Journal of Information & Optimization Sciences
ISSN 0252-2667 (Print), ISSN 2169-0103 (Online)
Vol. 44 (2023), No. 6, pp. 1023–1033
DOI : 10.47974/JIOS-1226

Task scheduling in cloud-fog computing using discrete binary particle swarm meta-heuristic with modified sigmoid function

Y. Nasir Ahmed *
S. Pakkir Mohideen †
Department of Computer Applications
B. S. Abdur Rahman Crescent Institute of Science and Technology
Vandalur, Chennai
Tamil Nadu 600048
India

Mohammad Pasha §
Department of Information Technology
Muffakham Jah College of Engineering and Technology
Hyderabad
Telangana (TS) 500034
India

Abstract

Cloud-Fog IoT networking is a resourceful technology to aid the processing of IoT end device requests. These devices generate tasks that need optimized computing and reduced latency for applications that operate in real-time environments. This article sets forth a cloud-fog task scheduler that schedules diverse tasks on vertically scaled Cloud and Fog virtual machines. For the proposed Binary Particle Swarm Optimizer (BPSO) based scheduler, an apt choice identified is to employ a modified sigmoid function with the logarithm decreasing inertia weight policy to deliver an optimal scheduling scheme. Moreover, the parameters of the BPSO are tuned inferring the best practices prescribed in literature. The results show that proposed method caters better than existing heuristic techniques to improve makespan and load imbalance.

Subject Classification: *Primary 68T20, Secondary 68M20.*

Keywords: *Discrete binary particle swarm algorithm modeling, Meta-heuristic algorithms modified sigmoid function, Logarithm decreasing inertia weight.*

* *E-mail:* **nasir_ca_phd_18@crescent.education (Corresponding Author)**
† *E-mail:* **pakirmoitheen@crescent.education**
§ *E-mail:* **mdpasha@mjcollege.ac.in**

© Taru Publications

1. Introduction

The type and amount of IoT tasks generated change dynamically and the interactions are far from being predictable. Cloud services provided users with virtual servers through virtual machines (VMs). Over the years to cut costs, the demand for these costly energy-mongering virtual servers has transformed into calls for cheap energy-saving, narrow function-based modules or micro-services [1]. The idea of moving the micro-services to the edge devices helps and complements the vision of green computing. This further supplement the needs arising from real-time or near real-time applications that have emerged due to the connectivity boom as part of a growing need for a smart-eco system [2].

As the number of these diverse tasks (compute-intensive and low latency) increase so will be the number of VM requests. Thus, vertical scaling of VMs to handle these requests efficiently is needed. This forms the problem of scheduling a large group of diverse tasks on an array of heterogeneous VMs. Further the time taken to evaluate the best (task, VM) pair upsurges rapidly. For such mapping problems, evolutionary computational algorithms can attain optimal or near-optimal resolutions in a viable time. Hence these are part of the selected choice for such computationally hard (NP-hard) problems [3].

In this research, a meta-heuristic task scheduler namely Binary PSO-based scheduler [4] with improved load balancing strategies (namely IBPSO and MIBPSO) is proposed to optimize makespan and reduce load imbalance. The experimented heuristics are implemented to evaluate the task assignment and scheduling methods in cloud and fog computing environments.

The subsequent sections of the present work are organized as follows: in Section 2, a summary of related works in heuristic task scheduling algorithms. Section 3 briefs the scheduling problem and its formulation with respect to BPSO and improvised algorithms. Section 4 presents the Simulation setup and various scenarios implemented for cloud and fog environments. The penultimate Section 5 presents the analyses of results and observations. To conclude, Section 6 covers-up the theme of the work.

2. Related Works

The time sensitive services deployed on the cloud computing requires fastest turnaround time from the datacenter to IoT device and vice versa and has been attained by the fog computing paradigm [1].The proposed

fog-cloud system has enabled critical heart beat tasks medical applications to collect and analyze so that overall cost may be reduced subsequently [4]. Particle Swarm Optimization is projected as optimal solution towards scheduling workflows for scientific and also the web applications [5]. Recommended multi-objective task-scheduling problem which optimizes the two parameters in fog-cloud system first is the makespan and the second is total costs [6]. Maximum usage of fog computing by subsequent decrease in response time and also decreasing the cost towards cloud compute resources is addressed [7].

Network delay, cost, processing time are offered in this paper to address and meet the deadline of the deployed resources on fog-cloud setting [8]. Classifier method is used to impart the workloads with respect to power consumption and delays at the edge of the networks in fog-cloud systems [9]. Task imbalance at fog nodes is addressed by implementing two algorithms and colony and particle swarm optimization [10]. Effective and managed energy consumption is attained by even distribution of tasks at the nodes in fog-cloud system which gives reduced latency and increased quality of service [11]. Usage of Harris Hawks's optimization algorithm enhanced the quality of service for Industrial IoT and fog computing users through cloud computing [12]. Heuristic approach to place various applications on the fog-cloud system to acquire the low latency and low energy consumption. [13]. Heuristic task scheduling which is of three staged is recommended and attained the best results towards performance in defined service level agreements [14].

Computational nodes received many healthcare tasks to both cloud and fog devices in order to reduce the schedule time in-turn usage of task features, thus attaining maximum mobility [15]. Maximum profit is offered at the integrated fog-cloud system by applying profit enhancement algorithm to the smart systems [16]. A cost effective blockchain based fog cloud is proposed to enhance the visibility of the patient's health problems and reduces their charges as the processing is done in the system [17].

3. Discrete Binary Particle Swarm Meta-Heuristic with Modified Sigmoid Function and Logarithm Decreasing Inertia Weight.

Particle swarm meta-heuristic, i.e. PSO is a remarkable technique that solves a numerically hard polynomial problem by intelligently repeating steps to improvise a candidate solution, making it approach a well-defined fitness measure. The Adapted form of this is what is dubbed as its Binary counterpart, named as BPSO, in short. The BPSO provisions solutions for

practical discrete optimization problems [4]. The significant distinction of the two forms is that the velocities and positions of the individual solutions are characterized as far as the progressions of probabilities and the particles are framed by zero or one. The velocity of the particle needs to be constrained to the interval [0, 1). A calculated sigmoid transformation can be utilized to restrict the velocity of a particle.

3.1 *Task Scheduling in Cloud-Fog systems*

The system consists of independent tasks (T) whose requested computation size and data size is specified in million instructions and bytes respectively. These tasks are to be mapped to Virtual Machines (VMs) whose Processing power as well as bandwidth, specified in million instructions per second (MIPS) and Megabytes per second (MBps) respectively vary according to where they are located. The Task scheduler first filters tasks based on suitability for execution on Cloud or Fog Nodes as shown in Fig. 3.1.

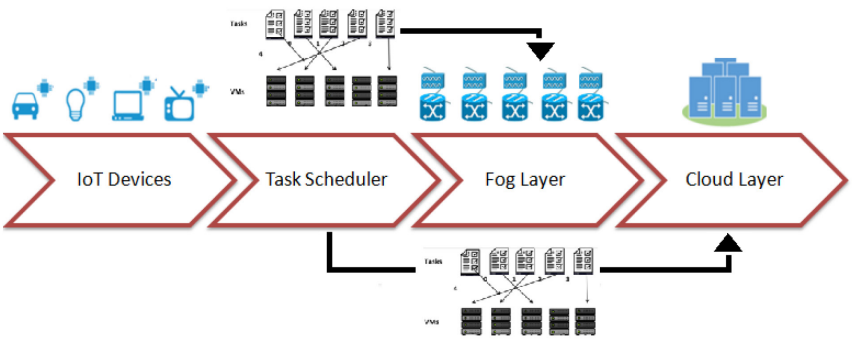


Fig. 3.1

Architectural Elements of Cloud-Fog Computing

Table 1.1

Symbols used in the problem formulation and implementation.

$R(T_i, VM_j)$	Runtime for the task t_i when executed on VM_j
T_l	Task length in million instructions (mi)
VM_{npe} *is*	Number of processing elements in VM
VM_{mips}	Processing power of each core in VM
T_{fs}	File-size to be sent for processing the task

Contd...

VM_{bw}	Bandwidth of the VM
$CT(VM_i)$	Completion time of all tasks assigned to VM_i
$P\{i, j\}$	Particle position
$V\{i, j\}$	New velocity of particle
ω, ω_{Max}, ω_{Min}	Inertia weight calculated using linearly decreasing property, Maximum and Minimum inertia weight resp.
$iter_{Max}$	Maximum number of iterations specified
$iter_{Curr}$	Current iteration
$c1, c2$	Cognitive/local coefficient and Social/global coefficient resp.
$r1, r2$	Random numbers in the interval [0, 1)
$lbp\{i, j\}$	Local best position value of a particle in the current iteration
$gbp\{i, j\}$	Global best position value of a particle across all iterations

Algorithm1: Binary PSO pseudocode

Algorithm 1: Discrete Binary Particle Swarm Optimization

Input: Task set, VM set
Output: VM to Task Mapping
Data: Task set, VM set, Particle set P

```
Evaluate the fitness function
for i=1,... ,Max Iteration do
    for each particle in P do
        Create individual particles with positions, velocities, best positions
        Update local best particle positions and velocities using fitness
         function
        Update global best particle using local best particles
for i=1,... ,VM set size do
    for j=1,... ,Task set size do
        if partilce position value is set to 1 then
            Assign VMi to execute the task Tj
return VM to Task Mapping
```

With respect to the BPSO algorithm given in Algorithm 1, whose parameters are defined in Table 1.1, initially we calculate the running times of all these tasks on all VMs. The resultant matrix of all tasks mapped to all VMs is termed as the completion time matrix. The task completion time is calculated using the details of each Task-VM pair. It is given by the following equation

$$R(Ti,VMj)=\left(\frac{Tl}{(VMnpe*VMmips)}+\frac{Tfs}{VMbw}\right) \quad \text{Equation (1)}$$

Further the completion time of all tasks assigned to a VM can be obtained using the following

$$CT(VMj)=R(Ti,VMj)*P\{i,j\} \quad \text{Equation (2)}$$

P {i, j} denotes the particle position which is a binary value of either 1 (if selected task is assigned to the particular VM) or 0 (if selected task is not assigned to the particular VM).

The maximum of the completion time on a VM gives the makespan for executing all tasks assigned across all VMs. Thus, Makespan is given by

$$Makespan(T,VM)=Max(CT(VMi)) \quad \text{Equation (3)}$$

The positions and velocities of all particles of PSO are initialized by calling the CreateParticles () method (Algorithm 1 Line 4). Here the position values are assigned a binary value of either 1, if the selected task is assigned to the particular VM) or 0 otherwise. The velocity is a uniform distributed random number in [0,1). These positions are updated to the best positions by comparing the value obtained from the fitness function. Next the particles positions and velocities are updated (Algorithm 1 Line 5) using the PSO equations as following

$$V\{i,j\}=(\omega*V\{i-1,j-1\})+(c1*r1*(lbp\{i,j\}-p\{i,j\})) + (c2*r2*(gbp\{i,j\}-p\{i,j\})) \quad \text{Equation (4)}$$

$$\omega=\omega Max-(\omega Max-\omega Min)*(iterMax-iterCurr)/iterMax) \quad \text{Equation (5)}$$

$$P\{i,j\}=\left(\frac{1}{(1+exp(-V\{i,j\}))}\right)=0\Leftrightarrow\left(\frac{1}{(1+exp(-V\{i,j\}))}\right)<r\{i,j\} = 1\Leftrightarrow\left(\frac{1}{(1+exp(-V\{i,j\}))}\right)\geq r\{i,j\} \quad \text{Equation (6)}$$

The update of local best particle position is compared for all particles from the current iteration. As shown in Fig.3.2, we modified the sigmoid function to reflect on the cognitive/local coefficient and social/global coefficients. Following this, the global best particle positions are obtained by comparison across all iterations. The VMs for which the position values are set are selected for allocation of tasks. To enhance the solutions

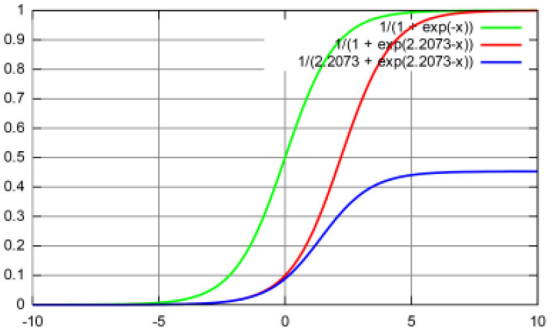


Fig. 3.2

Modified Sigmoid Functions used to reflect on the cognitive/local coefficient and social/global coefficients of BPSO.

obtained by BPSO, we use a rebalancing method to identify the heaviest loaded VMs in the solution set. We swap the assigned tasks from these with the tasks from the lightest loaded VMs where the reassignment outputs a reduced task completion time, hence improving the makespan.

4. Experimental Setup

The Experiments were carried on a PC with Intel Core i7-3770 CPU, 3.9 GHz, Memory 16 GB, Cloudsim 3.0 simulator on Windows OS.
The Virtual Machine configuration of Cloud (CVM) and Fog (FVM) nodes is as follows

$$x1 = CVM, 9k \leq pp(x1) \leq 10k\ , 1k \leq bw(x1) \leq 2k;$$
$$x2 = FVM, 1k \leq pp(x2) \leq 2k\ , 9k \leq bw(x2) \leq 10k$$

with pp(x) is in MIPS, bw(x) is in MBps units. The Task configuration for Cloud (CT, CW) and Fog (FT, FW) is

$$cb = CT, 9k \leq ts(cb) \leq 2k\ , 1k \leq fs(cb) \leq 2k;$$
$$fb = FT, 1k \leq ts(fb) \leq 2k\ \ 9k \leq fs(fb) \leq 10k;$$
$$cw = CT, 9k \leq ts(cw) \leq 2k\ \ 9k \leq fs(cw) \leq 10k;$$
$$fw = FT, 9k \leq ts(fw) \leq 2k\ , 9k \leq fs(fw) \leq 10k$$

with ts(x) defined in MI, fs(x) is in Bytes. Table 4.1 presents BPSO and Modified sigmoid transformation parameters used in experiments.

Table 4.1

The BPSO parameters and their corresponding values

Algorithm	Sigmoid	Particles	Rounds	c1, c2	W
BPSO	$\left(\frac{1}{(1+\ \exp(-V\{i,j\}))}\right)$	10	100	1.49, 1.49	04-0.9
MBPSO-1	$\left(\frac{1}{(1+\ exp(2.2073-V\{i,j\}))}\right)$	10	100	2.2, 2.2	-0.434
MBPSO-2	$\left(\frac{1}{(2.2073+\ exp(2.2073-V\{i,j\}))}\right)$	10	100	-2.2, 2.2	-0.434

5. Experimental Evaluation and Results

The Task count is varied as 100, 500, 1000 and 2000 tasks. The VM number is varied as 10, 25 and 50 VMs. The experiments to evaluate the makespan and load imbalance were performed for (i) Min-Min heuristic, (ii) BPSO (Binary PSO), (iii) MBPSO (Modified Binary PSO), (iv) IBPSO (Improved Binary PSO) and (v) MIBPSO (Modified Improved Binary PSO)

Fig. 5.1 plots the makespan of one heuristic (Min-Min) and four meta-heuristic algorithms namely BPSO, IBPSO, MBPSO, MIBPSO under

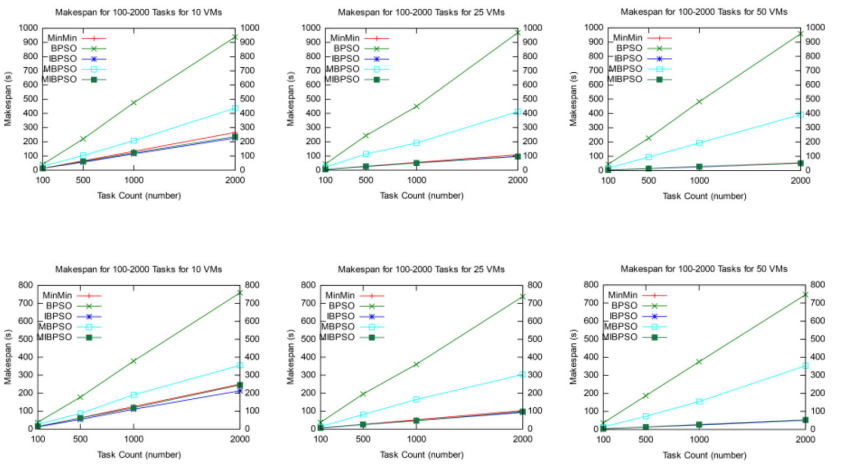


Fig. 5.1

Makespan comparison for implemented algorithms with varying Task and VM Count in Cloud and Fog scenarios

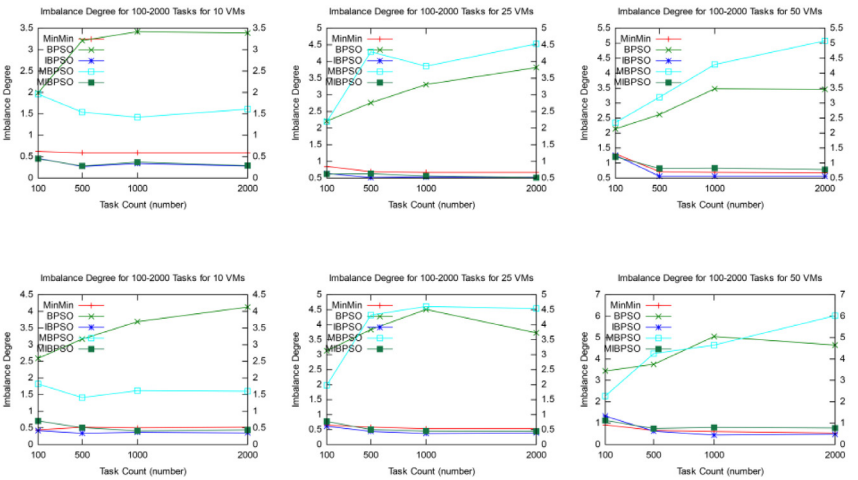


Fig. 5.2

Degree of Imbalance comparison for implemented algorithms with varying Task and VM Count in Cloud and Fog scenarios

varying Task and VM count. Top row plots depict Cloud task scheduling for 10, 25 and 50 VM count (let-center--right) whereas bottom row is for Fog task scheduling similarly. With the increase in VM count, the makespan of all scheduling algorithms decreases when load balancing is applied. The reason is that, at VM count 10, the tasks starting from task 11 are mapped to VMs that are already allocated. The number of free VMs increases for a given set of tasks as the VM count increases. Moreover, it can be seen that the MBPSO outperforms BPSO producing nearly 50% lesser makespan for the same set of tasks and VMs. The improved versions IBPSO, MIBPSO further improve makespan to 80% or more.

Fig. 5.2 plots the degree of load imbalance on VMs produced by various algorithms in a Cloud and Fog scenarios. Top row plots depict Cloud task scheduling for 10, 25 and 50 VMs respectively, whereas bottom row is for Fog task scheduling. Load imbalance increases with the increase in VM count. The degree of imbalance is reduced with IBPSO and MIBPSO algorithms due to application of rebalancing the solution obtained by BPSO and MBPSO algorithms.

6. Conclusion

This article presented implementation of Discrete Binary PSO and Modified Discrete Binary PSO algorithms to address efficient task

scheduling in cloud and fog deployment scenarios. To enhance the result further, the selected algorithms were load balanced. The relative improvement of makespan achieved across all scenarios is in the range of 80% to 93% for VM count of 50. The proposed version of BPSO algorithm can perform faster when compared to Original BPSO for similar settings in all performed experiments. It demonstrates the effect of tuning the BPSO parameters to reflect in customizing the sigmoid transformation. In the future work other meta-heuristic algorithms will be tested to enhance makespan, load balancing and other metrics.

References

[1] Aburukba, Raafat O., TahaLandolsi, and Dalia Omer, A heuristic scheduling approach for fog-cloud computing environment with stationary IoT devices, *Journal of Network and Computer Application*, 180 (2021): 102994.

[2] Kratzke, Nane. "A brief history of cloud application architectures." Applied Sciences 8.8 (2018): 1368.

[3] Mastoi, Qurat-ul-ain, et al, A novel cost-efficient framework for critical heartbeat task scheduling using the Internet of medical things in a fog cloud system, *Sensors* 20.2 (2020): 441.

[4] Poli, Riccardo, James Kennedy, and Tim Blackwell. "Particle swarm optimization." Swarm intelligence 1.1 (2007): 33-57.

[5] Farid, Mazen, et al, A survey on QoS requirements based on particle swarm optimization scheduling techniques for workflow scheduling in cloud computing, *Symmetry* 12.4 (2020): 551.

[6] Ali, Ismail M., et al, An automated task scheduling model using non-dominated sorting genetic algorithm II for fog-cloud systems, *IEEE Transactions on Cloud Computing* (2020).

[7] Mokni, Marwa, et al,Cooperative agents-based approach for workflow scheduling on fog-cloud computing, *Journal of Ambient Intelligence and Humanized Computing* (2021): 1-20.

[8] Naha, Ranesh Kumar, et al, Deadline-based dynamic resource allocation and provisioning algorithms in fog-cloud environment, *Future Generation Computer Systems* 104 (2020): 131-141.

[9] Abbasi, Mahdi, et al, Efficient resource management and workload allocation in fog–cloud computing paradigm in IoT using learning classifier systems, *Computer communications* 153 (2020): 217-228.

[10] Hussein, Mohamed K., and Mohamed H. Mousa, Efficient task offloading for IoT-based applications in fog computing using ant colony optimization, *IEEE Access* 8 (2020): 37191-37201.

[11] Kaur, Mandeep, and Rajni Aron, Energy-aware load balancing in fog cloud computing, *Materials Today: Proceedings,* (2020).

[12] Abdel-Basset, Mohamed, et al, Energy-aware metaheuristic algorithm for industrial-Internet-of-Things task scheduling problems in fog computing applications,*IEEE Internet of Things Journal,* 8.16 (2020): 12638-12649.

[13] Natesha, B. V., and Ram Mohana Reddy Guddeti, Heuristic-based IoT application modules placement in the fog-cloud computing environment, *2018 IEEE/ACM international conference on utility and cloud computing companion (UCC Companion)*, IEEE, 2018.

[14] Sang, Yongxuan, et al, A three-stage heuristic task scheduling for optimizing the service level agreement satisfaction in device-edge-cloud cooperative computing, *PeerJ Computer Science,* 8 (2022): e851.

[15] Abdelmoneem, Randa M., Abderrahim Benslimane, and EmanShaaban, Mobility-aware task scheduling in cloud-Fog IoT-based healthcare architectures, *Computer Networks,* 179 (2020): 107348.

[16] Mahmud, Redowan, et al, Profit-aware application placement for integrated fog–cloud computing environments, *Journal of Parallel and Distributed Computing*, 135 (2020): 177-190.

[17] Lakhan, Abdullah, et al, Smart-contract aware ethereum and client-fog-cloud healthcare system, *Sensors,* 21.12 (2021): 4093.