JOB SERVICE DEMAND BASED RESOURCE ALLOCATION POLICY FOR CLOUD-FOG COMPUTING ENVIRONMENTS

Tran Thi Xuan[*], **Phung Trung Nghia**
TNU - University of Information and Communication Technology

ARTICLE INFO

Received:	**19/6/2022**
Revised:	**22/8/2022**
Published:	**23/8/2022**

KEYWORDS

Cloud-Fog Computing
5G network core
Resource Allocation
Job characteristics
Resource heterogeneity

ABSTRACT

In the development of 5G mobile network, Fog computing becomes an emerging component of the Cloud computing paradigm as the 5G core to assure the diverse computational demands of IoT applications can be satisfied. A cloud-based application requires a combined use of cloud and local resources for its processing. Resource allocation for cloud-based jobs plays an important role to achieve both QoS and efficient resource utilization of a Cloud-based computing environment. This study considers a hierarchical computing architecture in 5G network made up of various computational machines from user device, edge server, fog to cloud center. The system is to handle diverse IoT applications in heterogeneous computational resources. This study proposes a resource allocation policy that takes account of job characteristics in terms of job service demands with the aim at improving job service quality. We develop simulation software to investigate the proposed policy. Numerical results point out that the proposal reduces the average response time by 5% to 9% and yields better resource utilizations in comparison to a best-effort Round Robin policy.

GIẢI PHÁP PHÂN BỔ TÀI NGUYÊN DỰA TRÊN YÊU CẦU DỊCH VỤ CỦA CÔNG VIỆC TRONG MÔI TRƯỜNG ĐIỆN TOÁN ĐÁM MÂY - SƯƠNG MÙ

Trần Thị Xuân[*], **Phùng Trung Nghĩa**
Trường Đại học Công nghệ thông tin và Truyền thông – ĐH Thái Nguyên

THÔNG TIN BÀI BÁO

Ngày nhận bài:	**19/6/2022**
Ngày hoàn thiện:	**22/8/2022**
Ngày đăng:	**23/8/2022**

TỪ KHÓA

Điện toán đám mây-sương mù
Mạng lõi 5G
Phân bổ tài nguyên
Đặc tính công việc
Đa dạng tài nguyên

TÓM TẮT

Trong sự phát triển của mạng di động 5G, điện toán sương mù là một thành phần của mô hình điện toán đám mây với vai trò mạng lõi 5G nhằm đảm bảo có thể đáp ứng các nhu cầu tính toán đa dạng của các ứng dụng IoT. Ứng dụng dựa trên đám mây yêu cầu sử dụng kết hợp tài nguyên đám mây và tài nguyên cục bộ để tính toán. Vấn đề phân bổ tài nguyên cho các ứng dụng đám mây đóng vai trò quan trọng để đạt được chất lượng dịch vụ (QoS) và tính hiệu quả trong việc sử dụng tài nguyên. Nghiên cứu này xem xét một kiến trúc điện toán phân cấp trong mạng 5G được tạo thành bởi nhiều máy tính toán khác nhau từ thiết bị người dùng, máy chủ biên, tầng sương mù đến trung tâm đám mây. Hệ thống điện toán thực thi việc xử lý các ứng dụng IoT đa dạng trên các tài nguyên tính toán có các đặc tính, khả năng xử lý khác nhau. Nghiên cứu này đề xuất một thuật toán phân bổ tài nguyên có xét đến yêu cầu về dịch vụ của công việc nhằm mục đích nâng cao chất lượng dịch vụ. Chúng tôi phát triển phần mềm mô phỏng để đánh giá giải pháp được đề xuất. Kết quả mô phỏng chỉ ra rằng đề xuất này giảm thời gian phản hồi trung bình từ 5% đến 9% và mang lại hiệu quả sử dụng tài nguyên tốt hơn so với chính sách phân bổ Round Robin.

DOI: https://doi.org/10.34238/tnu-jst.6194

[*] Corresponding author. *Email: ttxuan@ictu.edu.vn*

1. Introduction

5G mobile network has become crucial to enable Internet of Things (IoTs) and handle its increasing number of applications. The 5G mobile core defined by 3GPP utilizes cloud-aligned, service-based architecture (SBA) that spans across all 5G functions and interactions including authentication, security, session management and aggregation of traffic from end devices [1]. However, the ordinary cloud computing paradigm cannot fulfill requirements of all 5G functions and IoT applications that require diverse resource and service time demands. Edge computing, by which tasks are processed at the local network devices, was introduced as a part of 5G network to handle real-time tasks. However, IoT applications have a wide range of requirements for resources and service quality, which challenges the processing capacity of Cloud and Edge nodes. Fog computing appears as a complement of cloud to satisfy the diverse computational demands of IoT applications by moving cloud services to the edge network [2], [3].

As Fog nodes have differences in distribution and computing capacity, an effective scheduling policy can reduce service delay and energy consumption. As a result, job scheduling in Cloud-Fog computing environments has received a lot of attention in the literature works [4] – [8].

In [4], the authors proposed a batch-mode task scheduling algorithm based on the relationship between the fog node set and the task set. Compared with existing batch-mode scheduling algorithms (MCT, MET, MIN-MIN), the proposal yields a shorter total completion time of tasks. Rafique *et al.* [9] focused on balancing task execution time and energy consumption at Fog layer computing resources; the proposed NBIHA algorithm resulted in resource utilization, average response time, and energy consumption but an increase in task execution time. A time-cost based scheduling algorithm was introduced in [10], wherein authors applied a set of modifications of the Genetic Algorithm in their proposal. Fan *et al.* proposed a multi-objective scheduling algorithm that balances latency, load and reliability for an edge computational system [11]. Though numerous literature studies have approached to job scheduling/resource allocation problem in Fog-enabled system, it remains a topical, open problem due to the diversity of IoT applications, resource heterogeneity, and multiple criteria.

Previous works have not paid enough attention to the impact of job characteristics on the performance of task-resource mapping decision in a cloud-fog computing environment. Since applications have various needs of resources, service demands of jobs play a critical factor to scheduling decisions. This study investigates the task scheduling problem taking job service demand and workload models into account. The contributions of this work are highlighted in what follows.

1. This study argues that the job service time demand and resource processing capacity are important factors used for task-resource mapping in fog computing;

2. Numerical results indicate that using knowledge of workloads helps to improve both the performance and resource utilization of a Cloud-Fog computing system.

The paper is organized as follows. Section II describes the material used for the research including a system model and the proposed scheduling solution. Section III presents the experimental design for evaluation, the results, and discussions. Finally, Section IV concludes the paper.

2. Research materials

2.1. System modeling

As a cloud-fog environment is the integration of a large number of distributed devices and connections, processing and storage capacity of fog nodes are extremely varied. The considered model of a Cloud-Fog environment is illustrated in Figure 1. The system is composed of virtual machines in a remote cloud center, cloudlet (that is, a small-scale version of cloud data centers located geographically nearby users), low-performance Raspberry Pis (which are integrated within networking devices), and user smart devices (so-called edge subsystem). It is assumed that nodes in a subsystem (cloud, cloudlet, Raspberries, edge) are homogeneous, i.e. provide an equal computing capacity.

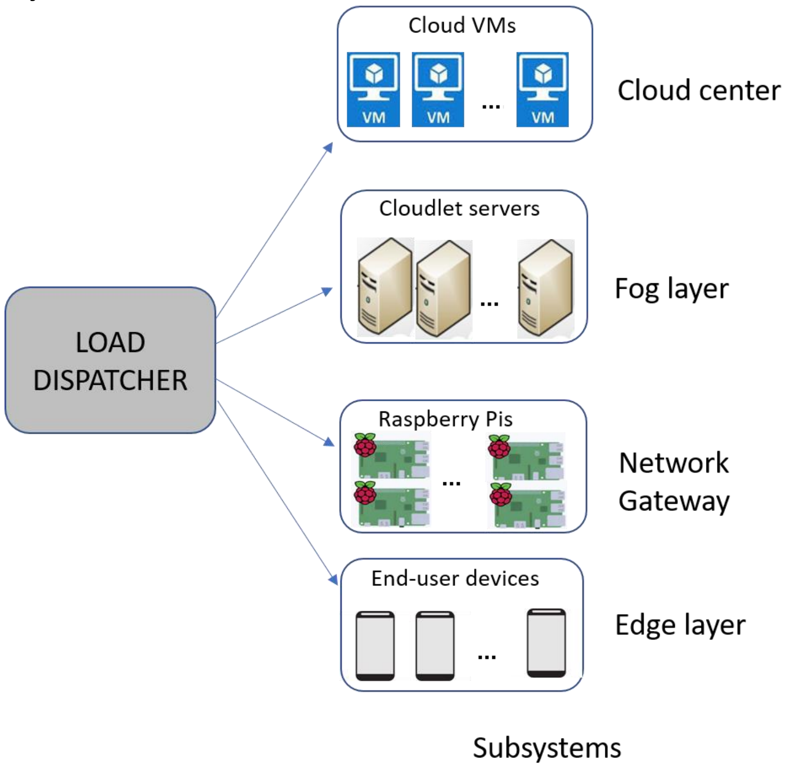


Figure 1. *A system model of Cloud-Fog computational architecture*

We assume input workloads:
- can be executed on any computing node;
- are attended to by First Come First Served (FCFS) service policy;
- are uninterruptible while being processed;
- are compute-intensive and have service time demand known by the scheduler; and
- are independent of each other.

An incoming job is to be executed by a fog node according to a specific scheduling algorithm. In what follows, we describe scheduling algorithms that allocate arriving jobs to a computational node in the cloud-fog system.

2.2. A proposal of resource allocation policy

As afore-mentioned, the system resources are distributed and heterogeneous (i.e., fog nodes have different resource capacity) and jobs have different service demands. In this section is the proposal of a scheduling algorithm that aims to reduce network delay for jobs with acute service time demands by allocating their tasks into as close nodes as possible. According to the Cloud-Fog architecture, edge devices are closest to jobs, then the Gateway Raspberries machines and Cloudlet server come, and the cloud center is furthermost. However, as Raspberries are low-performance devices, it should be not to use these resources for job execution that require short processing time.

The resource allocation policy (illustrated by Algorithm 1) follows rules as:

- The edge subsystem will have highest priority to execute job with acute service demand and the cloudlet comes as the second. Otherwise, subsystem with the lowest load is chosen;

- If job requires a long run, a subsystem is chosen according to low load rule, excluding the end-user devices since we want to avoid user suffering from the battery consumption and performance degradation of devices;
- A job will be served immediately if there is an available computing node;
- The shortest queue server is chosen if all servers are busy.

Let a job be identified by *(j, sd_j)*, where *j* is job identification and sd_j is the service time demand of job *j*. A threshold called statistical mean service demand is used to estimate the service demand of job as follows. Let *N(t)* and *β(t)* denote the number of historical incoming jobs and the average service time demand of those jobs up to the considered time *t*. The statistical mean service demand is calculated as given in (1):

$$\beta(t) = \frac{\sum_{j=1}^{N(t)} sd_j}{N(t)} \tag{1}$$

At a job submission, the scheduler compares jobs' service demand with the current threshold *β(t)*. The scheduler routes a job to a computing node as close to the edge as possible if the job's service demand is less than the threshold. Otherwise, a job's task will be allocated to the shortest queue node in the lowest load subsystem, excluding the end devices. Algorithm 1 presents the pseudo-code of the proposed policy.

Algorithm 1. Job service demand based scheduling Algorithm

```
for each new arriving job j do
    UPDATE β(t)
    CALCULATE loads of subsystems: Edge, Cloudlet, Raspberries, Cloud
    if $sd_j \le \beta(t)$ then
        if FIND free_server in the Edge subsystem then
            ROUTE job j to free_server of the Edge
        else if FIND free_server in the Cloudlet then
            ROUTE job j to free_server of the Cloudlet
        end if
    else
        CHOOSE subsystem with the lowest load from Cloudlet, Raspberries, and Cloud
        CALCULATE server queues of all nodes in the chosen subsystem
        ROUTE job j to the shortest queue server
    end if
end for
```

We evaluate the proposed service demand based algorithm by comparing to a Round-Robin based allocation policy wherein each computing node is selected with the same probability in a circle way with no consideration to resource characteristics.

3. Numerical results and discussion

The evaluation is conducted using a simulation software written in C. A simulation run is stopped after five million completions. System performance metrics in terms of subsystem resource utilization, average waiting time, and average response time per job (i.e. the average time period since a job arrives until it is responded with results) are measured.

3.1. System load

To construct a testbed for evaluation, we follow the rules that:
- the computing resources are most available at the cloud center,
- the number of end-user devices and the number of cloudlet servers are medium, and
- networking devices contributes the least portion of the computing system.

The Cloud-Fog computing environment used in simulation runs is composed of 128 virtual machines (VMs) located at cloud center, 64 end-user devices, 64 rich-resource cloudlet servers, 32 low-performance Raspberry Pi devices.

We assume that the inter-arrive time and the execution time of jobs are exponentially distributed with means of 1/λ and 1/μ, respectively. A end-user device, a cloudlet node, a Raspberry Pi, and a VM are assumed to have ability of executing tasks at service rate μ1 = 2.0 (tasks/s), μ2 = 1.0 (tasks/s), μ3 = 0.5 (tasks/s), μ4 = 1.0 (tasks/s), respectively. As a result, the considered system has the average service rate of at most (64× 2.0 + 64 × 1.0 + 32 × 0.5 + 128 × 1.0) = 336 (jobs/s). The simulation runs with a range from low to high load of 20% - 60%.

The proposed solution is compaerd with the best-effort Round-Robin (RR) policy. Parameters used for performance evaluation of those two algorithms are given as follows.

- Average response time (RT) is the mean time that a job stays in the system (since its arrival up to departure) as given in (2). Let $\boldsymbol{rt_j}$ be the response time of job j (j = 1, …, N) and N is the total completed jobs in the simulation run.

$$RT = \frac{1}{N}\sum_{j=1}^{N} rt_j \qquad (2)$$

- Average waiting time (WT) is the mean time delay of a job spending in queue and waiting for the service. Let wt_j be the waiting tim of job j, the average waiting time of N jobs in the simulation run is calculated as (3):

$$WT = \frac{1}{N}\sum_{j=1}^{N} wt_j \qquad (3)$$

- Subsystem resource utilization is the percentage of the busy time of nodes in a subsystem over the simulation time

3.2. Obtained results

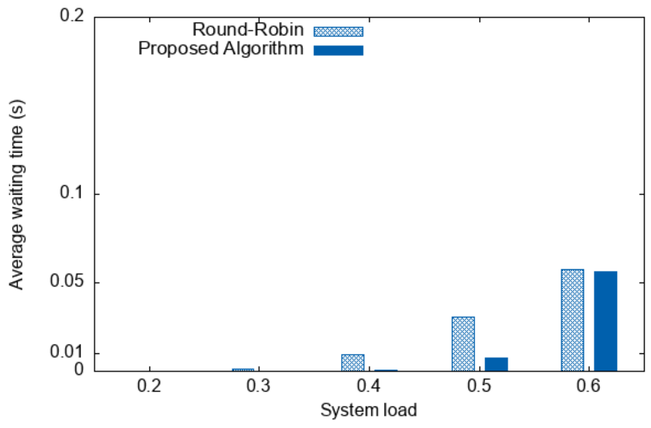


Figure 2. *Average waiting time of job (s)*

Figures 2 and 3 present the average waiting time and the average response time per job, respectively.

It can be observed from Figure 2 that jobs have to wait longer with Round-Robin, especially at low load. When load increases, the waiting time of job increases when the proposed algorithm is used and there is no difference in the delay time between two policies. This increase can be explained by the preference to nodes in cloudlet and cloud center when load is high. Since edge nodes have high performance capacity, the Round-Robin tends to put more tasks to the Edge subsystem. On the contrary, JCA aims keeping edge devices a low utilization to avoid users' disappointment.

Figure 3 presents that the proposed algorithm reduces the average response time per job by 5% to 9% in compared to Round-Robin solution, despite the change of workload intensity. The reason is that our proposal chooses a computing node based on their processing capacity and location; wherein, powerful servers are preferred if available and the proposed policy attempts a node that is as close to user's job as possible to avoid job transfer through the network.

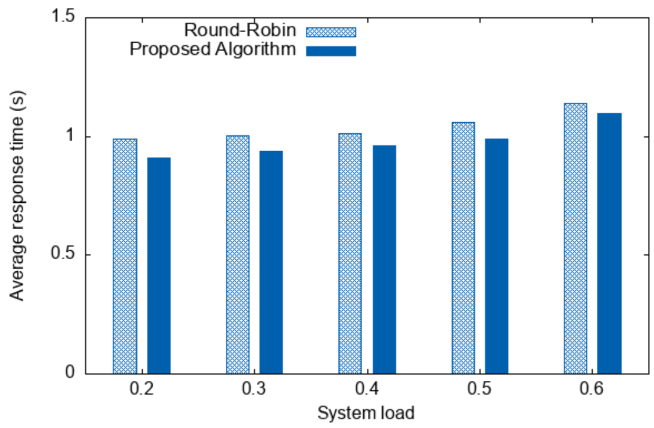
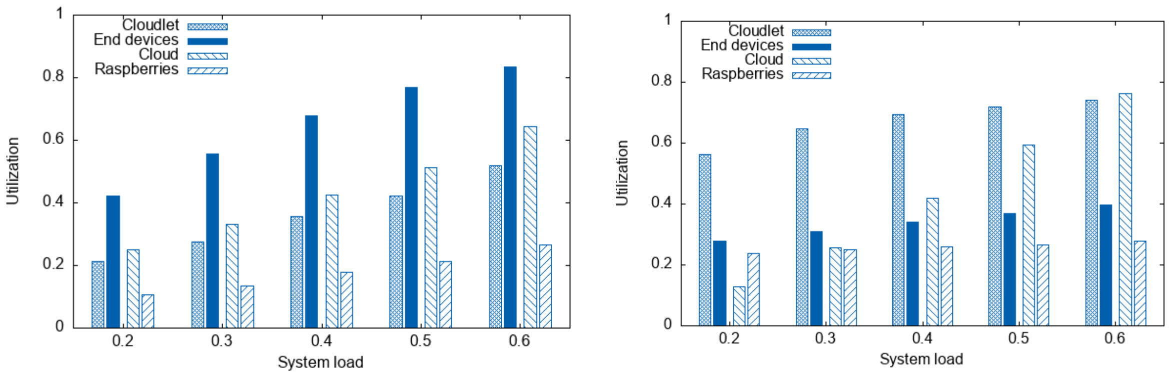


Figure 3. *Average response time of job (s)*



a) *b)*

Figure 4. *Subsystem utilization: a) with Round-Robin, b) with proposed algorithm*

The resource utilization of the four subsystems (Edge, Cloudlet, Raspberries, and Cloud center) is plotted in Figure 4. It can be observed that when the workload intensity increases, Round Robin policy (Figure 4a) puts load stress on edge devices (up to average of 80% busy nodes at the user side) while using less than 60% of powerful servers in Cloudlet and approximately 65% of Cloud center. The overuse of end-user devices to perform computational tasks may result in slowdown on user devices and cause a discomfort to users.

On the contrary, the proposed algorithm (Figure 4b) keeps the load on end-user devices at medium level (the average utilization of this subsystem is less than 40% even when the arrival intensity increases high). As shown in Figure 4b, when the arrival intensity is high, powerful computing servers of cloudlet and cloud center are preferred (with average utilization of 75% - 80%). Furthermore, the low-compute devices (Raspberries) are also not preferred and remains a low utilization less than 40%. In summary, the loads of subsystems are likely balanced with the proposal.

4. Conclusion

In this paper, we study the impact of job service demand on the performance of a Cloud-Fog computing environment which includes all types of computing resources from end-user smart devices to servers in cloud center. We address the inefficient utilization of computing resources when the naïve Round-Robin is used to allocate diverse applications to the system. The proposed resource allocation algorithm taking into account job service demand and resource computing capacity yields better performance and resource utilization of the system, in comparison to the Round-Robin algorithm. However, Cloud-Fog computing is a highly heterogeneous system therein functionalities of fog nodes can be different. The consideration of application types and node functionalities is beyond this study. In addition, energy efficiency is a crucial issue in all large-scale systems, which drives our further study. We intend to compare the solution with other novel algorithms and expand the investigation on the energy consumption issue in Cloud-Fog computing systems.

REFERENCES

[1] L. Chettri and R. Bera, "A Comprehensive Survey on Internet of Things (IoT) Toward 5G Wireless Systems," *IEEE Internet of Things Journal*, vol. 7, no. 1, pp. 16 – 32, Jan. 2020.

[2] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," *MCC '12: Mobile cloud computing*, 2012, pp. 13-16.

[3] I. F. Atlam, R. J. Walters, and G. B. Wills, "Fog Computing and the Internet of Things: A Review," *Big Data and Cognitive Computing*, vol. 2, no. 2, 2018, Art. no. 10.

[4] L. Liu, D. Qi, N. Zhou, and Y. Wu, "A Task Scheduling Algorithm Based on Classification Mining in Fog Computing Environment," *Wireless Communication and Mobile Computing*, 2018, doi: 10.1155/2018/2102348.

[5] T. Aladwani, "Scheduling IoT Healthcare Tasks in Fog Computing Based on their Importance," *Procedia Computer Science*, vol. 163, pp. 560– 569, 2019.

[6] H. Rafique, M. Shah, S. Islam, T. Maqsood, S. Khan, and C. Maple, "A Novel Bio-Inspired Hybrid Algorithm (NBIHA) for Efficient Resource Management in Fog Computing," *IEEE Access*, vol. 7, pp. 115760 – 115773, 2019, doi: 10.1109/ACCESS.2019.2924958.

[7] Y. Nan, W. Li, W. Bao, F. C. Delicato, P. F. Pires, Y. Dou, and A. Y. Zomaya, "Adaptive Energy-Aware Computation Offloading for Cloud of Things Systems," *IEEE Access*, vol. 5, pp. 23947-23957, 2017.

[8] S. Kabirzadeh, D. Rahbari, and M. Nickray, "A Hyper Heuristic Algorithm for scheduling of Fog Networks," in *Proceedings of 21st Conference of Open Inno. Assoc*., 2017, pp. 148-155.

[9] B. M. Nguyen, T. T. B. Huynh, T. A. Tran, and B. S. Do, "Evolutionary Algorithms to Optimize Task Scheduling Problem for the IoT Based Bag-of-Tasks Application in Cloud–Fog Computing Environment," *Applied Science*, vol. 9, no. 9, 2019, Art. no. 1730.

[10] N. Kaur, A. Kumar, and R. Kumar, "A systematic review on task scheduling in Fog computing: Taxonomy, tools, challenges, and future directions," *Concurrency and Computation practice and experience*, vol. 33, no. 21, 2021, doi: 10.1002/cpe.6432.