International Journal of Applied Engineering Research
ISSN 0973-4562 Volume 10, Number 12 (2015) pp. 31883-31898
© Research India Publications
http://www.ripublication.com

Application of Particle Swarm Intelligence for Load Balancing in Cloud Environment

L. Aruna and Dr. M. Aramudhan

Research Scholar, Periyar University, Salem, Tamil Nadu, India,
Associate Professor & Head, Department of IT, PKIET, Karaikal, Puducherry, India.
E-mail: aruna_pmu@yahoo.com; aranagai@yahoo.co.in

ABSTRACT

Live virtual machine (VM) migration may be a technique for achieving system load balancing in a very cloud setting by transferring an energetic VM from one physical host to a different. This technique has been planned to cut back the time period for migrating overloaded VMs, however it's still time- and cost-consuming, and an outsized quantity of memory is concerned in the migration method. To avoid these drawbacks, we propose a Process based Load balancing using Particle Swarm optimisation (PBLB-PSO) that achieves system load balancing by solely transferring further Process from Associate in reducing overloaded VM rather than migrating the whole overloaded VM. We have a tendency to conjointly style Associate in nursing optimization model to migrate these further Process to the new host VMs by applying Particle Swarm optimisation (PSO). To gauge the planned technique, we have a tendency to extend the cloud machine (Cloudsim) package and use PSO as its Process programming model. The simulation results show that the planned PBLB-PSO technique considerably reduces the time taken for the load balancing method compared to ancient load equalization approaches. moreover, in our planned approach the overloaded VMs won't be paused throughout the migration method, and there's no have to be compelled to use the VM pre-copy process. Therefore, the PBLB-PSO technique can eliminate VM time period and therefore the risk of losing the last activity performed by a client, and can increase the standard of Service practiced by cloud customers.

Keywords. Cloud computing, Particle swarm optimization, Virtual machine migration, Process scheduling, Cloudsim, Jswarm

INTRODUCTION

Cloud computing provides new business opportunities for each service suppliers and service requesters by suggests that of a platform and delivery model for delivering Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and software package as a Service (SaaS). A cloud encloses the IaaS, PaaS, and/or SaaS among its own virtualized resources, enabling it to hold out abstractions of its underlying resources. Typically, the virtualization of a service implies the aggregation of many proprietary processes collected in a virtual surroundings, known as Virtual Machine (VM). Clouds area unit typically adjoin a distributed virtualization infrastructure covering large geographical areas; as an example, the Amazon Elastic figure Cloud (Amazon EC2), Windows Azure (Microsoft's service cloud platform), and RESERVOIR (a European project adopting IaaS in cloud computing). Additionally, the prospect of cloud federation, wherever cloud suppliers use the virtualized infrastructures of different federate clouds, creates new eventualities within which several new services will be supplied exploitation the underlying resources of multiple cloud suppliers. In fact, clouds exploiting distributed virtualization infrastructures area unit able to offer new kinds of distributed IaaS, PaaS, and SaaS. A cloud computing platform exploitation virtualization technology achieves a dynamic load balance between servers, online VMmigration technology, it's attainable to on line re-map VMs and physical resources and dynamically succeed whole system load equalization. VM migration above all has been applied to versatile resource allocation or reallocation by moving death penalty VMs from one physical machine to another machine for stronger computation power, larger memory, quick communication capability, or energy savings. The fundamental disadvantage of exploitation ancient approaches for achieving system load balancing in a very cloud atmosphere is that the bulk decide to migrate overloaded VMs. This VM migration strategy has many drawbacks: (1) it prepares dirty memory that may accrue once pre-copy in on-line VM migration, (2) it utilizes a large amount of memory in each primary Physical Machine (PM) and new host PM, (3) it has to pause the first VM thereby inflicting VM downtime, (4) it carries the risk of losing recent client activities in on-line VMmigration, and (5) it's cost- and time-consuming. to beat these drawbacks in applying ancient approaches for achieving load balancing in cloud environments, we tend to projected a Process-based Load balancing (PBLB) abstract model in our previous work that achieves system load balancing by migrating Process from associate overloaded VM to a different uniform VM, rather than migrating the overloaded VM in its entireness. yet, the problem of finding associate optimum answer for allocating these further Process from associate overloaded VM to applicable host VMs remains open and desires to be solved . During this paper, we improve and complete our previous work to unravel this drawback, and propose the PBLB technique exploitation Particle Swarm optimisation (PBLB-PSO). we tend to additionally develop a multi-objective optimization model for migrating Process from overloaded VMs to minimize Process execution time and Process transfer time by applying the PSO formula. In addition, to guage our model, we have a tendency to extend the Cloudsim package and mix it with the Jswarm package to use our PSO-based Process planning model as Cloudsim's

Process planning formula. The contributions of our work to the present literature are often summarized as follows:

(1) Developing a PBLB-PSO methodology for system load balancing by migrating Process from Associate in Nursing overloaded VM to a different unvaried VM, rather than migrating the entire overloaded VM from a PM to a different PM.

(2) Developing a multi-objective Process planning optimisation model for migrating Process from overloaded VMs that minimizes each Process execution and transfer time, and developing a MOPSO based formula to unravel this planned optimization model.

(3) Extending Cloudsim and Jswarm packages, and validatory the operating for our method.

The rest of this paper is organized as follows. In Section 2, connected work for Process scheduling optimisation strategies is mentioned. In Section 3 we have a tendency to gift the abstract model and therefore the main formula of the PBLB- SO methodology, which can be increased by the developed optimisation model for optimum Process planning in Section 4, and the proposed formula for determination this Process planning model in Section 5. Our developed method is evaluated in Section 6. Finally, we have a tendency to gift our conclusion and future works in Section 7.

RELATED WORK

Virtualization technique has improved utilization and system load balancing by enabling VM migration, and has provided vital edges for cloud computing. Many ways are developed to migrate a running instance of a VM (a guest operative system) from one physical host to a different to optimize cloud utilization. Primary migration depends on method suspension and start. Several systems merely pause the VM and duplicate the state information, then resume the VM on the destination host. This stops the migrated application till all the memory states have been transferred to the migration destination, wherever it's resumed. These ways cause the applying to become unobtainable throughout the migration method. Osman et al. projected a system to migrate heritage and network application (ZAP) by providing a virtualization layer on prime of the package and transferring a process cluster. They win lower period of service, however still use stop-and-copy approach. to scale back the migration period and move the VM between hosts during a local space network while not disrupting it, VMotion and Xen utilize pre-copy migration technique to perform live migration and support seamless method transfer. In pre-copy migration technique, VMs migrate by pre-copying the generated run-time memory state files from the initial host to the migration destination host. If the speed for such dirty memory generation is high, it's going to take a protracted time to accomplish live migration, as a result of an oversized quantity of knowledge has to be transferred. In extreme cases when the dirty memory generation rate is quicker than the pre-copy speed, live migration will fail. Considering this reality, Jin et al. conferred a basic pre-copy model of VM live migration Associate in Nursingd projected an optimized algorithmic program to boost the performance of live migration by limiting the speed of memory modification by dominant the processor scheduler of the VM monitor.

Jun Associate in Nursingd Xiaowei designed an IPv6 live migration framework for VM primarily based on the IPv6 network surroundings. They designed a world management engine as a core complete IPv6 live migration for VM, Associate in Nursingd provided an IPv6 cloud computing service for the IPv4/IPv6 consumer. In their approach, the supply VM continues to supply services during the VM migration method and therefore the supply VM isn't stopped however it not provides new services till the completion of the previous service. The supply VM are going to be stopped then.

Liao et al. projected a runtime VMmapping approach to avoid wasting energy in an exceedingly cluster or information center. Their placement module centered on reducing power consumption. The main purpose of their approach is that the mapping of VMs onto atiny low set of PMs with minimum reduction in system performance. In their approach, they tried to show off the redundant nodes or PMs to avoid wasting energy, feat the active nodes to keep up system performance. They designed a probabilistic, heuristic algorithmic program that uses the simulated tempering optimisation approach to optimize VMs mapping onto a collection of PMs beneath the constraint of multi-dimensional resource consumption.

Nicolae et al. projected a zealous check purpose repository known as BlobCR. BlobCR takes live snapshots of the whole disk that's connected to the VM. BlobCR minimizes the performance overhead of check inform by applying VM disk snapshots asynchronously within the background by mistreatment selective copy-on-write technique. Their projected technique supports application-level and process-level check inform, as well as rollback filing system changes. Atif and Strazdins given a framework called ARRIVE-F to unravel the matter of heterogeneousness in virtualized reckon farms. They 1st divide the heterogeneous reckon farm into variety of homogenized sub-clusters. ARRIVE-F then performs a light-weight on-line identification of the electronic equipment, communication and memory subsystems of all the active jobs within the reckon farm and creates job execution time prediction model for every active job on the sub-clusters in the reckon farm. The framework relocates the reckon jobs to the acceptable hardware platforms supported the anticipated execution times. ARRIVE-F uses the live migration feature of Virtual Machine Monitors (VMMs) to transfer jobs from one sub-cluster to a different.

Lin et al. expressed the assumption that the majority of the projected strategies for ondemand resource management concentrate on optimizing physical resource allocation to their associated virtual resources and migrating VMs to realize system load balancing and maximize resource utilization. These strategies force the capital punishment cloud computing applications to be suspended owing to themandatory termination of the associated VMs. To overcome this disadvantage, they projected a threshold-based dynamic resource allocation scheme for cloud computing that dynamically allocates the VMs among the cloud computing applications supported their load changes. In their projected technique, they determined once migration ought to be done however they failed to specify the main points of however reallocation would occur.

A elementary disadvantage of most existing researches is that they think about complete VM migration to beat overload VM and accomplish system load balance. To improve on this side and complete the previous works, we tend to propose during this paper the PBLB-PSO method that transfers Process from overload VMs, rather than VM migration to achieve system load equalization. The projected approach not solely eliminates the suspension and start processes throughout VM migration however additionally omits pre-copy mechanism and manufacturing dirty memory in live VM migration.

PROPOSED ALGORITHM

In this section, we describe the essential idea of the PBLB technique and the projected PBLB-PSO abstract model and formula. The PBLB technique contains a abstract model and algorithm which designed to achieve whole system load balancing by migrating Process from full VMs. Both computing intensive and information intensive Process thought of during this technique. For information intensive Process, our programming optimisation model takes under consideration information measure as a variable to decrease the information movement, that decreases the transfer time. For computing intensive Process, our planning optimisation model takes under consideration the number of CPUs on host VMs to schedule the information on a high performance pc. In addition, within the projected PBLB methodology, idle PMs won't be chosen because the new PM hosts. this is often to decrease energy consumption, and thence the value of the operation of the information center, as a result of associate degree idle PM should be turned on if Process area unit transferred thereto, and this action can increase energy consumption and value. The PBLB methodology contains a flat solid on that all cloud schedulers WHO manage VMs on clouds share their data about VM features and death penalty Process. Fig.1 shows the methodology.

The criteria of Quality of Service (QoS) like SLA data, are noted on this flat solid. Additionally, the PBLB abstract model contains a Central Process Scheduler (CPS) to transfer Process from associate degree overloaded VM to a brand new similar VM by applying the data on the flat solid.

In the PBLB methodology, the time of Process migration from associate degree overloaded VM is calculated on the idea of the remaining work capability of a VM as follows:

$$VMremain = VMwork - VMpenalty \qquad (1)$$

where VMwork is that the VM work, and VMpenalty is that the variety of death penalty Process within the

VM . The VM are overloaded and arrival Process ought to be migrated to a different similar VM for execution when:

$$0 \leq VMremain \leq 1 \qquad (2)$$

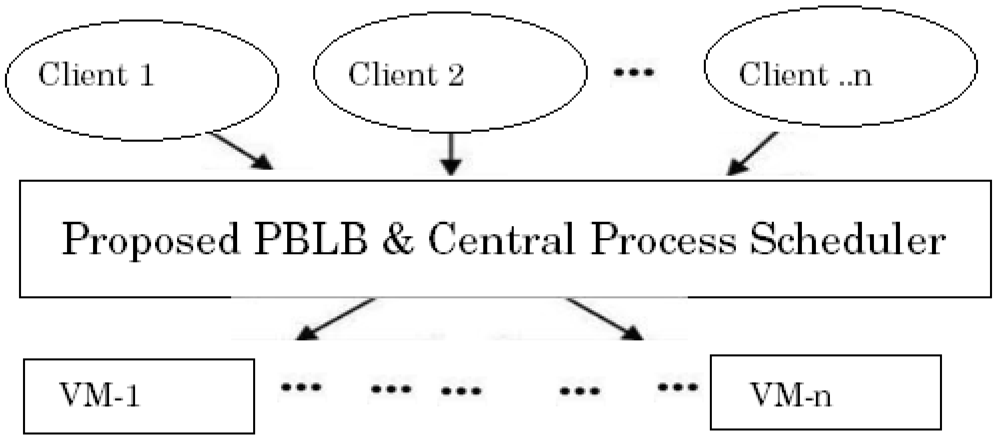

Fig. 1 PBLB-Methodology

In this paper, we improve the PBLB methodology and develop a technique for activity the process of CPS of the PBLB abstract model by applying the PSO formula. The CPS is answerable for finding applicable VMs as new hosts for the additional Process of the overloaded VMs. It then migrates these additional Process from the overloaded VM to the selected host VMs. The CPS ought to notice associate degree optimum thanks to portion additional Process to the new host VMs with less Process execution and Process transfer time that could be a multi objective optimization drawback. We propose an optimization model for migrating these additional Processes to the new homogenous VMs to attenuate Process execution and Process transfer time. This optimization model completes Steps three and four of the PBLB abstract model and creates a new PBLB-PSO methodology. To model this Process migration optimization problem, we tend to take into account information measure as a variable to attenuate the Process transfer time for data intensive applications. Also, we tend to take into account the properties of the new host VMs and PMs (memory, hard disk, variety of CPUs, etc.) to reinforce performance utilization for computing intensive applications. we tend to apply the knowledge of the PBLB flat solid as the computer file for the planned Process migration model to seek out AN acceptable host VM for the Process. Additionally, we tend to apply the PSO algorithmic program to resolve the planned optimization downside. Taking of these factors into consideration, the planned PBLB-PSO algorithm is delineated in keeping with the subsequent steps:

- [Begin PBLB-PSO algorithm]
- Step one:
  - Gathering information and knowledge concerning VMMs, VMs, PMs and SLA data from the world flat solid as inputs information for PBLB-PSO algorithmic program as follows:
    - VMs data includes: central processor (the range and speed of processors), free memory, the amount of capital punishment Process, Process execution time, Process performance model, Process' location, Process' needed resources (number of needed processors).

- PM criteria (total/current) includes: the amount and speed of processors (CPUs), free memory and disc, PM state of affairs (idle or active), its host VMM.
- SLA data.
- The objectives of the Process migration improvement model and their corresponding information as follows:
- Minimizing Process execution time: that is performance of Process' and VMs' properties.
- Minimizing Process transfer time: that is performance of VMs' information measure and volume of knowledge exchange between primary and host VMs.
- Step two :
  - observation knowledge and knowledge associated with the VMs' work flow state of affairs and determining VM employment data, full VMs, the Process that should be migrated from full VMs, and migration time.
- Step three :
  - Finding optimum solid VMs as new hosts for the Process of the full VMs (which is associate degree improvement problem) by CPS according to (1) the developed Process migration improvement model in Sect. 4, and (2) the proposed PSO-based algorithmic rule for finding this improvement model in Sect. 5.
- Step four :
  - Considering the optimum Process migration schema obtained, change the info related to the optimum Process execution time, optimum Process transfer time and current VM properties (executing Process, CPU, etc.) by CPS because the output of the PBLB-PSO algorithmic rule.
- Step five :
  - Transferring Process and their corresponding knowledge to the determined optimum host VMs.
- Step six :
- change the chalkboard and schedulers data in step with the outputs of Step 4.
- [End PBLB-PSO algorithm]

The details of Steps three and four of the PBLB-PSO are explained in the following.

OPTIMIZATION MODEL FOR PROCESS SCHEDULING IN PBLB-PSO

In this section, we describe our improvement model for Process programming victimization the PBLB-PSO technique. The best schema for migrating Process from overloaded VMs to the acceptable host VMs is achieved by finding this model. To work out Process execution/transfer time, we have a tendency to mix and improve the strategies planned in [22,23]. To minimize time consumption in our model, we have a tendency to not solely minimize the entire Process execution time, however conjointly minimize the most Process execution time. The variables used in our model area unit as follows:

Tset = {t1,t2,t3…n}= Set of Process that require to be migrated for load equalization VMj = Virtual Machine j, j = {1,2,3….n}

Bi j , i, j = {1,2,….}= The information measure between 2 nodes (virtual machines)

- m = the amount of VMs chosen as candidates to be the new hosts for Process in Tset
- n = the amount of Process
- yikz = one if Process i is assigned to VMz from VMk and yikz = zero, otherwise
- xik = one if Process i is assigned to VMk and xik = zero, otherwise
- DEik = the number of knowledge that the Process i assigns to VMk
- VMmk = the number of memory of VMk
- VMck = the amount of CUPs on VMk
- DTi j = the quantity of knowledge exchange between VMi and VMj

Applying these variables, we will outline and calculate the subsequent functions:

$$Texe_k = \sum_{i=1}^{n} x_{ik} * \frac{DE_{ik}}{VMm_k * VMc_k} \tag{3}$$

where Texek denotes the Process execution time on VMk . Supported this, the full Process execution time is calculated as

$$Texe = \sum_{k=1}^{m} Texe_k \tag{4}$$

The total Process transfer time is decided supported the quantity of changed information among VMs, and therefore the information measure capability between nodes, victimization the subsequent equation:

$$Ttrans = \sum_{i=1}^{n}\sum_{k=1}^{m}\sum_{z=1}^{m} y_{ikz} * \frac{DT_{kz}}{B_{kz}} \tag{5}$$

ALGORITHM FOR SOLVING THE PROPOSED PROCESS SCHEDULING PROBLEM USING PSO

In this section, we describe the preliminary definition and clarification of the PSO method. we describe the sub-PBLB-PSO formula that we've got developed to complete Steps three and four of the PBLB-PSO formula and to unravel the Process migration optimization downside.

Particle Swarm Optimization (PSO)

PSO shortly a standard international optimizer, mainly in issues in which the decision variables square measure real numbers. In PSO, particles square measure flown through hyper- dimensional search area. Changes to the position of the particles among the search area square measure supported the social–psychological tendency of people to emulate the success of different people. The position of every particle is modified in keeping with its own expertise which of its neighbors. Let X i (t) denote the position of particle

Table 1 Process Scheduling Pattern (Process Mapping)

Process	t1	t2	t3	t4	t5	… t*n*
VM number = Particle position	*vm*7	*vm*4	*vm*5	*vm*7	*vm*3	… *vmm*

i , at iteration t. The position of Xi (t) is modified by adding a speed Vi (t + 1) thereto,

i.e.: $X_i(t + 1) = X_i(t) + V_i(t + 1)$ (7)

The velocity vector reflects the socially changed data and, in general, is defined within the following way:

$$V_i(t + 1) = W V_i(t) + C_1 r_1(x_{pbest_i} - X_i(t)) + C_2 r_2(x_{gbest_i} - X_i(t)) \quad (8)$$

where C1 is that the psychological feature learning issue and represents the attraction that a particle has towards its own success; C2 is that the social learning issue and represents the attraction that a particle has towards the success of the whole swarm; W is that the inertia weight, which is utilized to regulate the impact of the previous history of velocities on the current speed of a given particle; x pbesti is that the personal best position of the particle i ; xgbesti is that the position of the most effective particle of the whole swarm; and r1, r2 ∈ [0, 1] are random values.

The Sub-PBLB-PSO Algorithm

This sub-algorithm determines the foremost applicable VMs to that the Process of the overloaded VMs are often allotted, and finds the optimum Process planning schema by applying the PSO algorithm adopted. This sub-algorithm applies the info and information determined in Steps one and a couple of of the PBLB-PSO algorithmic program as its inputs. In the Process scheduling model, there square measure n Process that ought to be appointed to m VMs to be dead. All particle positions X i = (x1, x2, . . . , xn) determined by PSO by applying Eqs. 7 and 8, square measure vectors with continuous values, however their corresponding distinct values square measure required to see the number of VMs chosen for Process. Therefore, we apply the little Position price rule to convert the particles' continuous position values vector to distinct vectors d(Xi) = (d1, d2, . . . , dn).

The sub-PBLB-PSO algorithmic program is delineate as follows and its flowchart is specified in Fig. 2:

[Begin PBLB-PSO algorithm-step 3]

- Step 3.1:
  Verify the full VMs by applying Eqs. 1 and 2.
- Step 3.2:
  verify the candidate host VMset by selecting the set of VMs as VMset = that satisfy following constraints:
- VM isn't full
- VM isn't settled on the idle PM
- VM has similar attributes to at least one of the full VMs
- VM has enough memory and therefore the needed range of idle CPUs to execute extra Process
- Step 3.3:
  verify the set of Process which require to be migrated from the full VMs because the migrating Process set
- Step 3.4:
  Apply the PSO methodology to resolve the projected improvement downside and assign the migrating Process (Test) to the host VMs to attenuate Process execution time and transfer time to attain this goal, the subsequent steps ought to be conducted:
- Step 3.4.1:
  produce AN initial population array of each particle i (Xi) with random positions and velocities on n dimensions within the search area.
- Step 3.4.2:
  Convert continuous position values vector of X i to separate vector d(Xi) using the SPV rule to work out the allocated VM for each arrival Process.
- Step 3.4.3:
  verify the worth of DEik ,VMmk ,VMck and Bik supported d(Xi) to calculate the worth of each fitness perform.
- Step 3.4.4
  for every particle, calculate Texek , Texe and T trans applying Eqs. 3, 4 and 5.
- Step 3.4.5:
  for every particle, calculate the fitness perform f (time) applying combining weight. 6.
- Step 3.4.6:
  for every particle, assess the specified improvement fitness perform.
- Step 3.4.7:
  Compare every particle's fitness analysis with its personal best fitness function worth (xpbesti). If the present worth is healthier than xpbesti , then set xpbesti adequate to the present worth, and therefore the best position pi adequate to the current location xi in n-dimensional area.
- Step 3.4.8:

establish the particle within the neighborhood with the most effective world success so far as xgbesti , and assign its index to the variable g because the best world position.

- Step 3.4.9:
  modification the speed and position of the particle in line with Eqs. 7 and 8.
- Step 3.4.10:
  If a criterion is met (usually a sufficiently smart fitness or a most number of iterations) then
- Step 3.4.10.1:
  Output best particle position in n-dimensional area d(X gbest) as the optimum Process migration schema
- Step 3.4.10.2:
  attend step 3.4.2
- Step 3.5:

o verify the subsequent criteria in line with optimum Process scheduling:

- New optimum Process execution/transfer times
- Current VM properties (CPU, . . .)
- Transfer Process and their corresponding knowledge to the host VMs

[End PBLB-PSO algorithm-step 3]

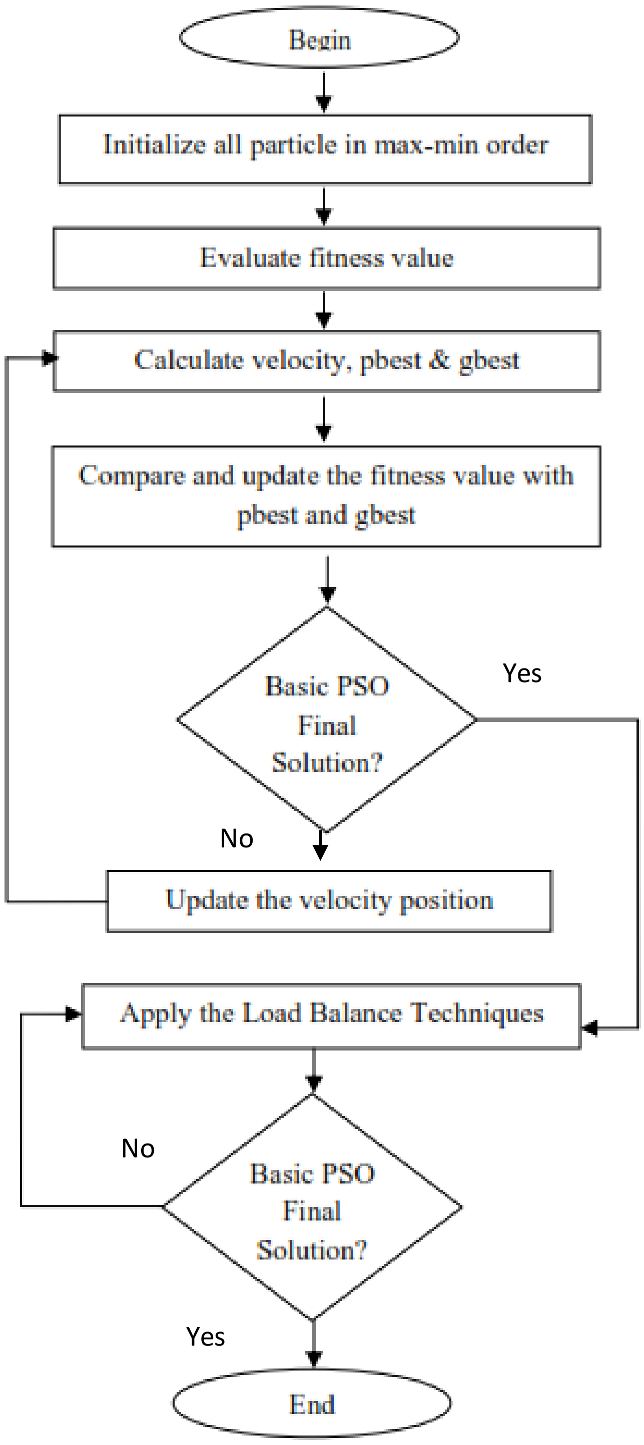
Begin
Initialize all particle in max-min order
Evaluate fitness value
Calculate velocity, pbest & gbest
Compare and update the fitness value with pbest and gbest
Basic PSO Final Solution?
Yes
No
Update the velocity position
Apply the Load Balance Techniques
No
Basic PSO Final Solution?
Yes
End

Fig. 2 Flow chart for Sub PBLB-PSO

EVALUATION

To evaluate our planned technique, we tend to extend Cloudsim by applying the Jswarm package. The "bind CloudletToVm()" technique within the "DatacenterBroker" category of Cloudsim is liable for distribution Process to VMs. The proposed methods only migrate tasks in the overloaded virtual machines to the hustle free virtual machines to balance the overloaded virtual machine, this will reduce the migration and processing time. Jswarm has the power to see the optimum Process arrangement among overloaded VMs in line with the PSO formula. The overloaded virtual machine is determined based on the optimal PSO algorithm. A Simulation environment with the following criteria: Four Virtual Machines and ten cloudlets.

Table 1. Cloudlet id and finishing time of various Load balancing algorithm

Cloudlets ID	FIFO		Ant Colony Optimization		Genetic algorithm		PB-LB algorithm	
	VM ID	Finishing time	VM ID	Finishing time	VM ID	Finishing time	VM ID	Finishing time
0	0	289.76	3	213.52	3	213.52	3	213.52
1	1	398.67	3	272.15	3	272.15	2	283.66
2	2	395.58	3	361.92	2	319.33	3	303.30
3	3	443.69	3	403.94	3	314.17	0	356.90
4	0	623.63	3	536.72	0	497.10	2	435.72
5	1	603.23	3	650.01	2	449.07	3	416.59
6	2	468.82	3	711.34	3	383.66	1	417.09
7	3	741.55	3	826.26	3	615.02	2	570.32
8	0	854.79	3	962.75	2	608.38	3	556.57
9	1	1007.58	3	1045.67	3	700.94	3	656.24

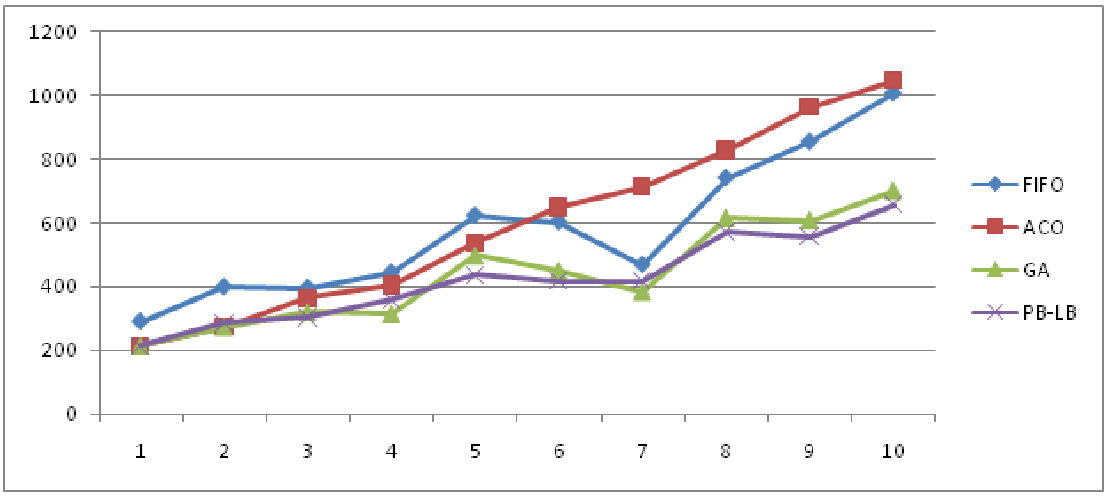

Fig 3. Finishing time graph for FIFO, ACO, GA and PBLB.

The simulation result produced by our method in the fig.3 shows reduced time for process migration among the overloaded virtual machine in the cloud environment over the FIFO algorithms and other evolutionary algorithms such as Ant Colony Optimization and Genetic Algorithm.

CONCLUSION AND FUTURE WORKS

VMmigration has been applied to realize system load balancing within the cloud surroundings by moving associate full VM from one PM to a different for stronger computation power, larger memory, quick communication capability, and/or energy savings. Though this approach reduces VM downtime and allows flexible resources allocation in an exceedingly cloud surroundings, the it's time- and cost-consuming. In addition, an outsized quantity of memory is concerned and there's the chance of losing the foremost recent activities of cloud customers throughout the VM migration method. To handle these shortcomings, we have a tendency to propose associated validate during this paper a brand new PBLB-PSO methodology to realize system load balancing within the cloud surroundings by migrating arrival tasks from an full VM to another uniform VM using PSO, rather than migrating the complete VM. This algorithmic rule contains a task migration optimisation model that minimizes task execution time and task transfer time. Additionally, we have a tendency to extend the Cloudsim package and mix it with the Jswarm package to use our PSO-based task programming model because the task programming algorithmic rule in Cloudsim and judge our projected model. The simulation results show that our projected PBLB-PSO methodology significantly reduces the time consumption of the load balancing method compared to alternative ancient strategies for load balancing.

The projected methodology additionally decreases energy consumption by avoiding the choice of idle PMs because the new host PMs. what is more, the projected methodology reduces period of time memory, memory usage and price consumption as a result of there's no got to pause VM throughput migration time, and far less idle capability within the host PM is needed. In our future work, we are going to extend PSO in Jswarm to multi-objective PSO and extend our optimisation model, to modify United States of America to contemplate a lot of aspects of task programming optimisation as new optimisation objectives.

REFERENCES

[1] Ramezani, F., Lu, J.,Hussain, F.: Tasks based system load balancing approach in cloud environment. In: International Conference on Intelligent Systems and Knowledge (ISKE), pp. 31–42 (2012)

[2] Nicolae, B., Cappello, F.: Blobcr: virtual disk based checkpoint-restart for HPC applications on IaaS clouds. J. Parallel Distrib. Comput. 73(5), 698–711 (2013)

[3] Calheiros, R.N., Ranjan, R., Beloglazov, A., De Rose, C.A.F., Buyya, R.: Cloudsim: A toolkit for modeling and simulation of cloud computing

environments and evaluation of resource provisioning algorithms. Softw. Pract. Experience 41(1), 23–50 (2011)

[4] Sapuntzakis, C.P., Chandra, R., Pfaff, B., Chow, J., Lam, M.S., Rosenblum, M.: Optimizing the migration of virtual computers. ACM SIGOPS Oper. Syst. Rev. 36(SI), 377–390 (2002)

[5] Whitaker, A., Cox, R.S., Shaw, M., Gribble, S.D.: Constructing services with interposable virtual hardware. In: 1st Symposium on Networked Systems Design and Implementation (NSDI), pp. 169–182 (2004)

[6] Jain, N., Menache, I., Naor, J., Shepherd, F.: Topology-aware VM migration in bandwidth oversubscribed datacenter networks. In: 39th International Colloquium, pp. 586-597 (2012)

[7] Nelson, M., Lim, B.H., Hutchins, G.: Fast transparent migration for virtual machines. In: USENIX Annual Technical Conference, pp. 391–394 (2005)

[8] Atif, M., Strazdins, P.: Adaptive parallel application resource remapping through the live migration of virtual machines. Futur. Gener. Comput. Syst. doi:10.1016/j.future.2013.06.028 (2013)

[9] Liu, H., Abraham, A., Snášel, V., McLoone, S.: Swarm scheduling approaches for work-flow appli- cations with security constraints in distributed data-intensive computing environments. Inf. Sci. 192, 228–243 (2012)

[10] Engelbrecht, A.P.: Fundamentals of Computational Swarm Intelligence. Wiley, Hoboken (2005)

[11] Engelbrecht, A.P.: Computational Intelligence: An introduction. Wiley, Hoboken (2007)

[12] Gao, Y., Zhang, G., Lu, J., Wee, H.-M.: Particle swarm optimization for bi-level pricing problems in supply chains. J. Glob. Optim. 51(2), 245–254 (2011)

[13] Buyya, R., Broberg, J., Goscinski, A. (eds.): Cloud Computing, Principles and Paradigms. Wiley, Hoboken (2011)

[14] Rochwerger, B., Breitgand, D., Epstein, A., Hadas, D., Loy, I., Nagin, K., Tordsson, J., Ragusa, C., Villari, M., Clayman, S.: Reservoir-when one cloud is not enough. Computer 44(3), 44–51 (2011)

[15] Ranjan, R., Buyya, R.: Decentralized overlay for federation of enterprise clouds. Technical report, The University of Melbourne (2008)

[16] Goiri, I., Guitart, J., Torres, J.: Characterizing cloud federation for enhancing providers' profit. In: IEEE 3rd International Conference on Cloud Computing (CLOUD), pp. 123–130 (2010)

[17] Clark, C., Fraser, K., Hand, S., Jacob, G.H.: Live migration of virtual machines. In: 2nd ACM/USENIX Symposium on Network Systems, Design and Implementation (NSDI), pp. 273–286 (2005)

[18] Jun, C., Xiaowei, C.: IPv6 virtual machine live migration framework for cloud computing. Energy Procedia 13, 5753–5757 (2011)

[19] Jin, H., Gao, W., Wu, S., Shi, X., Wu, X., Zhou, F.: Optimizing the live migration of virtual machine by CPU scheduling. J. Netw. Comput. Appl. 34(4), 1088–1096 (2011)

[20] Islam, S., Keung, J., Lee, K., Liu, A.: Empirical prediction models for adaptive resource provisioning in the cloud. Futur. Gener. Comput. Syst. 28, 155–162 (2012)

[21] Lin, W., Wang, J.Z., Liang, C., Qi, D.: A threshold-based dynamic resource allocation scheme for cloud computing. Procedia Eng. 23, 695–703 (2011)

[22] Zhao, C., Zhang, S., Liu, Q., Xie, J., Hu, J.: Independent tasks scheduling based on genetic algorithm in cloud computing. In: 5th International Conference on Wireless Communications, Networking and Mobile Computing, pp. 1–4 (2009)

[23] Juhnke, E., Dörnemann, T., Böck, D., Freisleben, B.: Multi-objective scheduling of BPEL workflows in geographically distributed clouds. In: 4th IEEE International Conference on Cloud Computing, pp. 412–419 (2011)

[24] Li, J., Qiu, M., Ming, Z., Quan, G., Qin, X., Gu, Z.: Online optimization for scheduling preemptable tasks on IaaS cloud systems. J. Parallel Distrib. Comput. 72(5), 666–677 (2012)

[25] Song, B., Hassan, M.M., Huh, E.: A novel heuristic-based task selection and allocation framework in dynamic collaborative cloud service platform. In: 2nd IEEE International Conference on Cloud Computing Technology and Science (CloudCom), pp. 360–367 (2010)

[26] Li, J., Peng, J., Cao, X., Li, H.-Y.: A task scheduling algorithm based on improved ant colony optimization in cloud computing environment. Energy Procedia 13, 6833–6840 (2011)

[27] Kolodziej, J., Xhafa, F.: Modern approaches to modeling user requirements on resource and task allocation in hierarchical computational grids. Int. J. Appl. Math. Comput. Sci. 21(2), 243–257 (2011)

[28] Lei, Z., Yuehui, C., Runyuan, S., Shan, J., Bo, Y.: A task scheduling algorithm based on PSO for grid computing. Int. J. Comput. Intell. Res. 4(1), 37–43 (2008)

[29] Taheri, J., Choon Lee, Y., Zomaya, A.Y., Siegel, H.J.: A bee colony based optimization approach for simultaneous job scheduling and data replication in grid environments. Comput. Oper. Res. 40(6), 1564–1578 (2013)

[30] Liao, X., Jin, H., Liu, H.: Towards a green cluster through dynamic remapping of virtual machines. Futur. Gener. Comput. Syst. 28(2), 469–477 (2012).