GWO Based Task Allocation for Load Balancing in Containerized Cloud

Dimple Patel
Dept. of Computer Science & Eng
National Institute of Technology
Rourkela, India
dimplepatel982@gmail.com

Manoj Kumar Patra
Dept. of Computer Science & Eng
National Institute of Technology
Rourkela, India
manojpatra.in@gmail.com

Bibhudatta Sahoo
Dept. of Computer Science & Eng
National Institute of Technology
Rourkela, India
bibhudatta.sahoo@gmail.com

***Abstract*—On-demand provisioning of computing services such as analytics, intelligence, networking, storage, and servers, etc. over the internet is the main function of cloud computing. Several servers are connected in a distributed manner over the internet to execute tasks. Recently, container technology has gained enormous popularity as it can improve overall application performance by providing OS-level virtualization in cloud computing systems. Based on the resources available on server, a server can accommodate more than one container running on it. The process of distributing the incoming requests or user tasks among all available servers in such a way that all the servers will have almost equal workload is called load balancing. In this paper, we proposed a Grey Wolf Optimization(GWO) based technique for load distribution in the containerized cloud and also to reduce the makespan. We have compared our results with the Genetic algorithm and Particle Swarm Optimization(PSO) based algorithm. The experimental result indicate that the GWO based technique is performing better in terms of load balancing and also having reduced makespan.**

***Index Terms*—Cloud Computing, Resource Allocation, Load Balancing, Container, Task Scheduling, Grey Wolf Optimization**

I. Introduction

One of the most talked topics in the field of internet technology in the last decade is cloud computing. Executing users' tasks or providing different services to the user on-demand remotely by a set of servers connected over is the main concept of cloud computing. A pool of virtualized resources is available in the server and are allocated to user application on-demand. Users can use cloud resources by pay-as-you-go basis i.e. they will only pay for the resources they are using. Due to the increasing number of cloud users in the last few years and limited numbers of available resources, improper distribution of clients' requests may cause some server to be overloaded and some to be idle without doing any work resulting in reduced resource utilization. The proper distribution of client requests or user tasks among all the servers so that no server will be overloaded or underloaded is called load balancing.

Over the year different algorithms have been proposed for load distribution in the cloud system and a lot of research is still going on to improve the existing approaches. Containerization technology is the recent development in cloud computing which offers better performance than the classic approach of virtualization. Container support software virtualization such as OS-level virtualization whereas virtual machine supports hardware virtualization. The container is light weighted than the virtual machine because it only requires the libraries and binaries needed to run an application. The startup time of a container is much faster(in sec) than that of a virtual machine(in min). Being a light weighted approach container performs better than virtual machine and resource utilization is significantly better than a virtual machine. In a containerized cloud server, the objective is to choose a container from a set of available containers on which the task is going to be executed by considering the load of all other containers. In this paper, we have proposed a Grey Wolf Optimization(GWO) based algorithm for load distribution in the containerized cloud and also to reduce the makespan. The performance of GWO algorithm is compared with the Genetic algorithm and Particle Swarm Optimization(PSO) based algorithm.

The rest of the paper is organized as: Section-II gives the related work has been done and motivated us for this work. In section-III, the proposed system model is described. Section-IV, describes the proposed GWO based algorithm for load balancing and reducing makespan. The experimental analysis and results are presented in section-V. Finally, conclusion is given in section-VI with some future directions.

II. Related Work

Hung et al. [1] proposed a modified max-min task scheduling algorithm for load balancing that tries to improve the completion time of the request. The algorithm first clusters the task according to their size and then make a cluster of the virtual machine according to their utilization. Then the algorithm assigns the largest task to the least utilized virtual machine. The result gives improved completion time of requests. Mohanty et al. [2] proposed JAYA algorithm for load balancing. This algorithm considers fewer control parameters and gives an optimized result. PSO and Genetic algorithms are used for comparison with JAYA and it is found that JAYA algorithm performs better than PSO and GA by minimizing response time.

Abed et al. [3] proposed an Adaptive firefly algorithm based on a round-robin algorithm. The proposed algorithm maximize CPU utilization and minimize response time by distributing the workload among different virtual machine by taking care of

the load and availability of each virtual machine. Farrag et al. [4] proposed a Grey wolf optimization(GWO) and an Ant-lion Optimization(ALO) for load balancing to reduce makespan. The result is compared with Particle Swarm Optimization (PSO) and Firefly Algorithm (FFA) and proves that GWO and ALO perform better than PSO and FFA. Xavier et al. [5] proposed a chaotic social spider algorithm for minimizing makespan and balancing resource utilization. The algorithm performance is evaluated in the cloudsim and the obtained result is compared with Artificial Bee Colony, PSO, and GA.

Mrhari et al. [6] proposed a new evolutionary algorithm SASPSOOLB based on modified particle swarm optimization and game theory to balance the load in cloud computing. The algorithm gives a better result in terms of makespan and response time. The result is compared with the genetic algorithm. In [7] Gao et al. explained the importance of load balancing in cloud computing. For quickly finding the candidate node in load balancing, two strategies namely max-min rule and forward-backward ant mechanism are used. Pheromone's initialization and update rules are defined based on the physical resource in the cloud data center. The simulation result gives dynamic load balancing with less searching time and better network performance. Sharma et al. [8] implemented a bat algorithm in MatLab for balancing load in the cloud and the result is compared with fuzzy GSO and round-robin. The result is compared iwith regars to the number of tasks and response time. The proposed algorithm minimizes the response time.

Mao et. al [9] proposed max-min based algorithm for scheduling task and also for balancing the load. The proposed algorithm keeps a status table for the task to estimate real-time virtual machine load and expected time to complete a task. Based on this status table, the workload is allocated among virtual machine and load balance is obtained. Khan et. al. [10] implemented an ant colony algorithm for scheduling tasks to balance the load. This paper considered different parameters like performance, service level agreement violation, energy, overhead. The algorithm improves performance by achieving good response time, throughput, less energy consumption and also load balance among nodes.

Fard et. al. [11] proposed multi-objective based approach for scheduling workflow in a heterogeneous environment. They consider four parameters namely economic cost, makespan, reliability, and energy consumption for optimization and got a better result when compared with bicriteria heuristic scheduling and genetic algorithm. Zuo et. al. [12] proposed the PBACO algorithm for a multiobjective method for scheduling tasks based on a cost model of resources. The algorithm improves four performance metrics namely makespan, deadline violation rate, resource utilization, and cost. The result of the algorithm is compared with FCFS and Min-MIn algorithm and they got a better result with PBACO. Basu et. al. [13] proposed the GAACO algorithm for scheduling IoT application tasks that are dependent. They combine Ant colony optimization algorithm and genetic algorithm for the selection of a combination of the task. GAACO gave a better result in terms of minimizing makespan when compared with conventional genetic algorithm and ant colony optimization algorithm. A dead-line sensitive task scheduling has been described by Sampa Sahoo et al. in [19]

Lin et. al. [14] gives a multi-objective ACO_MCMS algorithm for solving container-based microservice scheduling in the cloud. The algorithm considers the utilization of storage and computing resources of the physical machine, several requests for microservice and physical machine failure rate. The result obtained algorithm is compared with multiopt, GA_MOCA and spread algorithm and got a better result. Jena et. al. [15] proposed the TSPSO algorithm for optimizing the processing time of task and energy consumption in the data center. The result of the TSPSO algorithm outperforms RSA and BRS in terms of makespan and energy reduction. Adikari et. al. [16] implemented an accelerated particle swarm optimization algorithm for container allocation in cloud. The main objective of the algorithm is to minimize the completion time of the task and energy consumption with the effective utilization of resources. Abdi et. al. [17] implemented a modified version of PSO algorithm for scheduling task in the cloud. The algorithm minimizes the completion time of the task. The result obtained using GWO is compared with PSO and genetic algorithm.

III. System Model

The system model mainly consists of two main component 1.Task Manager and 2.Task Scheduler. The task manager collects the task request from the user and submits it to the task scheduler. The task scheduler will allocate these tasks to a container using the proposed Grey Wolf Optimization algorithm. It is assumed that the task comes with its requirements and specification. Also, task is independent of each other. Let T be the set of task $\{t_1, t_2, ..., t_n\}$. Each task is specified with task id, task arrival time in second, task length in MI. So task attributes are $\{T_{id}, AT, TL\}$. Let C be the set of container $\{c_1, c_2, ..., c_m\}$. Each container is specified with the amount of RAM in GB and processing capacity in MIPS. So, attributes of container considered are $\{M, P\}$, where M is memory size and P is processing capacity. The execution time of task is calculated as,

$$ET_i = \frac{TL_i}{P_j} \tag{1}$$

whre ET_i is the execution time and TL_i is the task length of task i and P_j is the processing capacity of container j on which task i is allocated. Completion time of the task is calculated as summation of waiting time and execution time.

$$CT = WT + ET \tag{2}$$

The makespan of all task is the total completion time taken by all the task. We have to minimize this makespan MS.

$$MS = \sum_{i=1}^{n} CT_i \tag{3}$$

Also, this paper consider load balance of each container by calculating load variation(LV) using variance formula.

$$LV = \frac{1}{m}\sum_{i=1}^{m} ETC_i - \overline{ETC} \tag{4}$$

where ETC_i is the total execution time of i^{th} container and $\overline{ETC}$ is the mean of total execution time of all container. So, main objective is to minimize makespan and load variation.

IV. Proposed GWO based Algorithm for Load Balancing

The makespan of a task can be reduced efficiently by allocating tasks to an appropriate container. This paper implements Grey wolf optimization(GWO) algorithm for scheduling task from the user in cloud computing. Grey Wolf Optimization algorithm imitates the leadership like hierarchy and hunting mechanism of grey wolf which is a population-based meta heuristic algorithm. They favor to live in a pack that is categorized in four-level namely omega, delta, beta, and alpha which belong to level-4, level-3, level-2, and level-1 respectively. Level-1 is the leader who can be male or female and responsible for making a decision. The level-2 group helps a group of Level-1 for making a decision and they are adviser and discipliner for the pack. Level-3 consists of scouts which are suppose to watch the boundaries, sentinels which are suppose to protect the pack, elders which were alpha or beta sometimes, hunters that help alpha for hunting, caretakers for caring waek, ill and wounded wolves. Level-4 groups are scapegoat in the pack which is last allowed wolves for eating.

Based on position of alpha wolf, beta wolf and delata wolf, other wolf search for prey. The solution produces by alpha is best, followed by beta solution and then delta. And the remaining solutions are produced by omega. The omega group update their position using the following mathematical formula,

$$\vec{D} = |\vec{Q}.\vec{S_p}(i) - \vec{S}(i)| \tag{5}$$

$$\vec{S}(i+1) = \vec{S_p}(i) - \vec{P}.\vec{D} \tag{6}$$

where $\vec{P}$ and $\vec{Q}$ are coefficient vectors and i denotes the current iteration number, $\vec{S}$ denotes the position vector of grey wolf and $\vec{S_p}$ indicates' the position vector of prey.

The vectors $\vec{P}$ and $\vec{Q}$ can be calculated as,

$$\vec{P} = 2.\vec{a}.\vec{r_1} - \vec{a} \tag{7}$$

$$\vec{Q} = 2.\vec{r_2} \tag{8}$$

where $\vec{r_1}$ and $\vec{r_2}$ are random vector. The vector component $\vec{a}$ is linearly decreased from 2 to 0 using,

$$a = 2 - \frac{2.i}{number\,of\,iteration} \tag{9}$$

where i is the current iteration number.

The solution for GWO algorithm is represented as $S = \{3, 2, 1, 3, 2\}$ which means that task t_1 is allocated to c_3 container, task t_2 is allocated to c_2 container, task t_3 is allocated to c_1 container, task t_4 is allocated to c_3 container and task t_5 is allocated to c_5 container. Then according to this allocation, makespan and load variation is calculated. The three solution which are giving minimum makespan and load variation are taken as alpha, beta and delta. And the rest of the solution are updated based on position of this three.

Algorithm 1 GWO Algorithm

Input: container_list, task_list, population_size, number_of_iteration

Output: task_allocation_solution P.

1: Initial solution of size equal to population_size for the algorithm is generated randomly.
2: Calculate value of fitness for all solution using equation 3 and 4.
3: Sort the solution according to fitness values in ascending order.
4: Define first three best solution as alpha ($\vec{S_\alpha}$), beta ($\vec{S_\beta}$) and delta ($\vec{S_\delta}$).
5: **for** i=0 to number_of_iteration **do**
6: **for** each remaining solution apart from alpha, beta and delta **do**
7: Initialize $a, \vec{A}$ and $\vec{C}$ as shown in equation 9, 7 and 8 respectively.
8: Update the current solution using equation 5 and 6.
9: **end for**
10: Update the value of $a, \vec{C}$ and $\vec{A}$.
11: Calculate value of finess function for all the solution.
12: Update $\vec{S_\alpha}$, $\vec{S_\beta}$ and $\vec{S_\delta}$ according to fitness value.
13: **end for**
14: return $\vec{S_\alpha}$.

Algorithm 2 Fitness Function

Input: solution_list, population_size,

Output: fitness value.

1: Initialize LV=0, MS = 0.
2: **for** each solution in solution_list **do**
3: Calculate $MS = \sum_{i=1}^{n} CT_i$.
4: Calculate $LV = \frac{1}{m}\sum_{i=1}^{m} ETC_i - \overline{ETC}$.
5: **end for**
6: return MS, LV.

V. Experimental Analysis and Results

The system model considers heterogeneous servers. We consider container processing capacity between 2500 to 4000 MIPS and container RAM size 2GB, 4GB and 8GB. The task arrival time is taken from 0 to 1000 using Poisson distribution. The number of tasks taken is 1000, 2000 and 3000 and the number of containers taken is 10, 20, 30, 40 and 500. The task size considered is between 5000 to 10000 MI. All parameter considered for our model is shown in TABLE-I.

The number of iteration for grey wolf optimization algorithm is decided by performing experiment on 10, 20, 30, 40, and 50 containers with 1000, 2000, and 3000 task and

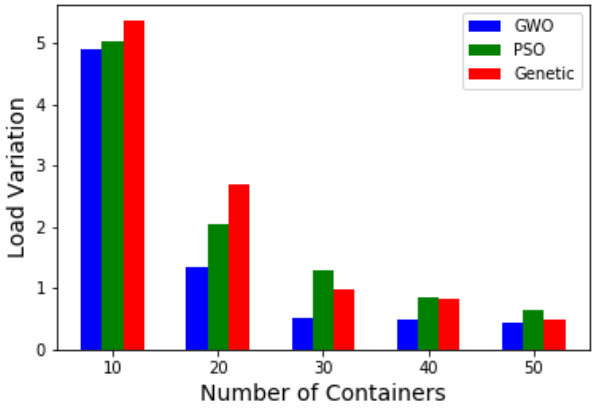

Fig. 1. Load variation with 1000 task

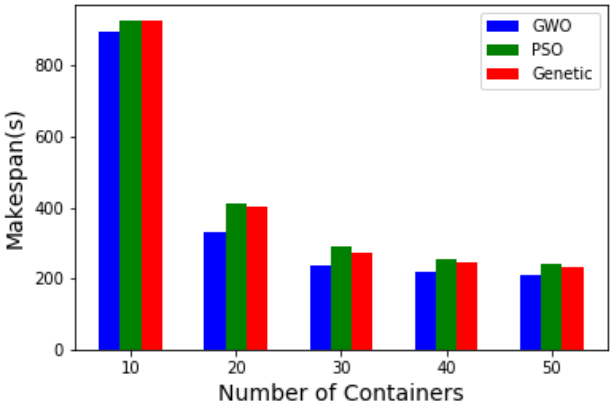

Fig. 4. Makespan with 1000 task

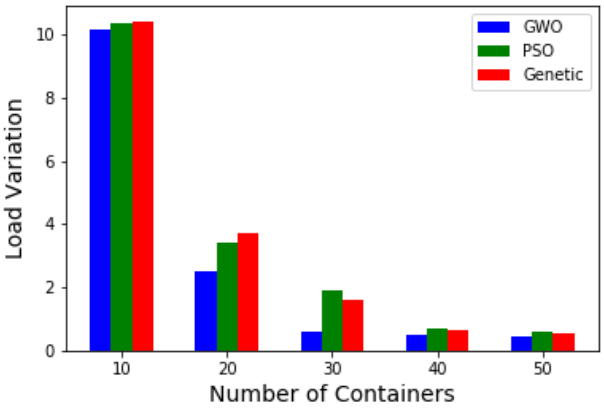

Fig. 2. Load variation with 2000 task

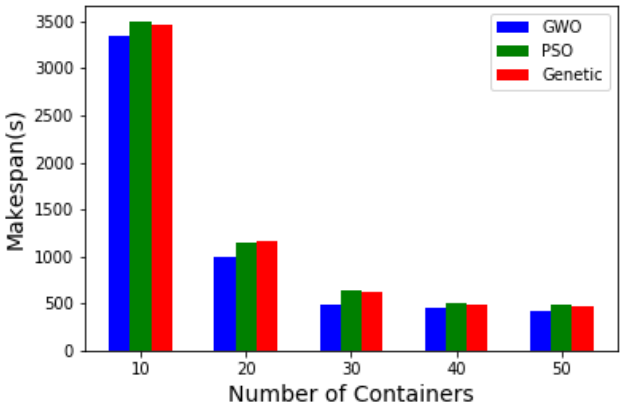

Fig. 5. Makespan with 2000 task

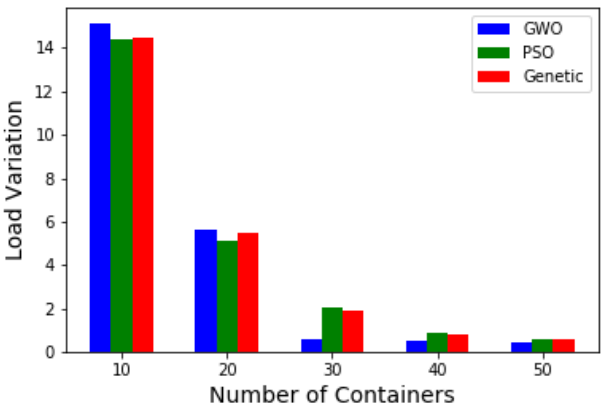

Fig. 3. Load variation with 3000 task

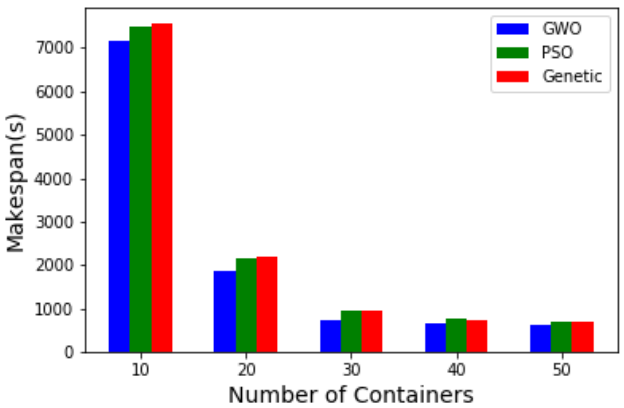

Fig. 6. Makespan with 3000 task

considering 5000 iteration. We got constant value after 1000 iteration for both makespan and load variation. Therefore we fix 1000 iteration for our proposed algorithm.

TABLE I
PARAMETER ASSUMED FOR GIVEN SYSTEM MODEL

Container Processing Capacity	2500 to 4000 MIPS
Container Memory Capacity	2GB, 4GB, 8GB
Task Arrival Time	[0, 1000] using Poisson distribution
Task Size	5000 to 10000 MI
Number of Task	1000, 2000, 3000
Number of Container	10, 20, 30, 40 and 50

We consider 100 population size for Grey Wolf Optimization algorithm. Then the experiment is performed with 1000, 2000, 3000 task with 10, 20, 30, 40, 50 container using Grey wolf optimization algorithm. The result of the algorithm is compared with Genetic algorithm and particle swarm optimization algorithm. We got better result with grey wolf optimization algorithm. The calculated makespan using Grey Wolf Optimization algorithm is less than that of Genetic Algorithm and PSO in different number of container. Similarly, the experimental result show that the load variation is more using GWO when the number of container is less. When the number of container increased, the load variation in GWO reduced significantly and performs better than Genetic Algorithm and PSO. The results are shown in Fig. 1, Fig. 2, Fig. 3, Fig. 4, Fig. 5 and Fig. 6. The result graph show that load variation and makespan decreases as the number of container increases.

VI. CONCLUSION AND FUTURE WORK

This paper mainly focus on distributing the workload equally among all the containers and reducing the makespan. Different approaches have been proposed by different researchers for load balancing such as Genetic algorithm based, PSO, ACO, Min-Max, etc. In this paper we have proposed a Grey Wolf Optimization based algorithm for load balancing and reducing the makespan. The performance of GWO based approach is compared with GA and PSO based algorithm. The makespan of GWO is less than GA and PSO. The number of container were varied from 10 to 50. Initially the load variation in GWO algorithm is more but when the number of container increases, load variation decreases and GWO performs better than GA and PSO.

The proposed algorithm can also be useful in some other technology such as Internet of Things(IoT) and Fog Computing. In future, we will try to apply this algorithm in the architecture model proposed in [18] for smart city using cognitive IoT to enhance the model.

REFERENCES

[1] Hung, Tran Cong, Le Ngoc Hieu, Phan Thanh Hy, and Nguyen Xuan Phi. "MMSIA: Improved Max-Min Scheduling Algorithm for Load Balancing on Cloud Computing." In Proceedings of the 3rd International Conference on Machine Learning and Soft Computing, pp. 60-64. 2019.

[2] Mohanty, Subhadarshini, Prashanta Kumar Patra, Mitrabinda Ray, and Subasish Mohapatra. "An Approach for Load Balancing in Cloud Computing Using JAYA Algorithm." International Journal of Information Technology and Web Engineering (IJITWE) 14, no. 1 (2019): 27-41.

[3] Abed, Marwa M., and Manal F. Younis. "Developing Load Balancing for IoT-Cloud Computing Based on Advanced Firefly and Weighted Round Robin Algorithms." Baghdad Science Journal 16, no. 1 (2019): 130-139.

[4] Farrag, Aya A. Salah, Safia Abbas Mohamad, and M. El Sayed. "Swarm Intelligent Algorithms for solving load balancing in cloud computing." Egyptian Computer Science Journal 43, no. 1 (2019).

[5] Xavier, VM Arul, and S. Annadurai. "Chaotic social spider algorithm for load balance aware task scheduling in cloud computing." Cluster Computing 22, no. 1 (2019): 287-297.

[6] Mrhari, Amine, and Youssef Hadi. "A Load Balancing Algorithm in Cloud Computing Based on Modified Particle Swarm Optimization and Game Theory." In 2019 4th World Conference on Complex Systems (WCCS), pp. 1-6. IEEE, 2019.

[7] Gao, Ren, and Juebo Wu. "Dynamic load balancing strategy for cloud computing with ant colony optimization." Future Internet 7, no. 4 (2015): 465-483.

[8] Sharma, Shabnam, Ashish Kr Luhach, and S. A. Sinha. "An optimal load balancing technique for cloud computing environment using bat algorithm." Indian J Sci Technol 9, no. 28 (2016): 1-4.

[9] Mao, Yingchi, Xi Chen, and Xiaofang Li. "Max–min task scheduling algorithm for load balance in cloud computing." In Proceedings of International Conference on Computer Science and Information Technology, pp. 457-465. Springer, New Delhi, 2014.

[10] Khan, Shagufta, and Niresh Sharma. "Effective scheduling algorithm for load balancing (SALB) using ant colony optimization in cloud computing." International Journal of Advanced Research in Computer Science and Software Engineering 4, no. 2 (2014).

[11] Fard, Hamid Mohammadi, Radu Prodan, Juan Jose Durillo Barrionuevo, and Thomas Fahringer. "A multi-objective approach for workflow scheduling in heterogeneous environments." In 2012 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (ccgrid 2012), pp. 300-309. IEEE, 2012.

[12] Zuo, Liyun, Lei Shu, Shoubin Dong, Chunsheng Zhu, and Takahiro Hara. "A multi-objective optimization scheduling method based on the ant colony algorithm in cloud computing." Ieee Access 3 (2015): 2687-2699.

[13] Basu, Sayantani, Marimuthu Karuppiah, K. Selvakumar, Kuan-Ching Li, SK Hafizul Islam, Mohammad Mehedi Hassan, and Md Zakirul Alam Bhuiyan. "An intelligent/cognitive model of task scheduling for IoT applications in cloud computing environment." Future Generation Computer Systems 88 (2018): 254-261.

[14] Lin, Miao, Jianqing Xi, Weihua Bai, and Jiayin Wu. "Ant Colony Algorithm for Multi-Objective Optimization of Container-Based Microservice Scheduling in Cloud." IEEE Access 7 (2019): 83088-83100.

[15] Jena, R. K. "Multi objective task scheduling in cloud environment using nested PSO framework." Procedia Computer Science 57 (2015): 1219-1227.

[16] Adhikari, Mainak, and Satish Narayana Srirama. "Multi-objective accelerated particle swarm optimization with a container-based scheduling for Internet-of-Things in cloud environment." Journal of Network and Computer Applications 137 (2019): 35-61.

[17] Abdi, Solmaz, Seyyed Ahmad Motamedi, and Saeed Sharifian. "Task scheduling using modified PSO algorithm in cloud computing environment." In International conference on machine learning, electrical and mechanical engineering, pp. 8-9. 2014.

[18] Patra, Manoj Kumar. "An architecture model for smart city using Cognitive Internet of Things (CIoT)." In 2017 Second International Conference on Electrical, Computer and Communication Technologies (ICECCT), pp. 1-6. IEEE, 2017.

[19] Sahoo, Sampa, Bibhudatta Sahoo, and Ashok Kumar Turuk. "A learning automata-based scheduling for deadline sensitive task in the cloud." IEEE Transactions on Services Computing (2019).