Optimization of task scheduling using improved crow search algorithm in a cloud environment

1Dr. A.Kousalya  2P.Sinduja  3V.Harshini Viswanathan  4Jeevitha

1Associate Professor, Department of Computer Science and Engineering, United Institute of Technology, India
kousivetri@gmail.com

2Assistant Professor, Department of Computer Science and Engineering, United Institute of Technology, India

3,4 Student, Department of Computer Science and Engineering, United Institute of Technology, India

ABSTRACT

In cloud computing, parallel processing is needed for the complex applications to execute the task. Due to the communication and synchronization among parallel processes, there is a decrease in utilization of control processing unit resources; therefore it is essential for the datacenter to achieve the utilization of nodes, while maintaining the level of responsiveness of parallel task. If the tasks are not scheduled properly, then it reduces the performance of the system. Hence resource allocation plays an important role in improving the performance of the entire system along with customer satisfaction, and also the resource allocation should be economical to the service provider. A new task allocation model is proposed for optimizing the task scheduling. Optimization algorithm called Improved Crow Search Algorithm (ICSA) is implemented, to achieve the optimized task allocation scheme that determines the optimal virtual machine to perform the task effectively hence, it reduce the task execution time, waiting time and makespan.

Keyword: Cloud Computing, Scheduling, Crow Search Algorithm, Virtual Machine, Particle Swarm Optimization

INTRODUCTION

Cloud computing enables companies to consume a computable resources such as virtual machine storage or an application as an utility-just like electricity rather than having to build and maintain computing infrastructure in house, cloud computing has some challenges for Eg: security, performance, resource management, reliability etc. One of the
resource management issues is related to task scheduling with a number of processing where only one process can use the CPU at a time and each process is specialized in one and one task. The benefits of cloud computing [12] are self provisioning, elasticity, pay per use, workload resilience and migration flexibility. Cloud computing deployment models are private cloud, public cloud, hybrid cloud and multi cloud. In public cloud model, a third party cloud provides services over the internet and are sold on demand, typically by the minute or hour long-term commitments are available for many services. A hybrid cloud is a combination of public cloud services and an on premises private cloud with orchestration and automation between the two companies can run mission critical workload or sensitive application on the private cloud and use public cloud to handle workload bursts or spikes in demand. Multi cloud enables application to migrate between different cloud providers or to even operate concurrently across two or more cloud providers. Private cloud services are delivered from a business data centers to internal users. This model offers the versatility and convenience of the cloud while preserving the management centers. The services models are Software as a Services(SaaS),Platform as a services(PaaS), and Infrastructure as a services (IaaS). SaaS is highly scalable internet based application are hosted on the cloud and offered as a services to the end users, PaaS is a platform used to design, develop build &test application provided by the cloud, IaaS is a data based management and compute, capabilities are offered on demands. The advantages of cloud computing are cost savings, reliability, manageability, strategy and edge. In cost saving most significant cloud computing benefits is in terms of IT cost savings. Business, No matters, what they are type or size, exist to earn money while keeping capital and operational expenses to a minimum. The cloud computing can save substitution capital cost with zero amounts, in server storage and application requirements. In reliability the managed services are platform, cloud computing is much more reliable then in IT infrastructure. The disadvantage of cloud computing is down time, security, vendor lock in, limited control. Task scheduling plays a vital role in cloud computing, and in many critical application, many scheduling policies are being used by the master nodes for computing nodes to efficiently distribute its tasks. This scheduling and utilization of resources is one of the most critical problem in cloud.

The next section contains related work, problem statement, proposed methodology, mathematical model, architecture combined with crow search and PSO, experimental result and conclusion.

RELATED WORK

Omara & Zohier (2010) analysed a dynamic task scheduling model which was used to improve the fuzzy decision in task scheduling on a network of processing elements by introducing new input parameters to an existing fuzzy model and, at the same
time, improving the load balance on the network in a dynamic environment. In this model, tasks are generated randomly and served based on First-Come First-Serve (FCFS) basis. The modified fuzzy logic model leads to more precise fuzzy decisions, even while dealing with a larger number of processors and/or a larger number of tasks, while increasing the number of involved parameters in the fuzzy model, relative to the existing one.

Ge & Wei (2010) proposed a new scheduler, which made the scheduling decision by evaluating the entire group of tasks in a job queue. A GA was designed as the optimization method for a new scheduler that provided better makespan and better balanced load across all nodes than FIFO and delay scheduling.

Li et al. (2011) proposed Hybrid energy efficient scheduling algorithm used for private cloud computing. This algorithm uses dynamic migration and the experiment results show that the proposed algorithm reduces response time, conserves more energy and achieves higher level of load balancing.

It was to address the cloud scheduling problem that Yan-Hua et al. (2011) designed a cloud scheduling algorithm with a dynamic combination of the Genetic Algorithm (GA) and Ant Colony Optimization algorithm (ACO), transforming the initial solution obtained by the former into the pheromone primary value, required by the latter, deriving the optimal solution. Genetic control function was constructed to control the opportunity of two algorithms’ integration, enhancing its efficiency for the sake of reducing the dynamic load of cloud.

In order to minimize the cost of the processing, Guo et al. (2012) introduced a model for task scheduling and Particle Swarm Optimization (PSO) algorithm. In order to enhance the efficiency, optimizing task scheduling is indispensable. Experimental result shows that the PSO algorithm accomplishes optimal solution and converges more rapidly in large tasks than the other two approaches do. Moreover, the execution time is lesser than that of the other two and it is clear that PSO is evidently more suitable.

Sourabh Budhiraja & Dheerendra Singh (2014) proposed a task scheduling algorithm based on Multi-Objective Genetic Algorithm (MOGA) for reducing the execution cost and time paying attention to crossover operators, selection operators, mutation operators, and the pareto solutions method. The results showed that the proposed MOGA algorithm had obtained a better solution than the rest.

Allocation-aware Task Scheduling (ATS) algorithm framework was introduced in heterogeneous multi-cloud systems by Panda et al. (2015). In this framework, allocation phase is used to reschedule the tasks to meet the scheduling requirements, playing a vital role in fulfilling the gap between scheduling and
Bhaskar et al. (2012) introduced a dynamic resource allocation method for cloud computing. Dynamic scheduling and consolidation mechanism were proposed to allocate resources based on the load of VM on infrastructure as a service. Based on the load and the consolidation, the users could dynamically add or delete the instances.

Shenai (2012) explained how cloud computing had dealt with different kinds of virtualized and physical resources. In cloud, users may use many virtualized resources for different tasks, and so, the manual scheduling is not a feasible solution. The automatic scheduling to a cloud environment enables the use of various cloud services to help framework implementation. The different types of scheduling algorithms surveyed include the workflow scheduling, as well as grid scheduling. Many algorithms, such as ACO, gang bang scheduling, and dynamic scheduling, were discussed, and the load was allocated based on priority.

According to Gupta & Champaneria (2013) the scheduling algorithms should order the jobs in such a way that there is balance between improving the performance and quality of service maintaining the efficiency and fairness of the jobs. This method considers various scheduling algorithms, like SJF Scheduling, job scheduling model based on MOGA, priority based job scheduling algorithm, SLA-Tree and Enhanced Max-min task scheduling algorithm.

**PROBLEM STATEMENT**

Task scheduling and resource allocation is a major issue in cloud computing. The scheduling of task with efficient resource allocation provides the maximum benefits to the cloud service provider in terms of quality of service assuming that there are ‘m’ number of task with execution time and processing time which is to be compute on ‘n’ computational resources the number of task considered is more than the resource (m>n), the goal of task scheduling is to maximize task execution, make span, time, response time, flow time, cost to improve throughput, resource utilization.

In cloud computing environment all the resource are sharing by using the virtualization technology, therefore the resources are represented as virtual machines and each virtual machine has a limited capacities, for example processing power, memory size and network bandwidth. The speed of each virtual machine is calculated in millions instruction per seconds, a job has processing requirement obtained in terms of millions instructions.

Task set is defined as $T=\{t_1,t_2,\ldots,t_m\}$, where $m$ represent the number of task and the $vm$ set is defined as $vm=\{vm_1,vm_2,\ldots,vmn\}$ where $n$ represent the number of virtual machine to evaluate the efficient makespan is extensively used metric for measuring the quality of a scheduling in cloud environment, the makespan can be formulate as
Makespan = completion time of last task – completion time of first task

**PROPOSED METHODOLOGY**

The proposed algorithm is the combination of Crow Search Algorithm and Particle Swarm Optimization called as ICSA. The objective of the proposed work is to find the availability of the resources and schedule the job to the appropriate virtual machine. The virtual machine is clustered using clustering intra cluster of nodes using PSO, the fitness function is evaluated based on the capacity of the virtual machine, and bandwidth based on the evaluated function the Pbest virtual machine is computed and in the next iteration if fitness value is found better than the previous fitness value, replace the newer one as Pbest and finally the best fitness value in the population is considered as gbest. Finally update the memory of the virtual machine by using the Crow Search Algorithm.

**MATHEMATICAL MODEL**

Let VM=vm1,vm2,...,vmn be the number of virtual machine that process m number of tasks represented as T=T1,T2,...,Tm. Our aim is to reduce the makespan which is denoted as CPmn, the processing time of each task for each virtual machine is calculated by the equation,

\[ P_{mn} = \frac{C_m}{PS_n} \]

Where \( P_{mn} \) represents the processing time of the task by the virtual machine and \( C_m \) represents the computational complexity of the task. \( PS_n \) represents the processing speed of \( vm \). The makespan is reduced by comparing the execution time of a task and execution time of all the tasks.

\[ et_{mn} \leq CP_{max} \]
\[ et_n \leq CP_{max} \]

Before assigning a task to the virtual machine its capacity, existing workload, bandwidth and the execution time is calculated as

\[ C_m = Pnum_m + Pmp_m + Mb_{wm} \]

Where \( Pnum_m \) is the number of processors, \( Pmp_m \) is million instructions per second of all processors and \( Mb_{wm} \) is the
communication bandwidth. Overall capacity of the VM is calculated as,

\[ C = \sum_{m=1}^{n} c_m \]

**K-means algorithm**

K-means is the unsupervised learning algorithm that solve the well known clustering problems. Clusters the data into K groups where K is predefined. Select K points at random as cluster centers. Assign objects to their closest cluster euclidean distance function calculated the centroid or mean of all objects in each cluster.

\[ T = \sum_{j=1}^{k} \sum_{i=1}^{k} (||x_i - c_i||)^2 \]

Where,

\[ ||x_i - c_i|| \] is the euclidean distance between xi and yi.

\[ k_i \] is the number of data points in i\textsuperscript{th} cluster.

\[ k \] is the number of cluster centers.

**Algorithm**

Step 1: Select k points as the initial centroids, where k represents the number of clusters specified by the users.
Step 2: Update the centroid of each cluster based on the number of points.
Step 3: Assignment and update process repeat until there is no change in the centroid position.
Step 4: Repeat step 2 & step 3 until the centroid is no longer more.
Step 5: End algorithm.

It produce a separation of the object into group from which it can be minimized.

**Crow Search Algorithm**

The crow search algorithm is a meta heuristic algorithm was proposed by Askarzadeh in the year 2016. This algorithm works based on the intelligent behaviour of crows, where the crows store excess food in hiding places and restore when it is needed. The hiding place of the food is memorized by crow \(i\), and in the search plane it tries to find the best food resources which is defined as \(n^k_i\). And finally the crow updates the memory.

**Algorithm**

**Step 1:** Initialize the problem (the position of N crow in the search space, flock size, flight length & maximum number of iteration).

**Step 2:** Initialize the position & memory of crows.

**Step 3:** Evaluate the fitness function of each crow.

**Step 4:** Implement the new position of each crow in the search space.

**Step 5:** Check the feasibility of new position.

**Step 6:** Evaluate the new position of the crow.

**Step 7:** Evaluate fitness function of new position.

**Step 8:** Update the memory of the crow.

**Step 9:** Check termination criterion in term of the maximum number of iteration.

**Step 10:** End algorithm.

**Partial Swarm Optimization Algorithm**

PSO algorithm is swarm intelligence based which is initialized with a group of random particles (solutions) and then searches for optima by updating generations. In every iteration, each particle is updated by following two "best" values. The first one is the best solution (fitness) it has achieved so far. This
value is called pbest. Another "best" value that is tracked by the particle swarm optimizer is the best value, obtained so far by any particle in the population. This best value is a global best values called as gbest.

Algorithm

Step1: Initialize the particle, input function f to optimize, swarm size s, problem dimension d.

Step2: Search space range, \([N_{\text{min}}, N_{\text{max}}]\), Velocity range \([M_{\text{min}}, M_{\text{max}}]\).

Step3: output is gbest, the best value found.

Step4: Initialize for all particle in problem space.

\[
M_i = (M_{i1}, \ldots, M_{id})
\]
\[
N_i = (N_{i1}, \ldots, N_{id})
\]

Step5: Evaluate \(f(x)\) in \(d\) variables and get Pbest; \((i = 1, \ldots, s)\)

\[
\text{Gbest} < \text{best of Pbest.}
\]

Step6: Repeat and calculate \(w\).

Step7: update \(M_i\) for all particle by equation

\[
M_{i(J+1)} = T + M_i(J) + (i + r_i + (\text{Pbest}_i(J) - N_i(J)) + c_2 + r_2 + (\text{gbesti}(J) - N_i(J))).
\]

Step8: update \(N_i\) for all particle by equation

\[
N_{i(J+1)} = J_i(J) + M_i(J+1).
\]

Step9: Evaluate \(f(x)\) in \(d\) variables and get Pbest \((i = 1, \ldots, s)\).

Step10: if \(f(x)\) is better than p best than p best =Ni.

Step11: if the best of pbesti is better than gbest then

\[
\text{Gbest} < \text{best of pbesti.}
\]

Step12: until stopping criteria (eg: maximum iteration or error tolerance is met).

Step13: return gbest.

Step14: End algorithm.

Hybrid Crow Search Algorithm and Particle Swarm Optimization Algorithm

CSA and PSO have their specific advantages while solving different problems. In this approach the individual features of the algorithms are combined to present a novel hybrid algorithm to optimize the task allocation scheme.

Initially the clustering is done by using k means clustering algorithm for clustering intra cluster of nodes. Initialize the population of the task and the virtual machine, and then evaluate the fitness function of each virtual machine by using the PSO. The fitness value is calculated by the bandwidth and the capacity of the virtual machine and for the task, with the minimum execution time and deadline are obtained. The best fitness value is chosen as the gbest virtual machine and then allocates the task to the optimal virtual machine. The algorithm for the proposed ICSA is given below.

Algorithm

Input: Task T, Task execution time and deadline.

Output: Optimal Resources Allocation to virtual machine.

Begin

Initialize random population \(n\) of VM.

Repeat

For all \(VM_i\) do

\[
T = \sum_{j=1}^{k} \sum_{i=1}^{k} (||xi-ci||)^2// \text{Using k means centroid.}
\]

\[
\text{Fitness}(x_i) < \text{bandwidth and capacity of VM.}
\]

Current <fitness\((x_i)\).
If current < Pbesti, then
   \[ P_i < X_i \]
   \[ P_{\text{best}i} < \text{current}_i \]
End if
Update \( X_i \)
End for
Gbest < best fitness value
Select gbest as optimal VM
// Generate new position of VM in cluster
   New position < gbest VM in intra cluster of nodes
Check the feasibility of new position of VM
// The fitness function of the new position of the VM in the cluster is calculated by the objective function.
If new position of VM in cluster < previous VM in cluster.
   Elect new position of VM in cluster in optimal VM.
Update the position of the virtual machine.
Assign task to virtual machine // for all VMi
End if
Until reach maximum iteration.
End.

EXPERIMENTAL RESULT
The experiment was carried out by CloudSim toolkit which is a simulation platform for modeling and simulating the cloud computing environment. CloudSim was integrated with Java-based IDEs (Integrated Development Environment) including NetBeans. Using NetBeans IDE, the CloudSim library could be accessed and the cloud scheduling algorithm was implemented. The simulation parameter values were used in the CloudSim tool for evaluating existing and the proposed scheduling algorithms in the cloud environment, and the simulation parameters were resource ID, VM, cloudlet, Bandwidth and RAM.
The final report of the implementation phase includes procedural flowcharts, record layouts, and a workable plan for implementing the candidate system design into an operational design. The implementation process done using net beans using cloud sim. The datacentre is created with datacentre id, operating system and the version is being updated. Then the new agent is created and they are assigned their respective agent id. The virtual machine is created using agent id, number of processor, RAM, Maps and bandwidth. Then the job is being created using broker id, length, file size and output size. The cloud length is mapped and analysed with the position to the assigned velocity. The PSO is estimated and they are scheduled to the assigned virtual machine. The gbest virtual machine is allocated to the assigned job.

RESULT ANALYSIS
WAITING TIME
The sum of the periods spent waiting in the ready queue.
MAKE SPAN

The makespan is the total time that elapses from the beginning to the end. The term commonly appears in the context of scheduling. There is a complex project that is composed of several sub-tasks.

RESPONSE TIME

Response time is the average time elapsed from submission till the first response is produced.

CONCLUSION & FUTURE WORK

The utilization of CPU resources is essential for a data centre to achieve the utilization of nodes. The efficient scheduling algorithm is needed for allocating the task to the resources, so that the waiting time of the
task can be minimized. We proposed an Improved Crow Search Algorithm (ICSA) which is a combination of Crow Search Algorithm and Particle Swarm Optimization Algorithm, for efficient utilization of resources. The ICSA algorithm outperforms the result when compared to the existing Cuckoo Search Algorithm in terms of Make span, flow time, response time, waiting time of the tasks are reduced, and it increases the throughput. In future work, the energy will be consumed so that the system will save more energy use and it will take less battery power, we will extend the studies of Energy Consumption Scheduling (ECS) seems to be at steady state if we increase the number of tasks and processors. The ultimate goal of energy efficient scheduling is to reduce cost and computing infrastructure.

References
