS Number of Tasks

The following formula is used to calculate make span:

For a cloud service, if x number of resources are used, then make span is obtained as

Make span= max {T1, T2, T3...Tx}

As the number of tasks was varied from 10 to 200, make span obtained by using PSA algorithm were linearly raised. From the figure 3, PSA algorithm has executed better than state of art works. From the figure 4, it was noticed that the resource utilization of the proposed algorithm has increased up to 97 % whilst number of tasks increased. Thus, PSA algorithm was able to perform better in the course of many tasks.

VI. CONCLUSION

Efforts have been taken to realize disaster response system as a cyber physical system. The contributions of this paper include: handling on-demand request and offering cloud services to users. This paper proposed an algorithm using Precedence Scheduling for task based resource scheduling to offer accessible cloud services to the users based on their requisitions. The proposed algorithm was compared against the state-of the art method to assess its efficiency. The results exhibited that the proposed algorithm surpasses the state-of-the art works in total execution time the rate of 50%.

ACKNOWLEDGMENT

We thank PSG collect of Technology, Coimbatore for supporting through AICTE- TEQIP III scheme under Research Fellowship Scheme.

REFERENCES

[1] Kumar JS, Zaveri MA, Choksi M. Task Based Resource Scheduling in IoT Environment for Disaster Management. Procedia Comput Sci 2017;115:846–52. doi:10.1016/j.procs.2017.09.167.

[2] Sotiriadis S, Bessis N, Buyya R. Self managed virtual machine scheduling in Cloud systems. Inf Sci (Ny) 2018;433–434:1339–51. doi:10.1016/j.ins.2017.07.006.

[3] Thiruvenkadam T, Kamalakkannan P. Virtual Machine Placement and Load Rebalancing Algorithms in. Int J Eng Sci Res Technol 2016;5:346–59.

[4] Zhong Z, Chen K, Zhai X, Zhou S. Virtual machine-based task scheduling algorithm in a cloud computing environment. Tsinghua Sci Technol 2016;21:660–7. doi:10.1109/TST.2016.7787008.

[5] Ray PP, Mukherjee M, Shu L. Internet of Things for Disaster Management: State-of-the-Art and Prospects. IEEE Access 2017;5:18818–35. doi:10.1109/ACCESS.2017.2752174.

[6] Buyya R, Shin C, Venugopal S, Broberg J, Brandic I. Cloud computing and emerging IT platforms : Vision , hype , and reality for delivering computing as the 5th utility. Futur Gener Comput Syst 2009;25:599–616. doi:10.1016/j.future.2008.12.001.

[7] Himthani E Scholar PM, Saxena Asso A, Manoria M. Comparative Analysis of VM Scheduling Algorithms in Cloud Environment. Int J Comput Appl 2015;120:975–8887.

[8] Tripathy L, Patra RR. S i c c. Int J Cloud Comput Serv Archit 2014;4:21–7.