International Journal of Emerging Trends & Technology in Computer Science (IJETTCS)
Web Site: www.ijettcs.org Email: editor@ijettcs.org
Volume 5, Issue 6, November - December 2016 **ISSN 2278-6856**

OPTIMIZED ANT COLONY SYSTEM (OACS) FOR EFFECTIVE LOAD BALANCING IN CLOUD COMPUTING

Rajasekhar Bandapalle Mulinti[1], Prof G.A.Ramachandra [2]

[1]Sri Krishnadevaraya University, Anantapuramu, Andhra Pradesh.

[2]Sri Krishnadevaraya University, Anantapuramu, Andhra Pradesh.

Abstract

In recent decade cloud computing is play major role in the Information Communication. Most of the application deployment is in large scale data centers. In deployment of applications, storage, network to optimization the number of task and make them as balanced load into the servers. We proposed a new architecture for identification or estimation of temporal load and calculate & allocate resources to user requests. In this paper, workload allocation problem identified and optimized workload of servers to save the energy. We proposed Optimized Ant Colony System (OACS) to solve load balance by estimation in heterogeneous servers in cloud computing. The results shows better than the existing system of load balance in the cloud computing environment.

Keywords: Cloud Computing, Resource allocation, Optimized Ant Colony System.

1. INTRODUCTION

Cloud computing is its utility-oriented approach and focuses on delivering services with a given pricing model, in many cases a "pay-per-use" strategy. It is possible to access online storage, rent virtual hardware, or use development platforms and pay only for their effective usage, with no or minimal up-front costs. All these operations can be performed and billed simply by entering the credit card details and accessing the exposed services through a Web browser. It provides three services delivered in the cloud computing. The fundamental problem is multidimensional resources such as CPU, storage, networking, etc. with load balance, energy efficiency. In order to achieve objectives of high performance, energy saving, and reduced costs, so data centers need to handle the physical and virtual resources in dynamic environment. The study of research is to identify best optimization techniques that will facilitate efficient management and scheduling of computing resources in cloud data centers supporting scientific, industrial, business, and consumer applications. To identify the problem of energy usage, it necessary to eliminate inefficiencies and waste in the way of electricity is delivered to computing resource and also how to use these resources are maximum utilized to server application workloads. We can avoid by improving both the physical infrastructure and virtual machine maintains of data centers, and the efficiency of resource allocation algorithms. In recent years, data centers are design to increase the efficiency of infrastructure, example Facebook, datacenter is achieved a Power Usage Effectiveness of 90 % of data center's energy consumption is saved by using the computing resources.

2. PROBLEM DEFINITION

Based on the different types of applications served by the cloud computing center, there is a vast diversity in the demand resource profiles. In general, computing tasks such as web serving are more process intensive, while database operations typically require high-memory support. One of the other essential characteristics of a cloud computing system is diversification of server resources as well as the types of workloads. The DCs update the configuration of their resources, the processing capabilities, and memory and storage spaces to construct new platforms based on the new high-performance servers while the older servers are still operational. Due to heterogeneity of both servers and workloads, designing an optimal resource allocation algorithm concerning load balance, energy and cost efficiency becomes very complicated.

The problem of VM placement and migration for power minimization, which is NP-hard will be solved according to the estimation and available resources. Depending on the time limit and complexity constraints, three methods of offline generation, estimation of load are also proposed for initiation, limiting the searching area, and optimization termination, respectively. These methods are added to mitigate the complexity order of the optimization problem further. The main contributions of this paper are as follows:

- Heterogeneous resources and workloads of a DC are modeled and power efficient network-aware resource allocation platform is proposed to optimize the power consumption of cloud data centers.

- Optimization of Ant Colony System is proposed to estimate the incoming workload and prediction error is also considered in the optimal resource allocation.

- Dynamic job scheduler with optimization of cloud power consumption. To offline pattern initiation, cut and solve method and call back approach are proposed to reduce the complexity and search space and to make it scalable regarding the scheduling deadline.

3. PROPOSED WORK

In these work, the scheduling tasks are allocated using the ant colony algorithm in cloud computing and these studies have been divided into three categories. We focus on scheduling task and male them load balances efficiency schedules tasks combined with the ant colony and bee colony algorithms. To optimizing the total execution time of tasks reducing the execution time by combining a process and an ant colony optimization algorithm using the approximation method. That work sets a pheromone threshold to avoid premature convergence and avoids the local optimum through the two-tier search strategy and by using the pre-execution time. The system performance, such as load balancing. For example, the work proposes a scheduling method based on the ant colony algorithm. To minimizes makespan and also achieves load balancing by minimizing the idle time of virtual machines in order to balance the load of the virtual machines. To improve ant colony algorithm based on the shortest delay time of tasks. The task to a virtual machine as the searching object of the ants, taking into account the shortest completion time and load balancing. The proposes a scheduling algorithm based on QoS constraints combined with GA and integration, taking into account both time and cost.

4. DEFINITION OF TASKS AND RESOURCES

We assumed that there are *K* tasks *T* D f*T*1; *T*2; : : : ; *Ti*; : : : ; *TK* g and *N* resources *R* D f*R*1; *R*2; : : : ; *Rj*; : : : ; *RN* g in the current system of cloud computing. Here, cloud resources refer to the virtual resources.

Definition 1 (Tasks): *Ti* D (*Ci*;*Mi*;*Di*; *Li*). The first two parameters are CPU usage and memory usage, which the user has applied. *Di* is the deadline of the task, and *Bi* is the budget cost of the user. These parameters come from the task manager and are submitted by users.

Definition 2 (Resources): Each virtual resource cloud is defined by the main parameters of its CPU and memory, *Rj* D (*Cj*;*Mj*). These two parameters are representative of CPU utilization and memory usage.

Assumption 1: In order for the research to proceed, it is necessary to provide an assumption related to the above definitions. We have assumed that the information submitted by the user is trusted and the information of resource demand, submitted by the user, is accurate. In cloud computing, virtualization technology can monitor resource usage. If the user-accessed resources, such as CPU and memory, exceed the number of users applying during optimization Ant Colony System.

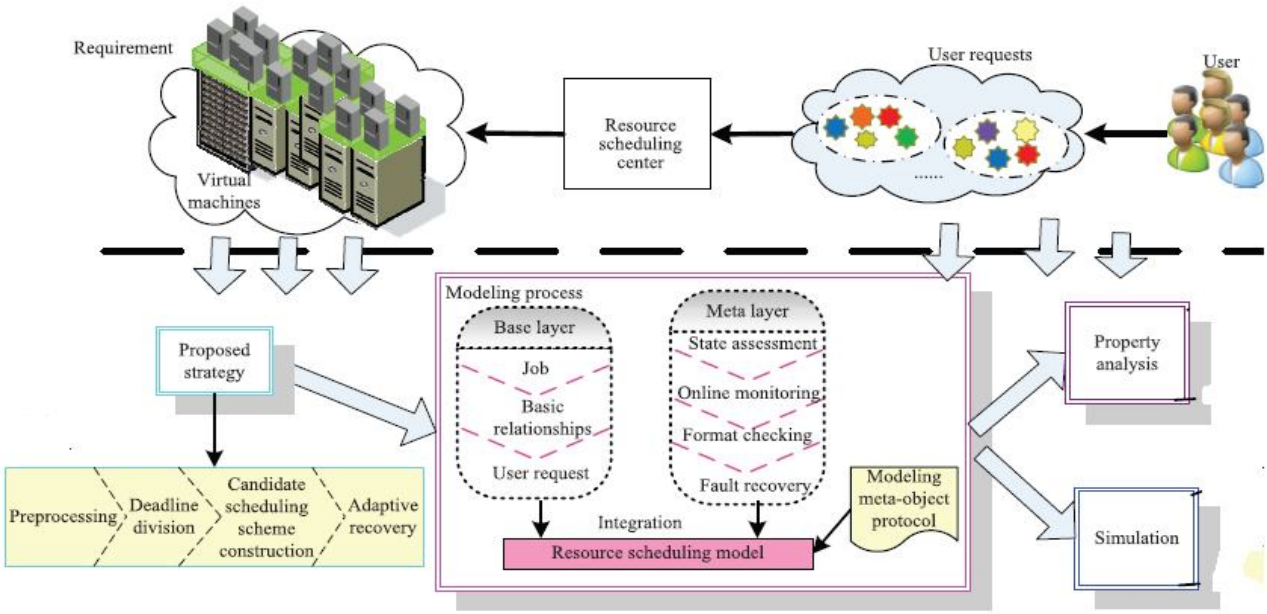
Ant Colony System: Dorigo and Gambardella proposed Ant colony system initially for solving the traveling salesman problem (TSP). Real ants are capable of finding the shortest path between the nest and the food source according to information passed via pheromone. In solving a TSP, ants construct a solution by visiting the cities one by one until all are visited. During the construction process, the next city to visit is selected according to the pheromone and heuristic information. Similarly, a Virtual Machine Allocation (VMA) can also use such a step-by-step construction assigning the VMs one by one to suitable servers. Thus, ACS is applicable to VMA problems.

Implementation of Optimization Ant Colony System (OACS)

To minimize the number of active servers and then reduce the energy consumption for cloud computing, we develop an OACS-based approach here, with order exchange and migration operations. The resultant existing algorithm searches for a solution with minimal number of servers to host all the VMs. We denote this minimal number as *M*min in the feasible globally best solution *Sgb*. Since *M*min is our optimization objective which is unknown in advance, we begin with *M*min=*N* in the initialization state, where *N* is the number of VMs. The initial feasible globally best solution *Sgb* is to place the *N* VMs on *N* servers with one VM mapping to one server. For each pair VM *k* and VM *j* ($j \neq k$), we introduce a pheromone value $\tau(k, j)$ and initialize it as τ 0= (*N*)–1. The pheromone value indicates the preference of two VMs to be assigned to the same server according to the historical experience.

The initialization, OACS goes to construct solutions iteration by iteration so as to find better feasible solutions with fewer servers. In each iteration *t* ($t \geq 1$), OACS aims to find a feasible solution with one server less than *M*min. Therefore, the ant tries to place the *N* VMs to the *Mt*=*M*min–1 servers. In each iteration, multiple ants construct their own solutions (assignment) with the

guidance of the construction rule. Each ant maintains a construction process by choosing vertices from a construction scheme. Fig. 1(a) shows an example of the construction scheme with N=5 VMs and Mt=4 available servers. It can be observed that the vertices of the construction scheme are arranged into an $Mt \times N$ matrix. Each vertex xij denotes a VM assignment to a server. The undirected arc between two vertices in the adjacent columns indicates the potential route of ants. Let consider a solution S = {$s1$, $s2$, $s3$, $s4$}, where si indicates a set of VMs assigned to server i, is constructed by the ant according to the path ($x11$, $x42$, $x23$, $x34$, $x15$) This solution shows that all the 4 servers are active to host the 5 VMs, with the consolidation as $s1$ = {1, 5}, $s2$ = {3}, $s3$ = {4}, and $s4$ = {2}. Each ant adopts similar process to construct a solution according to the construction scheme. The VMs are randomly shuffled before each construction process. Then the ant constructs a solution by assigning VMs one by one to the servers. VMs are not in particular order for solution construction in the evolutionary process. We describe the solution construction process based on one ant in the follows. It is based on a partial solution under construction. Consider totally N steps for an ant to construct a solution, with each step to select a proper server for the corresponding VM. In the lth ($1 \le l \le N$) step for placing VM j, a set of available servers Ij is whose element i stands for that server i has enough remaining resources (for both CPU and memory) to host the unassigned VM j. Then the ant uses a state transition rule to select a proper server i from the set Ij. For the pheromone, OACS deposits the pheromone between VMs rather than between a VM and a server. So we design a method to translate the pheromone between VM pairs into the preference between VM and server. The preference between VM j and server i represents the historical experience of packing the VM j together with those VMs (the set of si) that have already been placed on server i.



Load Balancing

In the interval time T, each compute node i=1 |Node| with time and current number of active nodes as taken actions based on the performance of the servers.

- If load is max, the compute node will try to migrate as many VMs as needed in order to reduce its power consumption below the maximum threshold.
- If load min, the compute node will try to migrate all its hosted VMs in order to switch off.
- If load is maintained equal, the compute node will maintain all its hosted VMs until the end of the next period.

Migration

If a compute node has exceeded its maximum power consumption threshold, it has become over utilized and is likely to violate the SLAs of the VM instances that is currently hosting. To mitigate, the over utilized compute node will start a partial VM migration process in order to shift part of its current workload to one or more nodes in the hypercube, whose current status is ok, idle, underutilized, or switched-off, so as to eventually reduce its power consumption to an acceptable level.

We assume that the data center has just started its operation, and thus each node is only aware of its immediate neighbors within the hypercube.

The distributed load balancing scheme is applied is divided in three phases that take place simultaneously and pertain to the VM migration strategies of compute nodes c1, c5, and c6, as follows:

Phase 1: At the end of period T, compute node c1 is hosting 20 VM instances, has a power consumption of 260 W, and is therefore considered over utilized. To remedy this, c1 will attempt to shift part of its workload to one or more of the not overloaded nodes, namely c2, c3, c5, and c6, until its power consumption drops below the acceptable threshold pmax ¼ 250 W. After retrieving their current state from its local cache, Nh, c1 will start with the most utilized ones, and will thus first try to migrate VM instances to its ok neighbor c3, which currently has a power consumption of 240 W. Taking the first VM instance out of its workload, c1 assesses the condition as expressed in Equation (7), and finds out that it can perform the migration. Continuing with the next VM instance, c1 reassesses the condition, but this time it turns out that c3 cannot accept any more workload without moving to the over utilized state. Hence, c1 picks the next neighbor, c2, and repeats the same process. As the current power consumption of c2 is 200 W, c1 is allowed to shift vm2 to it along with the next VM instance, vm3. At this point, the power consumption of c1 has dropped to 245 W, and its state has become ok. Hence, the partial M migration process successfully terminates.

Phase 2 : To compute node c5, which is underutilized as it is currently hosting two VM and is consuming 170 W. According to our load balancing approach, c5 will try to migrate both VM instances and switch off. Like already described, c5 first retrieves from its local cache the current state and power consumption of the other nodes. Since all nodes are synchronized, at this time the state of c1 is still over utilized, c3 and c2 are ok, while the state of neighbor c6 is underutilized shows, there are no other active nodes. Hence, c5 will migrate its two VM instances to c2 through c6 and switch off. The power consumption of c2 will be increased by 20 W.

Phase 3. The activities of phases 1 and 2 take place, compute node c6, which is also underutilized and currently hosting three VM instances, will try to shift all its workload to other non-overloaded nodes in order to switch off. At this point, c6 knows of two ok nodes: c2 and c3. Starting with the closest one, c6 will eventually manage to migrate all three VM instances to c2, eventually raising its power consumption to 225 W. Now, while migrating its VM instances, c2 also receives the two VM instances from c5, which in the meantime has switched off. Since c2 is still in ok state and is able to accept more workload without becoming over utilized, c6 will also forward those last two VM instances and will finally shut down

Let that threshold value be T(v). Now for every node, Ni, we need to find the load of each node. Let the total sum of loads be SL. To calculate the average of loads present in all nodes and introduce another factor called Cold Spot (α). Based on average value and cold spot we determine the threshold value. So nodes with loads greater than the threshold value will be considered as Cold spot. Step by step algorithm is shown below.

For each node Ni find load of Ni, i.e. Find L(Ni);

- Set SL=0, α be cold spot factor with value 1.2;
- For each node Ni do the following;
- SL=SL+L(Ni);
- Average (avg)=SL/Ni;
- T(v)=α * avg; end for;
- For each node Ni repeat up to step 9;
- If L(Ni) > T(v) then proceed to next step;
- Mark the node as hot spot node; end for;

Let HN be the list of hot spot nodes. A node or VM is not considered for migration if it is already a migrated one. Here in this algorithm we setup a three threshold values such as threshold-input-length (TIL), min-threshold-input-length (MTIL) and threshold time T. For every node or VM in HN we consider both the total balance input and processed input percentage. We select those nodes or VM with balance input greater than the TIL value and processed input lengths less than MTIL for migration

For each node Ni find load of Ni, i.e. Find L(Ni);

- Set SL=0, α be hot spot factor with value 1.2;
- For each node Ni do the following;
- SL=SL+L(Ni);
- Average (avg)=SL/Ni;
- HT=α * avg; end for;
- For each node Ni repeat up to step 9;
- If L(Ni) > HT then proceed to next step;
- Mark the node as hot spot node; end for;

RESULTS

We used CloudSim and cloud analyst in modeling and simulating Cloud computing environments. To evaluate the overhead in building a simulated Cloud computing environment that consists of a single data center, a broker and a user and performed series of experiments. The number of hosts in the data center in each experiment was varied from 100 to 1000. As the goal of these tests were to evaluate the computing power requirement to instantiate the Cloud simulation infrastructure and monitoring user workload.

Table 1: Average response time for different UBs in load balancing

Particle Size	OACS	MPSO	PSO
UB1(10)	49.05	49.85	50.64
UB2(20)	49.95	50.42	51.15
UB3(50)	50.20	50.92	51.83
UB4(100)	50.75	51.21	52.49
UB5(250)	190.65	195.25	200.9
UB6(300)	293.4	293.5	301.56

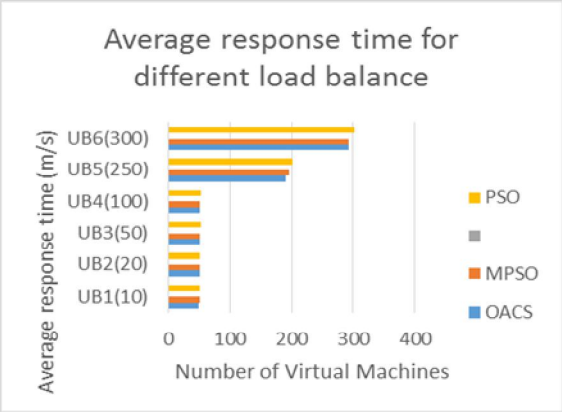


Table 2: Average processing time of DCs in load balancing

Data Centre with No. of VMs	OACS	MPSO	PSO
DC1(25)	0.621	0.689	0.786
DC2(50)	1.235	1.356	1.574
DC3(75)	1.801	1.938	2.1
DC4(100)	2.200	2.385	2.773

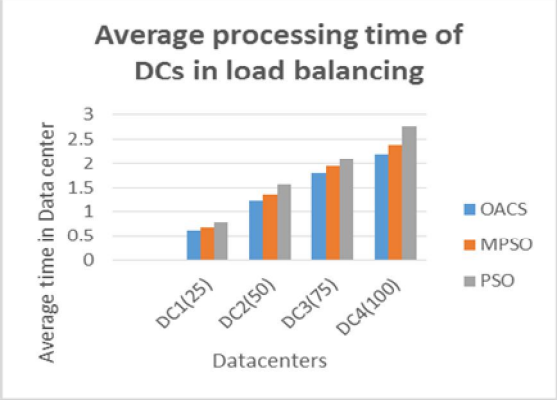
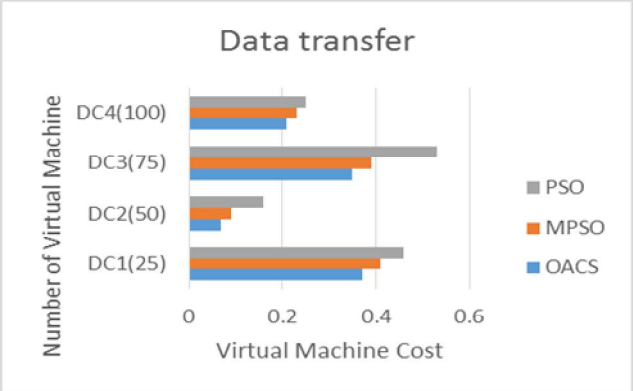


Table 3 : Data transfer cost of different data centers in LB

Data Centre with No. of VMs	OACS	MPSO	PSO
DC1(25)	0.37	0.41	0.46
DC2(50)	0.07	0.09	0.16
DC3(75)	0.35	0.39	0.53
DC4(100)	0.21	0.23	0.25



Optimization Ant Colony System was proposed based on the ant colony algorithm. First, the resource cost model can be used to define the task demand for resources in detail. This model reflects the rela- tionship between resource costs and task costs. Secondly, a multi-objective optimization model was proposed, based on the model. Its main objective was to optimize the scheduling of performance and user costs. Thirdly, an improved ant colony algorithm was proposed to solve the optimization problem. In order to prevent the ant colony algorithm from falling into a local optimal solution, this method used the performance and budget constraint functions to evaluate the costs and provide feedback on the quality of the solution. It then adjusted the quality of the solution according to the results of the evaluation and feedback. Experimental results show that the OACS method has a great advantage in terms of makes pan. Even in the worst case and it is equal to Min-Min algorithm, which has advantages regarding completion time. At its best, OACS increases nearly 56.6% relative to the FCFS algorithm. The OACS methods are still better than other similar methods regarding other metrics, such as cost, the violation rate of deadline and resource utilization, proving the effectiveness of OACS.

REFERENCES

[1]. R. Bajcsy and M. Tavakoli, "Computer recognition of roads from satellite pictures," IEEE Trans. Syst. Man. Cybern., vol. 6, no. 9, pp. 623–637, Sep. 1976.
[2]. J. B. Mena, "State of the art on automatic road extraction for GIS update: A novel classification," Pattern Recognit. Lett., vol. 24, pp. 3037–3058, 2003.
[3]. C. Poullis, "Tensor-cuts: A simultaneous multi-type feature extractor and classifier and its application to road extraction from satellite images," ISPRS J. Photogramm. Remote Sens., vol. 95, pp. 93–108, 2014.
[4]. T. Peng, I. H. Jermyn, V. Prinet, and J. Zerubia, "Incorporating generic and specific prior knowledge in a multiscale phase field model for road extraction from VHR images," IEEE J. Sel. Topics. Appl. Earth Observ. Remote Sens., vol. 1, no. 2, pp. 139–146, Jun. 2008.
[5]. T. Peng, I. H. Jermyn, V. Prinet, J. Zerubia, and B. H. B. Hu, "Urban road extraction from VHR images using a multiscale approach and a phase field model of network geometry," in Proc. Urban Remote Sens. Joint. Event, 2007, pp. 1–5.
[6]. Dr.C.Vijay Kumar ,Optimization of data storage and QoS in data replication using the priority datasets.Pezzottaite journal,109-113.
[7]. Vijay Kumar, Thrusting Energy Efficiency for Data center in Cloud Computing Using Resource Allocation Techniques.
[8]. Vijay kumar, "Energy Conservation for Datacenters in Cloud Computing using Genetic Algorithms" International Journal of Emerging Trends & Technology in Computer Science
[9]. X. Lin, J. Zhang, Z. Liu, J. Shen, and M. Duan, "Semi-automatic extraction of road networks by least squares interlaced template matching in urban areas," Int. J. Remote Sens., vol. 32, pp. 4943–4959, 2011.