Hybrid of Meta Heuristic Firefly and Genetic Algorithm for Optimization Approach in the Cloud Environment

C. Vinothini
Department of Computer Science & Engineering, Dr.N.G.P Institute of Technology, Coimbatore, India.
E-mail: vinuchidambaram@gmail.com

P. Balasubramanie
Department of Computer Science & Engineering, Kongu Engineering College, Perundurai, India.
E-mail: pbalu_20032001@yahoo.co.in

J. Priya
Department of Information Technology, Bannari Amman Institute of Technology, Sathyamangalam, India.
E-mail: priyaajothimani@gmail.com

Received March 22, 2020; Accepted May 20, 2020
ISSN: 1735-188X
DOI: 10.14704/WEB/V17I1/WEB17005

Abstract

Clouds are strengthening upcoming-generation storage systems as a virtual processing communication infrastructure. Cloud infrastructure resource allocation of processes is an essential component of allocating resources from either a cloud server. Utilization and distribution of resources will be as per Service Level Agreement (SLA) Quality of Service (QoS) to significantly reduce energy consumption. Demands for the cloud system services are simultaneous and efficient. Convenient utilization of resources is a basic requirement in the cloud server and can be carried out by introducing an automated and efficient optimization algorithm. Optimization is a scientific principle for finding the best solution to the problems through possible solutions. Therefore, plenty of the suggested algorithms concentrate on the quest for estimated VM scheduling algorithms solutions. Methodology, universe-heuristic and hybrid optimization strategy increasing in popularity as finding optimized solutions to critical issues in a timely manner. A hybrid meta-heuristic algorithm is recommended employing modified Firefly Optimization Algorithm (MOFA) and Genetic Algorithm techniques inspired by the natural. Behavior and performance result indicates through introduced solution tested on the cloudsim simulation platform.

Keywords

Cloud Computing, Hybrid Optimization, Load Balancing, Genetic Algorithm, Modified Firefly Optimization Algorithm (MFOA), Waiting Time, QoS.

Introduction

Virtualization, service-oriented architecture (SOA) and grid computing play a significant role in building a cloud computing ecosystem. Cloud computing seems to be a possibly the best-known technology, that uses the fee-per-use model to provide all the requisite services on request [1, 2, 3]. In cloud technology, one of the significant and important research area is balancing the load between virtual machines to distribute at data server. Cloud includes a large number of multiple and diverse closely linked devices, in which all data and services are stored. VMs are really the cloud infrastructure solutions that simulate and share information efficiently through execution of tasks whenever needed. A huge number of virtual machine are interconnected in the remote cloud maintaining the services secured manner, as a consequence services are not dispersed uniformly and maybe some virtual machines may not have the opportunity to obtain the services [2, 3]. The optimization tasks can be reversible or irreversible on the bases of parameters. Heuristic: probabilistic conditions are developed for getting a suitable solution to a real problem [4]. Selected conditions rely on the problems and are designed in a shorter period to work out a solution. Heuristic methods consider a cost-effective and appropriate method with lesser time period. Meta-heuristic: usually, meta-heuristic algorithms are constructed and simulated for specific problems [4]. Therefore, meta-heuristic maintains produce consistent for the layout of complex problems.

Hybrid: Algorithm homologous recombination could be based on many factors to identify the objective function, like applying the optimization technique to the basic Virtual machine allocation and then applying the meta-heuristic optimization to improve the location of virtual machines in effort to reduce migrating or conversely. Interbreeding offers a much more effective solution, as it can be implemented in any sequence, in recognized as an expense, times and solution space [5]. Conversely, the difficulty of deployment is enhanced.

The rest of this article is read out as mentioned. Second section calls for a study of the status of the arts for load balancing. Description of problems and fitness value formulation is described in section 3. Section 4 introduces the development of a proposed solution of load balancing in cloud computing.

Related Works

In [6] suggested a novel task scheduling methodology to a basic Genetic Algorithm (GA) to reduce the number of activities required in transmission and energy performance. This

argues that differences in genetic algorithms would provide more competitive outcomes in the end.

In [7] suggested a timing system based on the evolutionary algorithms. Historical information and contemporary characteristics of the system always had to determine the impact on the organization in detail while Virtual machines provider assets was implemented on the computer cluster and way to solve mostly with process which identifies effects to fulfill the query. This method achieves good load balance and minimizes flexible flow of migrants or prevents.

In [8] suggested a novel metaheuristic distribution of Ant colony optimization-VMM. The use of resources is provided separately with the aid of a relevant mobility consultant. Utterly pointless movements are minimized by careful control of official and unofficial situations. Two ants navigating approach taken to look for the near-optimal mapping through physical and the virtual systems.

An Enhanced Ant Colony Optimization was suggested by [9] to efficiently balance the workload in cloud computing. Forward- reverse agent migration framework and super-min strategy can be used to rapidly scan the member domains for task scheduling. Research reveals searches conducted with less time and with strong network efficiency in heavy and light weights.

In [10], presented a probabilistic ACO for cloud-based service load capacity. Technique of pollen updating provides effective load balancing and minimizes the make pan. The overall performance of CPU usage, space and networking load is factored into the equation with the question of load balancing.

Fazil et al [11] developed a novel methodology to address the problems of optimization. Through this, the crossover processes are performed with changing stage of firefly position, which enhances computation time and efficiency. The efficiency of the proposed method is compared with the conventional genetic algorithm, convergence rate optimization of ant colony, and various minimization or maximization operations.

Vinothini [12] et al suggested a novel approach for load balancing multi-servers employing meta-heuristic firefly algorithms. Here the firefly algorithm was changed to position the servers in an optimized way. The performance assessment is contrasted with other optimization algorithms in terms of cost of communication and the usage of resources.

Multi-objective Ant Colony System was introduced by Ashraf et al.[13] using a multi - criteria, multicolony method by promoting solitary-objective, solitary-colony ACO algorithms. The suggested solution removes the composite location – based and the purposes of improvement together in the appropriate fashion.

Singh et al. [14] suggested mixed meta-heuristic programming of VM employing ACO and Particle Swarm. Knowledge of the subject forecasts the difficult world for the new signing assignment. Demands Dismissal made prior to preparing that cannot meet the measurement time reducing requirements. Early approach identified through ACO and afterwards implemented by Optimization to boost efficiency using only strongest sustainable solution rather then the specific one.

Thiruvenkadam et al. [15] suggested a plan for hybrid genetically engineered VMs to mitigate waves of immigration. Primary concern issued on parameter host demands and dynamics of VM allocation. Probabilistic approach (i.e. greedy algorithm) to ensuing VM loading. After optimization made for GA-placed virtual machines according to weight vector.

Proposed Methodology

Genetic Algorithm (GA), excellently-known evolutionary algorithms search-based optimizing method is proposed based on the genetic factors and natural processes standards. In AI technologies, the evolutionary principle of the optimization algorithm for initial population generation is becoming trendier. Using a directed spontaneous search process, this genetic method creates optimized and best quality solution to the crisis.

In the very first step of the suggested meta-heuristic GA, optimal solution for n/2 variations were implemented from created population. As in GA strategies the pair of genotype of the same length is specified. The genome will become more strength at every execution as it crosses underneath the genetic algorithm shown in figure 1. The parents will be selected based on sampling process.

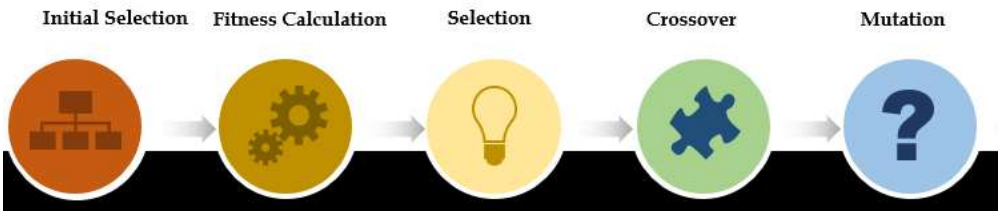

Figure 1 Operational Flow of Genetic Algorithm

Crossover feature introduced on picked parents one-point pairs to create new children. Frame-flip mutation task completely random introduced in state space to prevent the adhesiveness of random search engines. When a mutation creates advanced functionality in the populations and explores the latest search space choices.

Modified Firefly Optimization algorithm is already proposed by us in [12]. There, running meta-heuristic method is used to conduct optimized load balancing so that the server consumption can be handled effectively to maintain an efficient mission. FA can cope spontaneously and effectively with extremely variation, multidimensional optimization. This method intends to minimize the task scheduling processing time (Ri) at each application. In order to solve multiple goals simultaneously through firefly optimizing, it uses two fitness functions, namely load balancing and span making

1) Hybrid Mechanisms

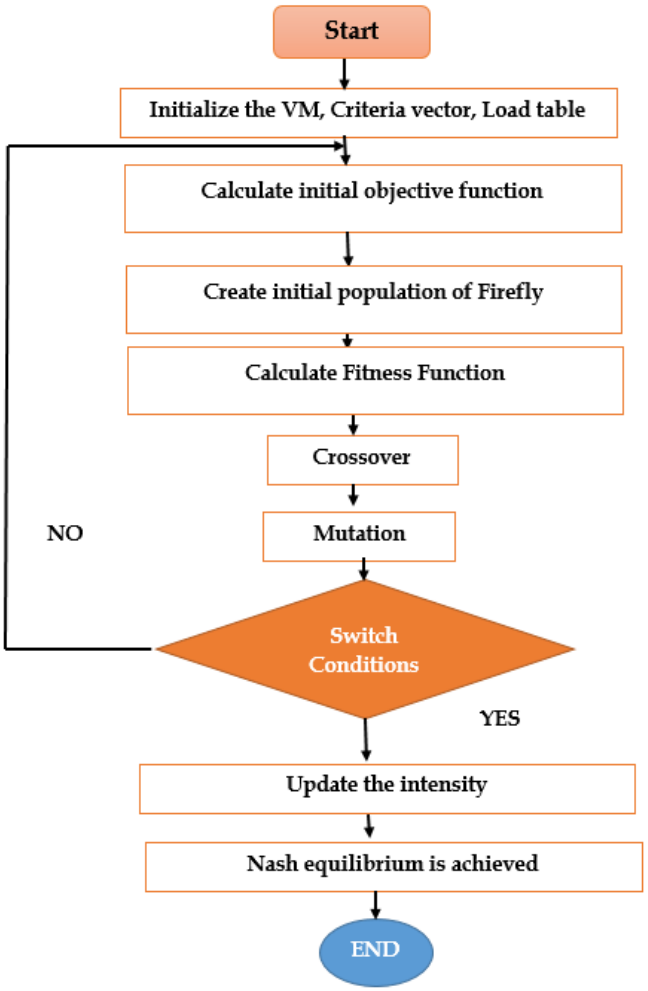

Fig. 1 The Architecture of Proposed method

Research studies indicated that modified Firefly should acquire convergence speed in successive iterations could be due to the absence of complexity. Predictions often demonstrated that the amount of a suitable agent could substantially accelerate the Firefly output. Therefore, there will be multiple options to enhance Firefly efficiency first to stimulate productive population through various configuration arrangements and secondly to maintain the variety to lead the swarm

Algorithm: Hybrid Optimization algorithm

Initialize Load Table ()

Firefly= getFirefly(vm)

if(Firefly==null) then

neededcloudServer = get needed CloudServer For Vm (CloudServer Index, vm)

Calculate Objective function

Firefly = novel Firefly(vm, needed cloudServer)

Firefly Group.add (vm, Firefly)

fori = 1: n all n fireflies

forj = 1: i all n fireflies

Calculate fitness function

Perform Crossover;

Perform Mutation;

Estimate resolutions and update light intensity;

Simulation Settings

Obtained from the simulation test, the evaluation of the proposed method has been examined. The cloud services research was carried via the CloudSim3.0.3 emulator and running on the computer with Intel core i5 processor, 8 GB RAM, 3.4 GHz CPU and Window 7 operating system.

1) Performance Evaluation

In this section, the performance evaluation of the proposed method is compared with traditional genetic algorithm, firefly and modified firefly method in terms of communication cost, running time and resource utilization based on the number of tasks.

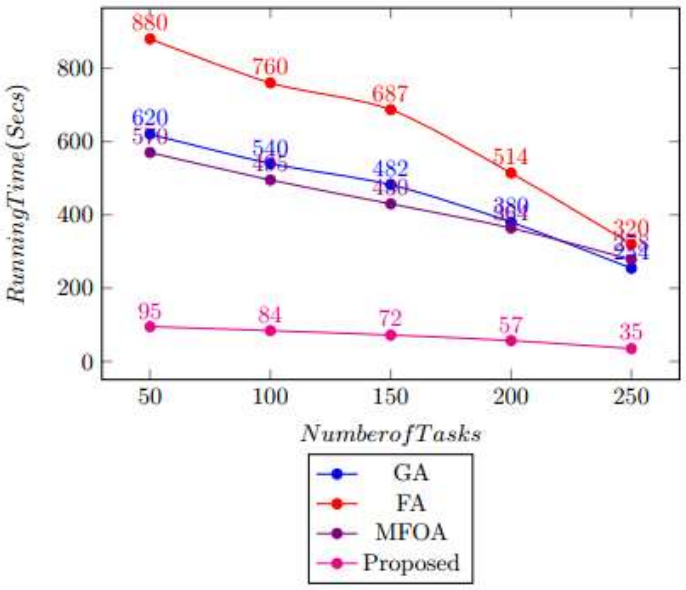

Fig. 2 Comparison of Running time

By Fig. 2, it could be interpretive that the total time comparison using predominant GA, FA, MFOA and proposed strategies is expected. The methods and the cumulative time are alternatively displayed on the x-axis and y-axis. It demonstrates that the predominant GA, FA, MFOA algorithm requires more time whereas the proposed method requires less time.

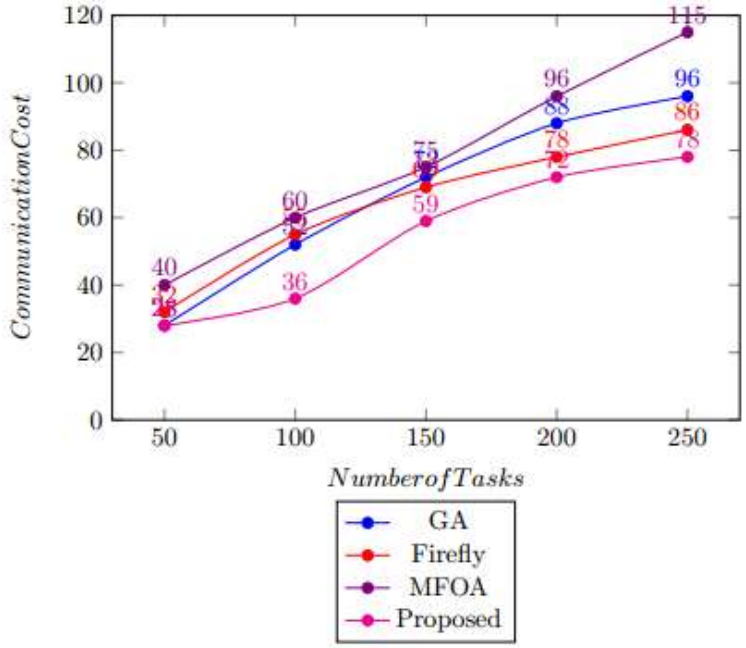

Fig. 3 Comparison of Communication Cost

By Fig. 3, comparison of contact costs using GA, FA, MFOA and proposed techniques can be reported. The methods and contact costs are shown, respectively, on the x-axis and y-axis. This reveals that the GA, FA, MFOA methodology requires higher cost of communications, whereas proposed methods require lower cost compared than GA, FA, and MFOA.

By Fig. 4, comparison of resources usage using GA, FA, MFOA and proposed approaches awaited. The methods and use of resources are shown on the x-axis and y-axis and it can be analyzed accordingly. This demonstrates that GA, FA and MFOA offers less resource usage whereas proposed method offers higher resource utilization.

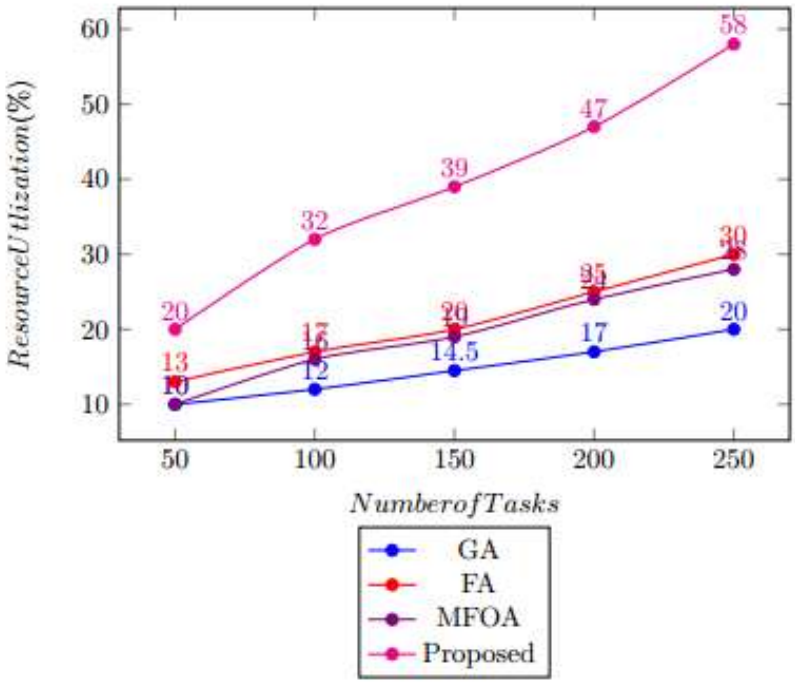

Fig. 4 Comparison of Resource Utilization

Conclusion

In this article the hybrid meta-heuristic methodology like MOFA + GA was introduced for task scheduling in the cloud computing platform for multiple tasks. The developed system provides a distributed by reallocating the loads to the related VMs, taking into account the objective function of VM. The developed system also increases the resource utilization and communication cost during task scheduling and efficiently decreases the processing time of the process comparison to different techniques such as GA, FA and MFOA.

References

Buyya R, Broberg J, Goscinski AM. Cloud computing: Principles and paradigms, John Wiley & Sons 2011.

Chinnasamy P, Deepalakshmi P. A scalable multilabel-based access control as a service for the cloud (SMBA CaaS). Transactions on Emerging Telecommunications Technologies 2018; 29(8): e3458.

Chinnasamy P, Deepalakshmi P. A Survey on Enhancing Cloud Security through Access Control Models and Technologies. International Journal on Computer Science and Engineering (IJCSE) 2017; 9(5): 326-331.

Talbi EG. Metaheuristics: from design to implementation, John Wiley & Sons 2009.

Mishra SK, Sahoo B, Parida PP. Load balancing in cloud computing: a big picture. Journal of King Saud University-Computer and Information Sciences 2020; 32(2): 149-158.

Dasgupta K, Mandal B, Dutta P, Mandal JK, Dam, S. A genetic algorithm (ga) based load balancing strategy for cloud computing. Procedia Technology 2013; 10: 340-347.

Kumar A, Dadheech P, Singh V, Raja L, Poonia RC. An enhanced quantum key distribution protocol for security authentication. Journal of Discrete Mathematical Sciences and Cryptography 2019; 22(4): 499-507.

Wen WT, Wang CD, Wu DS, Xie YY. An ACO-based scheduling strategy on load balancing in cloud computing environment. In Ninth International Conference on Frontier of Computer Science and Technology 2015: 364-369.

Gao R, Wu J. Dynamic load balancing strategy for cloud computing with ant colony optimization. Future Internet 2015; 7(4): 465-483.

Mishra R, Jaiswal A. Ant colony optimization: A solution of load balancing in cloud. International Journal of Web & Semantic Technology, 2012; 3(2).

Wahid F, Alsaedi AKZ, Ghazali R. Using improved firefly algorithm based on genetic algorithm crossover operator for solving optimization problems. Journal of Intelligent & Fuzzy Systems 2019; 36(2): 1547-1562.

Vinothini C, Balasubramanie P. Meta-heuristic firefly approach to multi-servers load balancing with independent and dependent server availability consideration. Journal of Ambient Intelligence and Humanized Computing 2020: 1-13.

Ashraf A, Porres I. Multi-objective dynamic virtual machine consolidation in the cloud using ant colony system. International Journal of Parallel, Emergent and Distributed Systems 2018; 33(1): 103-120.

Singh AK, Nasiruddin I, Sharma AK, Saxena A. Implicit control of eddy current braking system using fuzzy logic controller (FLC) and particle swarm optimisation (PSO). Journal of Discrete Mathematical Sciences and Cryptography 2019; 22(2): 253-275.

Thiruvenkadam T, Kamalakkannan P. Energy efficient multi dimensional host load aware algorithm for virtual machine placement and optimization in cloud environment. Indian journal of science and technology 2015; 8(17): 1-11.

Yoganingrum A. How to make biodiversity articles highly cited?. Webology 2020; 17 (1) (A211): 109-129.

Hazar MJ, Shaker BN, Ali LR, Alzaidi ER. Using received strength signal indication for indoor mobile localization based on machine learning technique. Webology 2020; 17 (1) (206): 30-42.