Enhanced Crow Search Algorithm for Task Scheduling in Cloud Computing

K R Prasanna Kumar , K Kousalya , S Vishnuppriya , S Ponni and K Logeswaran
Department of Information Technology, Kongu Engineering College, Erode,Tamilnadu, India
Department of Computer Science and Engineering, Kongu Engineering College, Erode, Tamilnadu, India
Department of Computer Science and Engineering, Velalar College of Engineering and Technology, Erode, Tamilnadu, India
*Email: krprasanname@ gmail.com

Abstract

The task scheduling in the cloud computing environment seems to be the most noteworthy problem. It is the important move to enhance the cloud performance and efficiency. The nature inspired algorithms are applied to perform the scheduling in cloud. The Crow Search Algorithm (CSA) is used to assign the task to Virtual Machine (VM). In the CSA the random selection of task is used which may downgrade the performance of cloud. The random task selection is improved with task selection and VM selection algorithms. The improved version of CSA is called as Enhanced CSA (ECSA). The performance of ECSA compared with CSA and Max-Min algorithm.

Keywords: Cloud computing, Task Scheduling, Crow Search Algorithm, Makespan, Optimization

1 Introduction

In the present information technology world the Cloud computing based applications and services are playing vital role[1]. It is providing the services in different forms based on the pay as you go model. In such cloud environment the major issue is to meet the requirements of customers. The cloud computing which interconnects resources around the globe and provides the service in the form of software, storage and platform[2]. Managing and effectively utilizing the resources is a biggest challenge. The mapping of task to VM is the major role which needs to satisfy both the task requirement and effective utilization of VM. The scheduling algorithm plays the important key role in the mapping of task to VM as well as ensuring the performance of cloud system[3]. The task scheduling

problem belongs to the class NP-hard. Several conventional algorithms and the metaheuristic algorithms such as swarm intelligence algorithms were used to solve the task scheduling[4]. The parameters most considered for the scheduling are the execution time, completion time, make span, load, cost and so on[5]. But still the task scheduling is a open problem which needs a optimum solution. The rest of the article is organized as defies: section 2 elaborates the different scheduling algorithms proposed for task scheduling. Section 3 defines the CSA and ECSA algorithms. Section 4 presents the simulation results and the section 7 defines the conclusion.

2 Literature Review

The Hybrid Moth Search Algorithm and Differential Evolution (MSDE) algorithm is proposed for task scheduling, which mainly focusing on makespan minimization[6]. It simulates the characteristics of moths to fly towards the light in nature using phototaxis and Levy flights techniques. These techniques represent the exploration and exploitation ability and these factors can be improvised by DE (Differential Equation) as a local search method[6]. The performance of MSDE algorithm is compared with experiment aims in testing over two criteria: by comparing over the traditional moth search algorithm and other heuristic and meta-heuristic algorithms which marked a acceptable performance thereby reducing the makespan.

Chemical Reaction Multi-objective Optimization Algorithm (CRMOA)considers four chemical reaction operators namely on-wall ineffective collision, inter-molecular ineffective collision, decomposition and synthesis[7]. These operators were used for directed acyclic graph based task scheduling. The experiment results indicates that the CRMOA can generate the optimal solution for mapping the task and VMs with the better performance[7].

Crow-Penguin Optimizer for Multiobjective Task Scheduling (CPOMTS) strategy is used to execute the task optimally using cloud resources in minimal period of time[8]. It is the fusion of the Crow Search optimization Algorithm (CSA) and the Penguin Search Optimization Algorithm (PeSOA) which exhibits better convergence rate with a global optimal solution[8]. The analysis was accomplished with the real time setup and it attained the result of maximum of QoS with the minimal resource utilization cost, make span and load.

The Grey Wolf Optimizer (GWO) scheduling algorithm is developed based on the hunting nature of the grey wolfs[9]. It minimizes the makespan value using GWO approach. A novel Modified Genetic Algorithm with Greedy Strategy (MGGS) is applied to optimize task scheduling process[10]. MGGS can find an optimal solution using less number of iterations. It focuses on coding, fitness function, crossover, mutation operator, greedy selection for better optimization of task scheduling process. MGGS algorithm uses binary code for its stability and with a large population of diversity[10]. Fitness value is calculated for the population and chooses several pairs of codes to do the crossover operation which forms new population. Mutation operation is performed through selection oper-

ator and greedy choices for newly formed population. The experimental findings indicate that the algorithm benefits from reducing the overall execution time, average response time and achieves load balancing over the cloud resources with minimum cost[10].

Modified Ant Colony Optimization (MACO) algorithm is applied for task scheduling[11]. The basic ACO algorithm dramatically boosts load of tasks in runtime, and it has lack of rapid adaptability. Thus increases execution time and decreases convergence rate. MACO algorithm is exploited to overcome these problems in real-time cloud environment[11].MACO can be further extended to various types of typical workloads on different combinations of VMs with their variety of processors that give several combinations of processing capability resulting in more efficient and less complex cloud computing frameworks.

Game Theory (GT) approach is proposed for the balanced task scheduling and energy management in cloud computing[12]. GT is able to organize the distribution of task and the distribution of energy for managing huge amount of data. The task scheduling problem in cloud computing is moulded as a multi-stage sequential game model. Reliability of the node is considered as the optimization objective. Game strategy is used on the node to calculate the rate allocation and a mathematical model to calculate the stability of nodes[12]. The task scheduling algorithm based on sequential game shows better benefit in response time. The experimental result proves that the algorithm has better optimization effect.

3 CSA for Task scheduling

3.1 Crow Search Algorithm

The traditional food gathering pattern of the crow clearly states that it achieves a better food source by its repeated spying and position movement activity[13]. It is applied to task scheduling in cloud to reduce the makespan, degree of imbalance and to improve the VM utilization [14]. It is assumed that each task is more compute intensive which represented as T. The VMs are the computing elements. Each of the tasks would hold a VM as well as involve in the search process to find a better alternative VM, and whenever it comes across a better VM, the task would replace the current VM with the newly identified one[14]. The CSA is used to identify the better VM for the task. The task is considered as crow and the VM is considered as source.

CSA starts with an initial schedule and the execution time and completion time are components considered in task scheduling which directly create an impact upon the makespan value. Based on the initial schedule for each task, the execution time and completion time are calculated. The CSA is applied to each Ti in the schedule. In this, the new position denotes the alternative VM (VMx). A random task Tj is selected from the schedule which is considered as the task to be followed by Ti. The CSA position movement is applied on the Ti based on the Tj values using the Equation (1)[14].

where AT denotes an allocation table which maintains the information of the currently allocated VM for each task in the schedule. r_{i} and $\mathrm{r_j}$ are random values generated between 0 and 1.$\theta^{i,k}$ is considered as 0.05 and $\lambda^{i,k}$ is increased from 1.5 to 2.5 [14]. The value of $\mathrm{VM_x}$ is calculated when the $r_j \geq \theta^{i,k}$, otherwise a random VM is selected by the $\mathrm{T_i}$. The completion time of the task in the $\mathrm{VM_x}$ is calculated and compared with the present completion time of the task. If the $\mathrm{VM_x}$ provides lesser completion time than the present allocated VM, then the present VM details will be replaced with the $\mathrm{VM_x}$ and completion time of the task will be updated accordingly. The modified schedule is considered as a new schedule. Similarly, for each task in the current schedule, new $\mathrm{VM_x}$ is identified and updated in the new schedule[14].

At each iteration of CSA algorithm, the makespan value is computed for the new schedule and compared with the existing schedule[14]. If the new schedule yields lesser makespan than the existing schedule then the new one will be considered for the next iteration, otherwise it will be discarded and the CSA will proceed with the existing schedule. This CSA changes the VM of the schedule in its own style which identifies the VM that provides minimum completion time.

3.2 Enhanced Crow Search Algorithm

In the CSA for task scheduling a random task is selected for follow-up process during each iteration. In real-time cloud scenario, the random task selection in certain cases yields best results, but in some cases, it may downgrade the performance of the cloud[14]. The reason is that one task gets benefitted from others, so if the select task has better resource, then it will minimize the makespan of the schedule; otherwise, it will increase the makepsan of the schedule. Even though there are a number of crows in the flock, identifying the suitable mate to follow up plays a requisite role in CSA process, which increases the hit ratio. So, instead of selecting a random task, selecting a better task would offer better results. Selecting the fittest task would improve the makespan further in each iteration. Local changes are applied to the search space again and again by moving the candidate solutions until getting an optimal solution. The selection of T_j to follow may be done in different ways by examining the existing schedule similar to the spying activity of the crow. In the task scheduling problem considered, the execution time and completion time are the two parameters that are closely associated with makespan. The execution time of the task is purely based on the characteristics of the individual processing capacity of VM and the length of the task. Further, it may not consider the present availability or the workload of the VM. So, the VM selection based on execution time may not yield better performance. Alternatively, the completion time would be another notable criterion which is computed by adding the execution time of the task with the present available time of the VM. If the VM is already allocated with more tasks, the new completion time of the task would be higher than the present completion time of the task. So, the completion time indirectly indicates the current load of the VM. Selecting a task T_j which has lesser completion time than the T_i considerably yields a better result than the random selection method

because if a VM executes a task faster, then it may execute other similar tasks in the same speed [14]. In this way, the process of CSA is enhanced and the new version is called ECSA. The steps involved in finding the task with minimum completion time are indicated in Algorithm 1. ECSA improves the probability of getting a better VM for T_i and reduces the makespan of the schedule.

Algorithm 1 CSA Task Search Algorithm

```
Variables:
CT_i            -       Completion time of the task i
 Input:
        CT_i, CT
Output:
        Task id
Search_min_task:
Begin
        For j = 1 to N do
                If  CT_j<CT_i then
                        return j
                End If
        End For
 End
```

Similarly, when the r_j value is lesser than $\theta^{i,k}$, a random VM is selected because it is similar to the crow's random position selection process. The status of the randomly selected VM is not known, which gives a scope for different questions with respect to the performance and as well as the outcome of the schedule. Hence, it is better to choose a VM based on the current load and as well as the one which is readily available to execute. This way of selection of VM would balance the load among the VMs. The present load value is calculated for each VM and the average load of the VM is also calculated. The CSA VM selection algorithm selects a suitable VM which has the load below the average load and with minimum free time. This is another enhancement made on the CSA algorithm which further reduces the makespan and balances the load among VMs. VM selection algorithm processes given in Algorithm 2.
Algorithm 2 CSA VM Selection Algorithm
Variables: Input:

```
        M, FT, LD
Output:
        Min_VM
CSA_Select_VM_Algorithm:
Begin
        For j=1 to M do
                Total=Total+LD_j
        End For
        Avg_load=Total/M
        Min_FT=FT_1
Min_VM=1
        For j=2 to M do
                If  ((FT_j<Min_FT) && (LD_j<Avg_load)) then
                        Min_VM=j
                        Min_FT=FT_j
                End If
        End For
        returnMin_VM
```

4 Results

The simulation environment is configured with 16 VMs and tasks from 500 to 2500 using the CloudSim simulator. In the ECSA tuning parameters of awareness probability is set at 0.05 and the flight length is increased from 1.5 to 2.5 [14]. Figure 1 illustrates the performance of the CSA[14], ECSA and Max-Min [15]algorithms. It can be seen that the makespan value of the ECSA is minimized well while comparing with Max-Min and CSA.

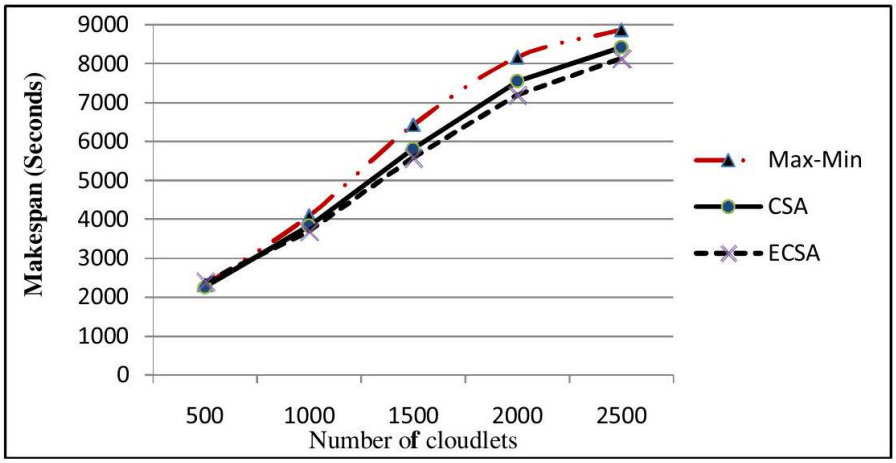


Figure 1.Makespan comparison of Max-Min, CSA and ECSA
Figure 2 indicates, ECSA reduce the degree of imbalance while comparing with CSA and Max-Min algorithms even when the number of the task is increased gradually because of the improvements carried out in the VM selection process. In the ECSA, the VM selection algorithm considers the completion time of the task, current load and free time of the VM, so it ensures the task allocated to the VM with the minimum load. The ECSA achieves an overall minimum degree of imbalance, which assures that it balances the load among the VMs well compared to even CSA.

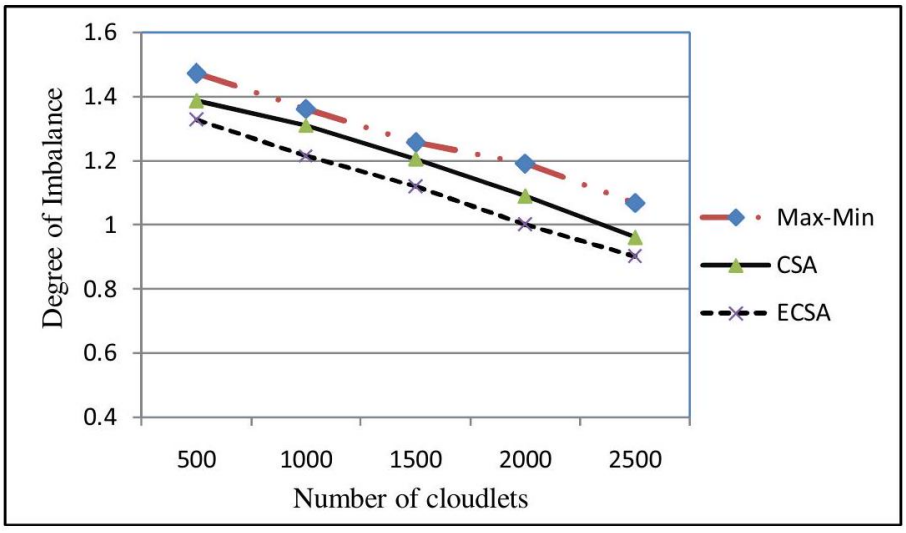
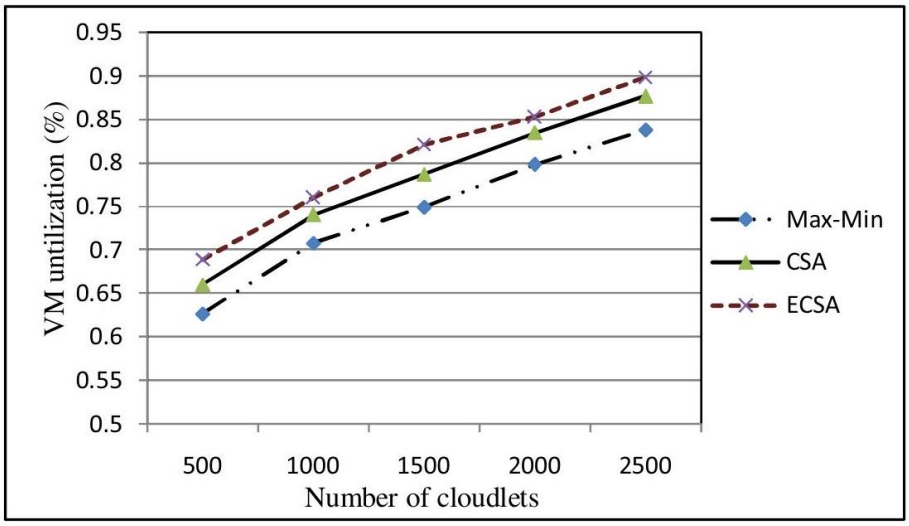


Figure 2.Comparison of Max-Min, CSA and ECSA based on degree of imbalance Conclusion

The VM utilization is taken into account as another parameter. The ECSA utilize the VM in a better way while comparing with Max-Min and CSA as shown in Figure 3. The CSA algorithm mostly distributes the task to different VMs and performs better than CSA and Max-Min. The ECSA utilizes the VM very effectively and a high level of utilization is reached.



5 Conclusion

In this paper discussed the ECSA for task scheduling in cloud. In ECSA, the task selection and VM selection algorithms are directly used to control the random selection of the CSA algorithm. The results of CSA and ECSA have been compared with CSA and Max-Min based on different benchmarks like makespan, degree of imbalance and VM utilization, Based on the set of benchmark values, it is observed that CSA and Max-Min algorithms are outperformed by ECSA. From the results, it is seen that the performance of ECSA is good and the algorithm find the optimal schedule for the task scheduling in cloud.