Indian Journal of Science and Technology, Vol 9(48), DOI: 10.17485/ijst/2016/v9i48/106514, December 2016
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

An Enhanced Approach for Performance Improvement using Hybrid Optimization Algorithm with K-means++ in a Virtualized Environment

A. P. Nirmala[1*] and S. Veni[2]

[1]New Horizon College of Engineering, Bangalore 560103, Karnataka, India; nirmala_ap@yahoo.com
[2]Department of Computer Science, Karpagam University, Coimbatore – 641021, Tamil Nadu, India; venikarthik04@gmail.com

Abstract

Virtualization plays a vital role in cloud computing. It provides better manageability, availability, optimistic provisioning, scalability and resource utilization in current cloud computing environments. However the performance issue is a major concern in virtualization. The performance of the application running inside the virtual machine gets affected by the interference of the co-virtual machines. This approach provides a novel task scheduling mechanism that allocates the suitable resources to virtual machines which are running in parallel. An interference prediction scheme is proposed to utilize characteristics that are collected when an application running on virtual machines to maintain less system overhead. Nelder-mead method is employed in prediction to create relationship model from the observed response and control variables. A hybrid algorithm: Ant Colony Optimization and Cuckoo search algorithm with K-means++ is adopted for task scheduling process. This approach shows effective improvements in terms of throughput and execution time.

Keywords: Ant Colony Optimization, Cuckoo Search, K-means++ Algorithm, Performance Interference, Throughput, Virtualization

1. Introduction

Cloud Computing (CC) is considered as a novel technique in the Information Technology arena. This cloud computing mechanism is considered as the extension of distributed computing, parallel computing and grid computing. In CC, delivery models based on its services are Software as a Service, Infrastructure as a service and Platform as a Service. The user can access these services by the Pay per-Use-On-Demand model that allows the usage of shared IT resources such as data storage, server, application, network through internet[1].

In CC, virtualization is a eminent technology where a single server sharing among multiple customers. In virtualization, a server is divided into multiple Virtual Machines (VMs) which provide convenient management abstraction and resource confinement among users. However hypervisor does not provide performance isolation effectively and this leads to existence of interference in a VM due to other guest VMs remains a bottleneck. Thus the performance of one VM may be influenced by other guest VMs if the hypervisor fails to impose suitable priorities between guests.

Zhang Y et. al.[2] proposed a scheduling algorithm to improve response time, throughput and utilization in large super computing environment. A schedule algorithm proposed by Angelou et. al.[3] considers workload interference in order to improve host efficiency and VM performance.

Rui Han et. al.[6] proposed a dynamic and interference-aware scheduler designed for large-scale cloud services. In this approach, latency of component and performance are predicted by collecting the workload and resource data on running services at each scheduling interval. The interference-aware scheduler identifies the non uniform components and performs the near optimal component-

*Author for correspondence

node allocations based on the predicted performance. This approach is designed with modules namely on-line monitoring, performance prediction and scheduling. However this system does not reduces the component latency variability and tail latency effectively.

Wei Zhang et. al.[7] designed a Minimal Interference Maximal Progress (MIMP) scheduling approach that maintains VM CPU and Hadoop Job scheduling for interference reduction. It also improves the overall system efficiency. It increases the cluster utilization by providing the priority of different VMs and the availability of resources on various servers. This approach enables web applications to achieve twice the throughput and gives response time of the web application which is, as similar to response time when it runs alone. The execution time is reduced effectively but this approach is still meet more deadlines.

Chin-Fu Kuo et. al.[8] studied the issues faced in allocating the resources in a distributed environment for real-time systems and developed a framework for resource allocation in a two ways such as on-line and off-line phase. The first phase uses the greedy approach whereas the second phase applies the dynamic programming approach for sub-optimal resource configurations. The author proposed this approach to deal with run-time overhead which includes time and memory space by performing resource configuration assignment in a best possible way.

Yu-Chon Kao and Ya-Shu Chen[9] introduced a data locality aware MapReduce real time scheduling model which enhances the quality of service for interactive MapReduce applications. A scheduler is used for scheduling two-phase Map Reduce jobs and a dispatcher is applied for assigning jobs to computing resources by enabling the consideration of blocking and data-locality. The author also discussed the dynamic power management for run-time energy saving.

Hadi Salimi and Mohsen Sharifi[10] proposed a novel batch scheduler for a consolidated multi-tenant virtual environment to reduce the interference of running tenant VMs. This is done by pausing VMs based on severity of impact caused by intensity of interference. In this approach, the authors discussed a model which identifies interference by considering utilization of network and processor of VMs. This model is used in the scheduler to estimate the interference of a virtualized environment based on the quantity of concurrent VMs and utilization of processor and network in VMs.

From the above observation the prior techniques does not build a more accurate model and does not reduce system overheads effectively. To overcome these limitations, the efficient task scheduling mechanism proposed in our paper which uses statistical machine learning techniques with model and control. The remainder of this paper discusses proposed methodology in section 2 and results and discussion in section 3. Section 4 discusses conclusion with future work.

2. Proposed Methodology

The proposed approach is designed for the virtualized environment to minimize execution time and increase throughput of data-intensive application. This section describes the proposed methods and technologies in detail. The proposed work has two phases: Interference prediction model and Scheduling phase.

2.1 Interference Prediction Model

Interference Prediction Model extrapolates the performance of the running data-intensive application in terms of resource utilized by VM. In this approach, Nelder–Mead method[4] is utilized to frame the relationship between the observed response and controlled variable. This is done by characterize each application by controlled variables such as the read throughput, the write throughput, incoming bandwidth, outgoing bandwidth, Number of Processors, CPU utilized by guest VM and CPU utilized by virtual machine monitor.

2.1.1 Nelder-Mead Method

The Nelder–Mead method is represented as a downhill simplex and numerical method to obtain the minimum/maximum objective function in a multidimensional space. The Nelder-Mead method[11,12] is adopted widely as direct search method to solve unconstrained optimization problem. Consider the term, $\min g(a)$, where $g: R^l \to R$ is a objective function in l dimension. Given l dimension a simplex is denoted with vertices $a_1, \cdots, a_{l+1}$. This method repetitively produces a series of simplices to approximate an optimal point of $\min g(a)$. Based on objective function values, vertices $\{a_i\}_{j=1}^{l+1}$ of the simplex are ordered at each iteration. The objectives values are

$$g(a_1) \leq g(a_2) \leq \cdots \leq g(a_{l+1}) \quad (1)$$

where a_1 denotes the best vertex and a_{l+1} denotes worst vertex. According to Jeffrey C. Lagarias et. al.[13], the objective values of several vertices are same, then rules to be followed to handle tie-breaker. The method performs four operations namely reflection, expansion, contraction and shrink. Each operation has a parameter which are scalar and denoted as ρ , μ , τ and δ for reflection, expansion, contraction and shrink respectively. The values of these satisfy ρ >0, μ >1, 0< μ <1, and 0 < δ <1. If $\bar{a}$ be the centroid of *l* best vertices then,

$$\bar{a} = \frac{1}{l}\sum_{i=1}^{l} a_i \quad (2)$$

2.1.2 Model Training and Learning

Interference profile is generated for an application by running it in a single VM while remaining VMs are running different workloads in background in an environment where there are n VMs. By doing so a collection of information is obtained on interference effects under several background workloads. This approach can be simply automated and it helps online learning from interference prediction mechanism in a cloud platform. When it is not possible to obtain interference relationships among multiple applications, this model can be monitored and updated dynamically. This is due to the changes in operating systems, application workloads, VMs, and infrastructures of cloud environment.

2.2 Hybrid Optimization Algorithm with K-means++

Based on the model training phase the available virtual machine is clustered according to the CPU capacity using K-means ++ algorithm[5]. The incoming task is allocated to each cluster by Hybrid algorithm: Ant Colony Optimization (ACO) and Cuckoo Search (CS).

ACO is an efficient algorithm for solving the computational problems that finds the best path using graphs. An ant deposits pheromone during the food searching process that can be used for exchange of information, like the status of their "work". The ant searches the food source and then it reverts to nest. While coming back, a trace of pheromone is left in the path by it. Iteration is based on ant movement from one state 'a' to another 'b'. Each ant k performs set $A_k(a)$ of optimal expansion to current state of an ant during each loop. Move attractiveness α_{ab} and the trail of the move Tr_{ab} are the two moves used to measure the probability ρ_{ab}^{k} of the moves from one state to another. The probability of the k^{th} ant from the state from a to b is:

$$\rho_{ab}^{k} = \frac{(Tr_{ab}^{c})(\alpha_{ab}^{d})}{\Sigma(Tr_{ab}^{c})(\alpha_{ab}^{d})} \quad (3)$$

where Tr_{ab} represents the pheromone amount deposited from state 'a' to 'b'. 'c' is used for controlling the impact from Tr_{ab}. α_{ab} denotes state transition desirability. d is for controlling the influence of α_{ab}. When the solution is completed by all the ants, Pheromone updated by the following equation is:

$$\alpha_{ab}^{d} \leftarrow (1 - C_P)\alpha_{ab}^{d} + \sum \Delta\alpha_{ab}^{k} \quad (4)$$

where C_P denotes coefficient of pheromone evaporation. $\Delta\alpha_{ab}^{k}$ represents state intensity of pheromone by k^{th} ant.

CS is a powerful algorithm with the feature of simplicity. It uses a single parameter *pa* with n denotes population size. Implementation of this approach is simple and it is adopted as suitable one when breed behavior to be followed. Thus it is applied in various combinatorial optimization problems. CS provides improved performance when compared with Meta Heuristic algorithms. In CS, each egg represents a solution in the nest. It aims at obtaining the best solution by replacing the solution which are not so-good. The equation of deriving the new solution from the application of random walk and Levy flights is

$$X_{t+1} = (X_t + jG_t) \quad (5)$$

G_t is normal distribution obtained by Levy flights. 'j' represents a fixed number of iterations to determine distant of a random walker.

A novel optimization algorithm is developed by combining benefits from ACO and CS. In ACO, ants move in random directions in order to find food source around the colony. During this process, the ants deposit pheromone on their path. In solving optimization problems, a local search is performed which takes considerably more time. The above draw backs can be overcome by using cuckoo search. CS performs local search in ACO to overcome the above problem. One of the benefits of CS is that, distinct parameter is used apart from population size. In this approach the hybrid algorithm is used to schedule the

task in the available VM in cluster. The following procedure is performed in each cluster to effective schedule the task in available resources.

Step 1: Initialization
Initialize the random nests (R_n)
Pheromone trail, heuristic information (H_{info}),
Step 2: Get Input task from 1 to n
Step 3: Apply transition rules
Step 4: for each task T_1 to T_k do
For each virtual machine Vm_1 to Vm_l do
Assign $T_1 = Vm$
End for
End for
Step 5: perform random walk by cuckoo search from equation 5
Step 6: Pheromone updation
For each pheromone ph_1 to ph_n do
Step 7: Evaluate ρ^{*}_{ab} using equation 3
Step 8: perform the pheromone tail updation by equation 4
Step 9: if current task $\geq$ n task
Then n++ and go to step 2
Step 10: Return value

Figure 1. Pseudo code of ACO and cuckoo search algorithm.

3. Results and Discussion

The experiment is developed in Java language using the Net beans IDE with the hardware configurations: Intel Core2 duo CPU, 3.40 GHz; RAM size, 4GB; Hard disk, 500 GB. Cloudsim simulator tool is used for simulating the virtualized environment and Mysql is used as database. The proposed work is implemented using input files like PDF, Text and Image types. Hybrid ACO and CS with K-means++ algorithm obtains the efficient result with throughput, execution time as performance analysis.

The throughput measures number of tasks accomplished per second. Through this parameter, efficiency of the proposed work can be determined. The parameter such as execution time and the workload are used to measure the throughput.

Table 1. Execution time and throughput from various numbers of files.

No. of Files	Execution Time (Sec)	Throughput (GB)
10	6	0.36
20	6.5	0.39
30	6.9	0.43
40	7.5	0.45

When the number of tasks varies, the difference in execution time and the throughput of hybrid algorithm is shown in Table 1. In hybrid algorithm, the parameters such as bandwidth and number of processors are used to measure the performance of algorithm. Table 2 shows the execution time and throughput of hybrid algorithm with bandwidth and number of processors.

Table 2. Execution time and throughput with different bandwidth and number of processors.

Bandwidth	Number of processors	Execution Time(Sec)	Throughput (GB)
1478.228329	2	8.2	0.48
3391.135942	2	7.9	0.42
2672.404207	3	7.1	0.40
1921.456323	1	8.7	0.57

Figure 2 gives throughput comparison for the hybrid ACO and Cuckoo search with k-means++ and the existing PSO based K-means++ algorithm. The proposed work has throughput of 0.48 GB/per second which is higher than existing technique as shown in Table 3.

Table 3. Performance evaluation on throughput.

Algorithms	Throughput (GB)
ACO and Cuckoo search with K-means++	0.48
PSO based K-means++	0.2

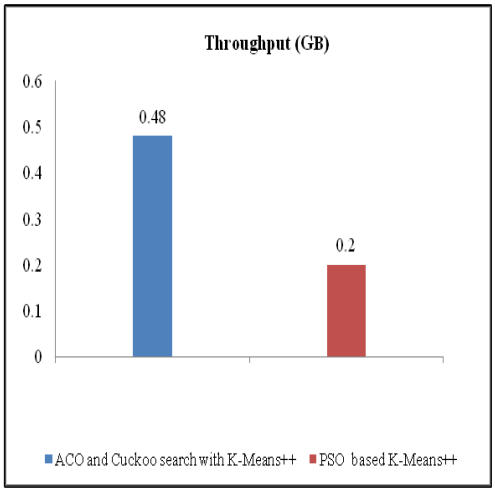

Figure 2. Performance evaluation on throughput.

Figure 3 gives the execution time of hybrid approach compared with existing method. The proposed ACO and

cuckoo with k-means++ obtains 8 seconds whereas PSO based K-means++ obtains 20 seconds is shown in Table 4.

Table 4. Performance evaluation on execution time.

Algorithms	Execution time (Seconds)
ACO and Cuckoo search with K-means++	8
PSO based K-means++	20

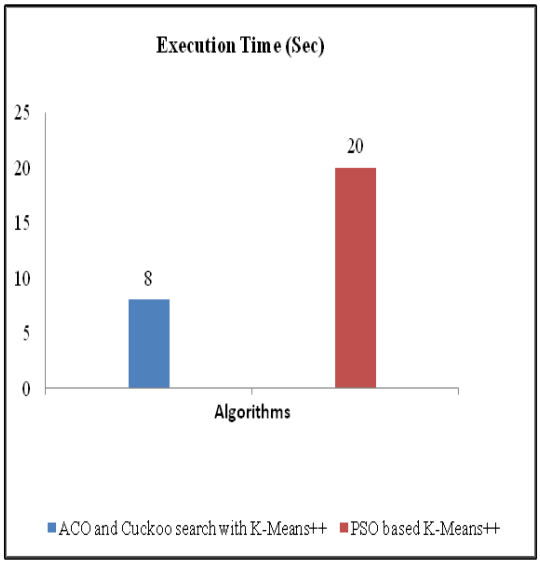

Figure 3. Performance evaluation on execution time.

4. Conclusion

In cloud environment, it is a real challenge to manage and allocate resources among the virtual machines. In order to maintain the better resource utilization, this approach provides a novel task allocation mechanism based on Ant Colony Optimization and Cuckoo Search algorithm with k-means++. An interference prediction scheme is proposed based on Nelder-mead method to utilize characteristics which are collected from an application when running on virtual machines to maintain less system overhead. A hybrid algorithm is adopted for task scheduling process. The CPU utilization, bandwidth, number of processor of the virtual machine are consider for the task allocation to obtain the better resource utilization. The experiment of proposed research work showed that the hybrid ACO and cuckoo search with K-Means++ achieves the better performance than PSO based K-Means++ algorithm. The future work is expected to be in the direction of by designing different scheduling algorithm with various file types.

5. References

1. Tsai Jinn-Tsong, Jia-Cen Fang, Jyh-Horng Chou. Optimized task scheduling and resource allocation on cloud computing environment using improved differential evolution algorithm, Computers and Operations Research. 2013; 40:3045–55.
2. Zhang Y, Franke H, Moreira JE, Sivasubramaniam A. A Comparative Analysis of Space-and Time-Sharing Techniques for Parallel Job Scheduling in Large Scale Parallel Systems, IEEE Transactions on Parallel and Distributed Systems. 2002.
3. Angelou, Evangelos, Konstantinos Kaffes, Athanasia Asiki, Georgios Goumas, Nectarios Koziris. Improving Virtual Host Efficiency through Resource and Interference Aware Scheduling, arXiv preprint arXiv:1601.07400. 2016.
4. Ali AF, Tawhid MA. A Hybrid Cuckoo Search Algorithm with Nelder Mead Method for Solving Global Optimization Problems, SpringerPlus. 2016; 5:473. DOI: 10.1186/s40064-016-2064-1.
5. Bahmani, Bahman, Benjamin Moseley, Andrea Vattani, Ravi Kumar, Sergei Vassilvitskii, Scalable k-means++, Proceedings of the VLDB Endowment. 2012; 5(7):622–33.
6. Rui Han, Junwei Wang, Siguang Huang, Chenrong Shao. Interference-Aware Component Scheduling for Reducing Tail Latency in Cloud Interactive Services, In: 2015 IEEE 35th International Conference on Distributed Computing Systems (ICDCS); 2015. p. 744–45.
7. Wei Zhang, Sundaresan Rajasekaran, Timothy Wood, Mingfa Zhu. Mimp: Deadline and Interference Aware Scheduling of Hadoop Virtual Machines, In: 2014 14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid); 2014. p. 394–403.
8. Chin-Fu Kuo, Chi-Sheng Shih, Tei-Wei Kuo. Resource Allocation Framework for Distributed Real-Time End-To-End Tasks. In: 12th International Conference on Parallel and Distributed Systems; ICPADS 2006.
9. Yu-Chon Kao, Ya-Shu Chen. Data-Locality-Aware MapReduce Real-Time Scheduling Framework, Journal of Systems and Software Volume. 2016; 112; 65–77.
10. Hadi Salimi, Mohsen Sharifi. Batch Scheduling of Consolidated Virtual Machines based on their Workload Interference Model, Future Generation Computer Systems. 2013; 29(8):2057–66.
11. Nelder JA, Mead R. A Simplex Method for Function Minimization, Comput. J. 1965; 7:308–13.
12. Luersen M, Le Riche R, Guyon F. A Constrained, Globalized, and Bounded Nelder–Mead Method for Engineering Optimization, Structural and Multidisciplinary Optimization. 2004; 27:43. Doi: 10.1007/s00158-003-0320-9.

13. Jeffrey C. Lagarias, James A. Reeds, Margaret H. Wright, Paul E. Wright**.** Convergence Properties of the Nelder-Mead Simplex Algorithm in Low Dimensions, SIAM J. Optim. 1998; 9(1); 112–147.