湖南大学学报（自然科学版）
Journal of Hunan University (Natural Sciences)

第 50 卷 第 1 期
2023 年 1 月

Available online at http://jonuns.com/index.php/journal/index

Vol. 50 No. 1
January 2023

Open Access Article https://doi.org/10.55463/issn.1674-2974.50.1.7

Priority-Based Resource Allocation Scheme for Resources Usage in Mobile Edge Computing Platform

Zubair Sharif[1]*, Low Tang Jung[1], Muhammad Ayaz[2], Mazlaini Yahya[3], Shahneela Pitafi[1]

[1] Computer and Information Sciences Department, PETRONAS University of Technology, Seri Iskandar, 32610, Malaysia

[2] Sensor Networks and Cellular Systems (SNCS) Research Center, University of Tabuk, Tabuk, Saudi Arabia

[3] Head IoT Automation, Petronas, Malaysia

* Corresponding author: zubair_20000285@utp.edu.my

Received: November 15, 2022 / Revised: January 11, 2023 / Accepted: January 16, 2023 / Published: February 28, 2023

Abstract: Mobile edge computing offers cloud-like services at the edge of mobile networks to fulfill the escalating user demands for applications that are latency sensitive and require quick computation. However, this paradigm is confined to restricted resources; hence, an efficient and effective resource allocation strategy is necessary for optimum resource usage. Considering this fact, this article aims to describe a new method named the priority-based resource allocation scheme for efficient use of the available resources in this paradigm. Computing resources are allocated adaptively by considering the time-constraint nature of the incoming requests. The proposed approach shall adapt to meet the resource requirements and incoming requests' priorities to accomplish the task. After determining the received request type, which can be either a priority-based or ordinary request, each will be handled in one of three possibilities. At the edge node, available resources are managed in such a way as to handle the maximum number of incoming requests and the optimum use of scarce resources. Simulation results show that our proposed priority-based resource allocation scheme method outperforms the benchmarked schemes in average response time, average latency, resource usage, task execution time analysis, and energy consumption.

Keywords: priority-based resource allocation scheme, mobile edge computing, cloud, resource allocation, resource usage.

基於優先級的移動邊緣計算平台資源使用分配方案

摘要：移動邊緣計算在移動網絡邊緣提供類似雲的服務，以滿足用戶對延遲敏感且需要快速計算的應用程序不斷增長的需求。但是，這種範例僅限於有限的資源； 因此，為了優化資源使用，高效且有效的資源分配策略是必要的。考慮到這一事實，本文旨在描述一種名為基於優先級的資源分配方案的新方法，以有效利用該範例中的可用資源。 通過考慮傳入請求的時間限制性質，自適應地分配計算資源。提議的方法應適應資源需求和傳入請求的優先級以完成任務。在確定接收到的請求類型（可以是基於優先級的請求或普通請求）之後，每個請求都將以三種可能性之一進行處理。在邊緣節點，可用資源以處理最大數量的傳入請求和稀缺資源的最佳使用的方式進行管理。仿真結果表明，我們提出的基於優先級的資源分配方

案方法在平均響應時間、平均延遲、資源使用、任務執行時間分析和能耗方面優於基準方案。

关键词：基於優先級的資源分配方案，移動邊緣計算，雲，資源分配，資源使用。

1. Introduction

The number of wireless devices, including, smartphones, tablets, and other handy smart gadgets, has been swiftly mounted over the last decade [1]. The convenience and advancement of such wireless gadgets have led to numerous IoT-based applications, for instance, healthcare monitoring, smart hospitals, smart cities, autonomous vehicles, tracking systems, video surveillance, smart agriculture, and signal control and traffic cameras, etc. [2–4]. The aforementioned devices, IoT-based applications, mobile computing, and some other evolving technologies are all producing a massive amount of data. Some facts are mentioned here: according to professional predictions, there will be 75 billion Internet of Things (IoT) installed worldwide by 2025 [5–8]. In 2019, Cisco reported that data generated by machines, and human users have reached 500 zettabytes. Furthermore, a report published and funded by Seagate, is expecting 175 zettabytes of data to be produced yearly by 2025 [9–12]. As many mobile computing and IoT-based applications are demanding rapid and real-time response. Thus the mobile edge computing (MEC) is consequently offering the viable solutions to meet these demands (i.e., to help in governance of massive amount of data and to meet the restricted response required by the applications) [13, 14].

This new MEC paradigm offers a solution to the time-delay problems by moving the critical data processing to the end devices or at the edge of the network. Instead of depending on centralized processing, where data should deliver back to the server constantly, edge-based devices can accumulate and process more effectively the data in real time, hence can respond quicker and more effectively [15]. By doing so, not only the latency is reduced, but the amount of data can be exchanged in lower volumes hence consumes less bandwidth. This system provides a cost-efficient data processing mechanism [16–18].

According to researchers, using the MEC may lessen the energy consumption by 30 to 40 percent and improve response times by 80 to 200 milliseconds for the end devices [19]. Considering these benefits and use cases of the MEC, it is expected that approximately 75 percent of all data will be analyzed at the edge by 2025. Furthermore, many experts predicted that 45% of the data produced by IoT would be kept and managed by the MEC [20].

Along with the benefits mentioned above, edge-computing researchers and experts should be well conscious of various challenges, particularly system dependability and resource efficiency. These resource-restricted end devices could fail easier due to short battery life or scarce wireless communications. Although MEC differs from cloud computing, some security and privacy concerns still exist in this paradigm. These security concerns are largely linked to network topology when various inexpensive personal mobile devices are considered a part of the system infrastructure [21–24].

Fig. 1 illustrates the architecture of MEC, where the edge computing nodes are situated between the end devices and the cloud server [25]. It greatly boosts the response-time (due to the smaller distance between end devices) for the latency stringent applications. This paradigm has three tiers, first is the IoT/mobile devices layer, which comprises different end devices and various other sensors. The next (second) layer is the MEC layer, which contains Edge Nodes (ENs) and having various components including an EN manager, resources such as computing and storage, virtual machines (VMs). The third layer is the cloud layer, although cloud servers have more storage and processing power compared to the ENs, these resources are typically located far (physically) from the end/edge devices, which causes delays and latencies because of the huge distance. To alleviate the issue, MEC aims to lessen the delays by bringing cloud-like services nearer to end devices. Data processing and computation at the edge layer can be more interactive and more responsive for the runtime response required applications due to the shorter distance.

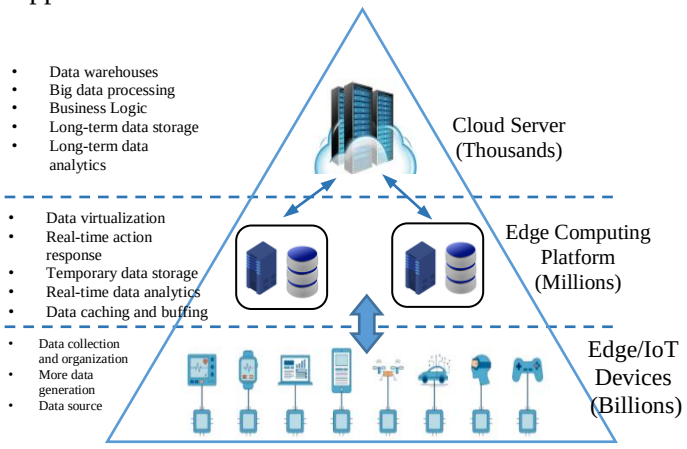

Fig. 1 Fundamental architecture of the mobile MEC paradigm

MEC considers the end users' applications that some may require real-time response, some may demand quick sensitive data exchanging, and others may rapidly generate a massive amount of data for fast data storing. Lacking in timely real-time response can be a critical cause leading to loss of life and lost capital

investments when crucial deadlines are not met due to latency restrictions posed by delays [26–28]. These are the challenges to be overcome within the MEC. Using this paradigm, end devices and IoT-based applications can offload their intensive computation tasks to satisfy the increasing demand for higher computational power and low latency. The accessibility of the scarce resources in MEC is often a challenge [29–31], so fulfilling the demands of application to have maximum allocation of the available resources is always very critical. The optimum usage of MEC-limited resources requires an intelligent mechanism in resource allocations.

The remaining paper is structured as follows: Section 2 discusses some of the prominent and closely related works. Section 3 provides the proposed priority-based resource allocation scheme (PBRAS) with details and necessary explanations. Section 4 discusses the simulation findings and the performance evaluations of the presented work. Section 5 concludes this paper.

2. Literature Review

Efficient resource management is one of the key research areas under the MEC paradigm. Many researchers have focused on this issue for achieving resource allocation efficiency. A broad range of resource allocation (RA) techniques and schemes have been suggested to deal with the challenges of resource management or RA in the MEC platform.

Effective resource usage plays a significant role in efficiently accomplishing MEC tasks. The authors in [32] proposed a technique to handle the challenges of computation offloading, content caching, and RA for MEC system. The presented method decays the primary problem into smaller problems and finds the solution in an efficient and dispersed way. Similarly, an approach is presented in [33], where the authors designed an RA algorithm by considering both the user's demands for resources and the number of VMs needed to perform the massive computational tasks. The suggested work was also applicable for the optimum selection of data centers in the MEC architecture and to lessen the maximum latency.

The authors presented a framework in [34] to address the resource optimization problem. In this proposed work, it involved network architecture, communication, and computing resource elements. Each base station (BS) simply should handle its own assigned tasks/problems without swapping to the other BSs. Consequently, the complex issues related to the signaling overhead problem can be considerably lessened. Similarly, a technique was offered in [35] to perform the RA efficiently for the MEC environment. The technique offers various services and resource packages to enhance the resources usage of MEC within certain budget restrictions. The proposed method ensures impartiality (fairness) during the resources sharing among the sets of end users and the edge service providers.

An effective RA effort to enhance the energy efficacy of end devices was offered in [36]. The proposed method was useful in reducing the energy dissipation for an offloading system. In the proposed work, two factors were considered: one is the task computation duration and the other is the energy expense for data transmission throughout the communication process. The authors in [37] offered a method to lessen the energy consumption and latencies in the end devices and the MEC server. In the suggested technique, the average amount of energy used to complete a task was deemed to be a performance evaluation metric. To minimize the energy consumption for offloading computational tasks by considering both the offloading decisions and data transmission, a strategy was developed in [36]. Additionally, the primary objective of this strategy was to minimize offloading system energy usage within the given delay restrictions. Another similar energy-efficient task scheduling and resource allocation scheme is presented in [38], in which the network traffic and the computation offloading strategy were jointly improved for the MEC system. In this technique, the interference issue between the mobile users was reduced. However, this method is an impractical model, as only a single mobile user was considered in this proposed system. Further, the offloaded data were susceptible to cyber-attacks.

A technique called as Zenith was offered in [39] to handle and perform the computation of RA in MEC. The intended algorithm enables resource-sharing contracts between the service providers (SPs) and the edge infrastructure suppliers. Based on the established agreements, SPs use a resource provisioning strategy that enables tasks to fulfill their delay restrictions. Although it is suitable for resource scheduling, it does not take requests' priority into account. A similar technique for the paradigm of edge or fog computing resources allocation was offered in [40]. In the proposed method, the authors presented a three-tiered model for efficient resource allocation. The proposed method is intended for task allocation to the end devices on the MEC server. The assessment benchmarks contained a trade-off between cost, user comfort, and the response time. The drawback of this technique is that it allocates resources before task processing or execution because there is no facility available for the runtime resources' allocation.

Many authors have paid attention to the efficient resources usage and to lessen the end-to-end delays. Some of the related works are discussed here considering this fact. A method was designed in [41] to decrease the end-to-end latency problem by performing the RA effectively. The authors also tried sorting out the problem of when computing resources are scarce by efficiently using them for task processing. By allocating the resources to the requested tasks

effectively, and enhances the resource efficiency is the target of the study presented in [42], where the authors proposed an algorithm to assign the set of tasks to the nodes where these nodes were situated at the edge of the network. The aims of this algorithm were to lessen the memory needed for the task execution and processing time at the edge nodes. The authors of [43] proposed another method to reduce the latency for mobile applications while considering the resources usage limits. Although their technique was good enough for latency minimization, they did not pay attention to the security issues as the application data were not secured from the attacks before being sent to the server. Furthermore, this method was limited to a single offloading request use case. Table 1 shows a concise comparison of IoT, edge, and cloud computing properties.

Table 1 Traits of IoT, edge, and cloud computing [13]

Characteristics	IoT	Edge	Cloud
Implementation	Decentralized	Decentralized	Centralized
Computational	Restricted	Restricted	Unrestricted
Elements	Physical devices	Edge nodes	Virtual resources
Storage	Very limited	Scarce	Unrestricted
Big data	Source	Process	Process
Response time	NA	Fast	Slow

The above-mentioned techniques have their own limitations and are not applicable, especially where the priority-based RA is required for the latency constraint applications. Some requests from different applications have to be prioritized based on application demands. Accepting the challenge and considering the above-mentioned situation, we have proposed a priority-based RA scheme to effectively use the limited resources in the MEC paradigm.

3. Methodology

For delay constraint applications, a local task computation in the end devices is a possible solution, but nearly all end/edge devices have restricted resources to satisfy the needs of several applications. MEC might be a viable solution to deal with such a condition. We are thus inspired to offer a method to attain an optimum RA and usage in the MEC paradigm. The key idea in our intended technique is that data is being produced at the end devices, e.g., smartphones etc., where they send requests to the edge nodes (ENs) for different resources for the data processing. This frequently happens because of the devices' own restricted resource capability for data processing and storage.

If the resource demand is not delay-constraint, and there is no EN accessible at the time of request, the end device can directly connect to the centralized cloud for cloud services. Otherwise, the request will be handled in accordance with the proposed method for resource usage and optimization, (expressed in further details in Fig. 2).

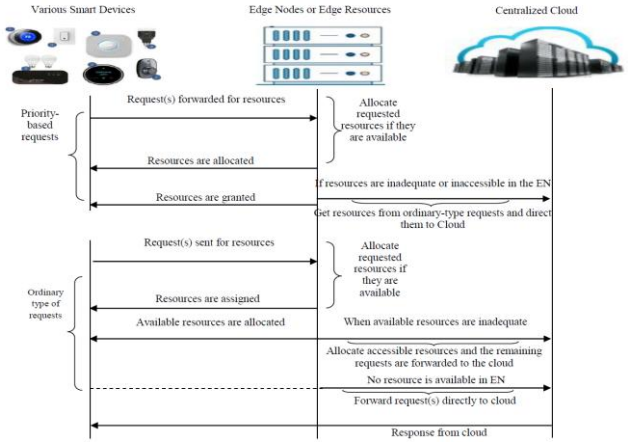

Fig. 2 Schematic for handling requests and allocating resources

To handle the accessible resources and manage the incoming requests, the following sets are deemed during this research work, whereby the ENs are ordered as $EN_l = \{en_1\ en_2, en_3,\ldots, en_n\}$ to perform and manage the allocation of various resources. End users are indicated by $E_u = \{eu_1, eu_2, eu_3, \ldots, en_n\}$ that rise requests for various resources. All the incoming requests are recorded in $R_r = \{P_r, N_r\}$, $R_r \in E_u$; and $P_r = \{pr_1, pr_2, pr_3, \ldots, pr_k\}$ handles priority-based or delay-sensitive requests. $N_r = \{nr_1, nr_2, nr_3, \ldots, nr_m\}$ is considered a set for the normal type or latency tolerable requests, and the last set $A_r = \{ar_1, ar_2, ar_3, \ldots, ar_n\}$ is used to manage the number of currently accessible resources at the EN.

Smart vehicles, ambulances, hospitals, etc., or any other end users can request resources. Each task for processing can be grouped by two considerations, and these are denoted as $T_k = \{\omega_k, \mu_k\}$, $T_k \in R_r$, where ω_k signifies the size of input data for computation to be sent to MEC (i.e., the amount of input data for offloading), and μ_k indicates the number of CPU cycles needed to execute that task.

The incoming requests are classified into two categories namely priority request P_r and normal request N_r. Shown below describes how to consider newly received requests R_r from the E_u. When a request R_r arrived, initially it is analyzed to be either as a priority-based request (pr_i) or an ordinary request type (nr_i). We presume that the priorities of requests/tasks are already defined that can depend on user or application demands so that these will be handled accordingly. When an end user eu_i intends to create a request for computation offloading, then it will select one option for any request from predetermined two options. As shown in Table 2, '0' represents P_r and '1' represents N_r, which will be recognized in the RA method. With an end user eu_i selecting the desired option, then the EN will assign the requested or available resources according to the selected action.

Table 2 Instances of priority-based and ordinary types of requests

"0" for priority-based request	"1" for normal type request
Ambulance	Movie
Hospital	Gamming
A patient police line	Watching video
Traffic signal	Any data related to
Fire brigade	entertainment

A newly received request will be projected to either 0 or 1 as per the end user eu_i choice stated in (1). The request mapping function is

$$R_r \longrightarrow [0,1] \quad (1)$$

s.t $pr_i \longrightarrow 0 \quad \forall\ pr_i \in P_r\ \&\ P_r \in R_r\ \&\ R_r \in E_u$

$nr_i \longrightarrow 1 \quad \forall\ nr_i \in N_r\ \&\ N_r \in R_r\ \&\ R_r \in E_u$

There might be three possibilities for RA for both categories of requests. Initially, let us start by considering the P_r requests. If the number of P_r are less than or equivalent to, A_r in this scenario, first the P_r will be calculated (in accordance with their limits pr_1 to pr_k) and their values will be added in P_r as by using the expression in (2). Then, likewise, the current available resources will be determined (counted) as per their limits and will be stored into A_r as represented in (3). Following these calculations, the A_r will be allotted to P_r which is stated as in (4). For this procedure, first both summation symbols will compute their values as per to their beginning and closing limits (pr_1 to pr_k and ar_1 to ar_n respectively). After these computations, then RA will be made to P_r as $\eta_{EN_{P_r}}$. Here, $\eta_{EN_{P_r}}$ denotes RA to P_r at EN. Further, condition C1 specifies that the overall RA at the EN cannot be outstripped to the EN's utmost resource limit.

$$P_r = \sum_{i=1}^{k} pr_i, \quad \text{s.t} \quad P_r \in R_r \quad (2)$$

$$A_r = \sum_{i=1}^{n} ar_i, \quad \text{s.t} \quad A_r \in RC_i \quad (3)$$

Case 1 (for priority-based requests):

$$(P_r \le A_r), \quad \text{s.t} \quad \forall\ P_r \in R_r\ \&\ \forall A_r \in RC_i$$

$$\sum_{i=1}^{k} pr_i \left(\sum_{i=1}^{n} ar_i\ \eta_{EN_{P_r}} \right) \le A_r \quad (4)$$

s.t C1: $0 \le \eta_{EN_{P_r}} \le RC_i$, where $P_r \in R_r$

In the second case, the number of P_r is greater than A_r. Here, before allocating the resources, a task can be divided into smaller tasks and these substituted tasks can be computed in sequence or parallel or in the mixture of both relying on the resources' accessibility. Similar to the previous situation (scenario), initially, P_r and A_r will be calculated using (2) and (3), respectively. In this case A_r are fewer than P_r, so the remaining demanded resources are obtained back from N_r and put into A_r as by using the expression in (5) and (6). After getting back the demanded resources, these are assigned to P_r by using the expression in (7) and the uncompleted N_r will be forwarded to the cloud data center by considering the expression in (8). By this approach, the utmost requests/tasks can be handled, and usage of resources can be assured efficiently.

Case 2 (for priority-based requests):

$$(P_r > A_r) \& (A_r \ne 0), \quad \text{s.t} \quad \forall\ P_r \in R_r\ \&\ \forall A_r \in RC_i$$

$$(P_r - A_r) = S \quad (5)$$

$$A_r \longleftarrow (N_r - S) \quad (6)$$

$$\sum_{i=1}^{k} pr_i \left(\sum_{i=1}^{n} ar_i\ \eta_{EN_{P_r}} \right) \quad (7)$$

$$S \longrightarrow \gamma_{CC} \quad (8)$$

s.t C2: $0 \le \eta_{EN_{P_r}} \le RC_i$, where $P_r \in R_r$

In the expression (8), symbol γ_{CC} indicates the uncompleted N_r (after getting back the resources), which are sent to the cloud.

The third possibility is that when P_r received at the EN, if there is no instant resource accessible (i.e., $A_r = 0$) to fulfill the demands of requests, then it is possible to manage the situation just like the second case scenario as mentioned above. In such exceptional instance, resources needed by the new P_r request(s) can be taken back from the ordinary type N_r request(s) (same way as scenario case 2 for P_r) and be assigned to the demanded P_r. This is applied in accordance with the expression in (11). These unfinished N_r will be directed to the cloud datacenter as stated by (12).

Case 3 (for priority-based requests):

$$(P_r > A_r) \& (A_r = 0), \quad \text{s.t} \quad \forall\ P_r \in R_r\ \&\ \forall A_r \in RC_i$$

$$(P_r - A_r) = S \quad (9)$$

$$A_r \longleftarrow (N_r - S) \quad (10)$$

$$\sum_{i=1}^{k} pr_i \left(\sum_{i=1}^{n} ar_i\ \eta_{EN_{P_r}} \right) \quad (11)$$

$$S \longrightarrow \gamma_{CC} \quad (12)$$

s.t C3: $0 \le \eta_{EN_{P_r}} \le RC_i$, where $P_r \in R_r$

The idea RA for P_r is relevant to the situations for the normal or ordinary requests (N_r) but with a few modifications. In the first situation, when A_r are greater than N_r; in this situation, first N_r will be computed (based on their limits nr_1 to nr_m) and their values added in N_r as represented by (13). Then the A_r will be determined as per their limits and the values will be added into A_r as expressed in (14). After these calculations, resources will be allotted to N_r according to the expression in (15) where both summations symbols will be compiled as per their limits (nr_1 to nr_m and ar_1 to ar_n respectively). Following these calculations, resources will be assigned to N_r as $\eta_{EN_{N_r}}$. Here $\eta_{EN_{N_r}}$ indicates RA to N_r at EN. Additionally, condition C4 specifies that the overall RA at the EN should be less than or equal to A_r and must not be surpassed to its upmost available resources.

$$N_r = \sum_{i=1}^{m} nr_i, \quad \text{s.t} \quad N_r \in R_r \tag{13}$$

$$A_r = \sum_{i=1}^{n} ar_i, \quad \text{s.t} \quad A_r \in RC_i \tag{14}$$

Case 1 (for normal type requests) :

$$(N_r \le A_r), \quad \text{s.t} \ \forall \ N_r \in R_r \ \ \& \ \ \forall A_r \in RC_i$$

$$\sum_{i=1}^{m} nr_i \left(\sum_{i=1}^{n} ar_i \ \eta_{EN_{N_r}} \right) \le A_r \tag{15}$$

s.t C4: $0 \le \eta_{EN_{N_r}} \le A_r$, where $N_r \in R_r$

The second case for N_r is when A_r is accessible but not sufficient to satisfy the demanding requests. In such case, the available resources will be assigned to N_r by implementing the expression as (16) and the remaining N_r will be forwarded to the cloud data center by applying the expression in (17).

Case 2(for normal type requests) :

$$(N_r > A_r) \& (A_r \ne 0), \quad \text{s.t} \ \forall N_r \in R_r \ \ \& \ \ \forall A_r \in RC_i$$

$$\sum_{i=1}^{m} nr_i \left(\sum_{i=1}^{n} ar_i \ \eta_{EN_{N_r}} \right) \ge A_r \tag{16}$$

$$\gamma_{CC_{N_r}} \longleftarrow \left(N_r - \eta_{EN_{N_r}} \right) \tag{17}$$

s.t C5: $0 \le \eta_{EN_{N_r}} \le A_r$, where $N_r \in R_r$

Algorithm 1 : Priority-based Resource Allocation Scheme (PBRAS) for MEC

Set of all requests $R_r = \{P_r, N_r\}$, $R_r \in E_u$
Set of priority-based requests $P_r = \{pr_1, pr_2, pr_3 \ldots pr_k\}$, $P_r \in R_r$
Set of normal type requests $N_r = \{nr_1, nr_2, nr_3 \ldots nr_m\}$, $N_r \in R_r$
Set of all resources at the EN, $RC_i = \{rc_1, rc_2, rc_3 \ldots rc_u\}$
Set of available resources $A_r = \{ar_1, ar_2, ar_3 \ldots ar_n\}$, $A_r \in RC_i$
Variables $(R_r, P_r, N_r, RC_i, A_r, \eta_{EN_{R_r}}, \gamma_{CC_{N_r}})$
Initialization:
Requests R_r received at EN
for each R_r **do**
  Identify the request type either it is pr_i or nr_i as Eq. (a)
  Compute P_r
    for P_r **do**
      $P_r \longleftarrow \sum_{i=1}^{k} pr_i$ s.t $\forall P_r \in R_r$
    end for
  Calculate N_r
    for N_r **do**
      $N_r \longleftarrow \sum_{i=1}^{m} nr_i$ s.t $\forall N_r \in R_r$
    end for
end for
for each RC_i **do**
  Count A_r
    $A_r \longleftarrow \sum_{i=1}^{n} ar_i$ s.t $\forall A_r \in RC_i$
end for
Check the type of request and perform resource allocation
if P_r **do**
  if $(P_r \le A_r)$ **do**
    $\sum_{i=1}^{k} pr_i \left(\sum_{i=1}^{n} ar_i \ \eta_{EN_{P_r}} \right) \le A_r$
  else if $(P_r > A_r)$ && $(A_r \ne 0)$ OR $(P_r > A_r)$ && $(A_r = 0)$ **do**
    $(P_r - A_r) = S$
    $A_r \longleftarrow (N_r - S)$
    $\sum_{i=1}^{k} pr_i \left(\sum_{i=1}^{n} ar_i \ \eta_{EN_{P_r}} \right)$
    $\gamma_{CC_{N_r}} \longleftarrow S$
  end if
else
  N_r **do**
  if $(N_r \le A_r)$ **do**
    $\sum_{i=1}^{m} nr_i \left(\sum_{i=1}^{n} ar_i \ \eta_{EN_{N_r}} \right) \le A_r$
  else if $(N_r > A_r)$ && $(A_r \ne 0)$ **do**
    $\sum_{i=1}^{m} nr_i \left(\sum_{i=1}^{n} ar_i \ \eta_{EN_{N_r}} \right) > A_r$
    $\gamma_{CC_{N_r}} \longleftarrow \left(N_r - \eta_{EN_{N_r}} \right)$
  else $(N_r > A_r)$ && $(A_r = 0)$
    $\gamma_{CC_{N_r}} \longleftarrow N_r$
  end if
end if
Output: PBRAS is performed

The third case for N_r is when there is no A_r accessible or no possibility for assigned resources to become available within the maximum wait time of requestors or requests for resources. In such situation all new receiving N_r will be forwarded to the cloud data center by using the expression in (18). Algorithm 1 represents the whole steps of the proposed PBRAS for the MEC system.

Case 3(for normal type requests) :

$$(A_r = 0), \quad \text{s.t} \ \forall N_r \in R_r \ \ \& \ \ \forall A_r \in RC_i$$

$$\gamma_{CC_{N_r}} \longleftarrow N_r \tag{18}$$

s.t C6: $0 \le \eta_{EN_{N_r}} \le A_r$, where $N_r \in R_r$

Algorithm 1 represents the whole steps of the proposed PBRAS for the MEC paradigm. The requests for resources are inputs, and the resource allocation has to meet the demanding requests for resources are the outputs. It follows the defined steps and processes as explained in the above scenarios for efficient and adaptive resources usage in the MEC paradigm.

4. Results and Discussion

Simulations were executed to assess the effectiveness of the intended PBRAS. Resources are allocated to the incoming requests in accordance with several particular parameters, such as request priority, the required resources, processing time, and the input data size.

The descriptions of the network link and processing nodes for the simulation framework are described in Tables 3 and 4, respectively. The iFogSim was used for simulation purpose. The reason for choosing iFogSim is that this simulator has been widely used by the scholarly community for various edge and fog computing-based simulations [44]. Fig. 3 depicts the graphical user interface (GUI) network layout for the simulation environment.

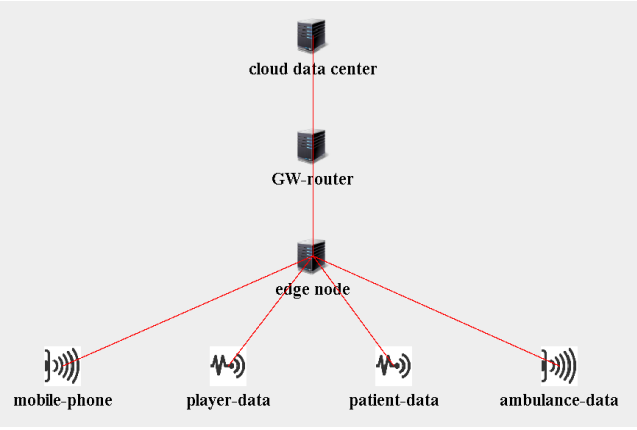

Fig. 3 GUI layout for simulation setting created through iFogSim

For the simulation topology, different components were setup. The entities' characteristics were specified during the construction of topology. Outcomes of the proposed PBRAS scheme were evaluated with different performance evaluation metrics. For essential comparison, PBRAS was compared with benchmark

algorithms such as First Come, First Served (FCFS), shortest job first (SJF) [45], and the Zenith technique [39].

Table 3 Network links description

Source node	Destination node	Latency in ms
GW-router	Cloud data center	100
Edge-node	GW-router	70
Amulance-data	Edge-node	50
Patient-data	Edge-node	45
Player-data	Edge-node	55
Mobille-phone	Edge-node	60

Table 4 Simulation of environment parameters

Parameters	Edge node	Cloud node
CPU MIPS	[500, 3000]	[2000,10000]
CPU usage	0.5	2.0 (Cloud VM)
RAM capacity	32 GB	64 GB

4.1. Average Latency

Fig. 4 depicts the average latency for the stringent and ordinary request types. During the simulations, the maximal delay tolerance was set to 450 ms and 250 ms for delay-tolerant and delay-restricted requests, respectively. Based on the achieved findings, it was observed that the outcomes for both request types were acceptable in terms of the highest latency tolerance limitations. The average latency rises steadily up to 350 ms when considering 100 requests without distinguishing their types. These findings show that PBRAS fulfills the request' maximum delay tolerance.

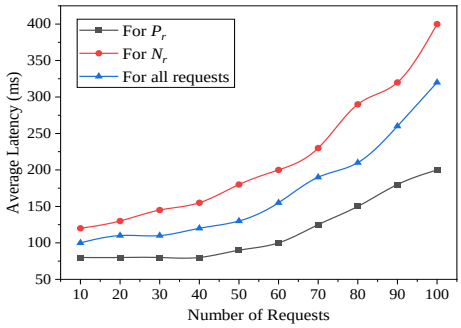

Fig. 4 Average latency by contemplating the number of requests

4.2. Average Response Time

Fig. 5 illustrates the assessment of the average response time for various numbers of arriving requests. When the number of tasks rose, the response time also surged for all methods. SJF and FCFS performed well when there were few requests, but as the number of requests increased, their response times were also noticeably affected. As FCFS assigns resources based on the arrival sequence, that's why incoming requests will continue to queue up or keep waiting if the required resources are not currently available. Further, in the case of the SJF algorithm, it prioritizes the shortest job; therefore, bigger requests confront higher waiting times for their executions. Whereas Zenith's average response time is smaller than the SJF and FCFS, which could be due to its suitable resource scheduling ability, but it does not take the priority of requests into account.

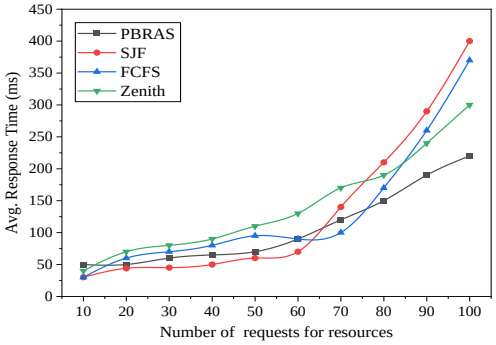

Fig. 5 Response time relative to the number of incoming requests

In contrast, in PBRAS, the tasks were arranged according to their priorities and the resources they required thus, it may react or respond to requests rapidly as far as feasible by considering the requests' nature. The overall response time of PBRAS is obviously presenting superiority over the compared techniques. Hence, the proposed PBRAS significantly performs better than the other conventional schemes by taking the average response time into account as the number of tasks increases.

4.3. Resource Usage

Fig. 6 demonstrates the proposed scheme's overall resource usage compared to the other techniques. Noticeable variances/fluctuations exist in resources usage in the other methods. FCFS suffers with uneven resources' usage since it allocates the resources without considering the required resources for their executions, execution time, and size of data that needs to be processed. There might be instances where requests could often need to wait even when they require processing a lower amount of data. Likewise, to the FCFS, the SJF also experiences irregular resources usage and is facing higher variations. Additionally, the Zenith algorithm has smaller fluctuations and achieves a better performance than the FCFS and SJF because of its resource scheduling ability.

Conversely, it appears that the PBRAS outperforms compared to the others in terms of resource usage. This indicates that more requests were assured. Resource management and the task scheduling strategies are the primary contributing aspects that eventually improve the resource usage.

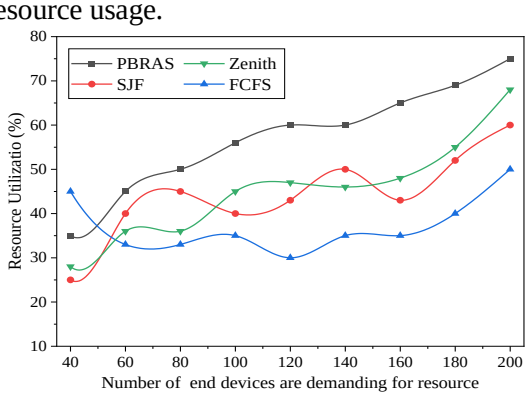

Fig. 6 Usage of resources when various end/edge devices are requesting resources at various times

4.4. Analysis of the Task Execution Time

In Fig. 7, we considered four distinct simulation configurations for four algorithms according to given tasks to assess the execution time. From the figure, in configuration settings 1 and 2, FCFS is facing more time than the SJF and Zenith, while in other cases it is varying. The SJF is taking more time than Zenith in configurations 2 and 3, while the situation is contrariwise in 1 and 4. In contrast, PBRAS is taking less time across all configuration iterations.

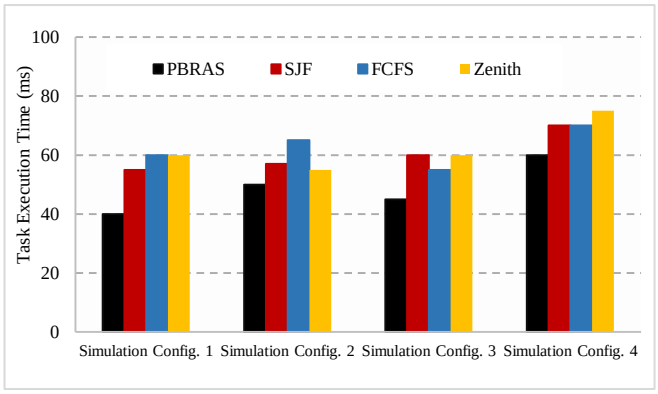

Fig. 7 Comparison of the task execution time of algorithms for four different configuration settings

Since FCFS assigns resources to the tasks without ensuring their amount of input data, processing time, or required resources, that's why it takes more execution time. In several situations where requests wait even, they are needed to process a smaller amount of data. Further, the situation is similar when we are considering SJF as it leads to boost in the waiting time due to a mismatch of the resource demands and accessibility. Furthermore, when we talk about Zenith, it leads to a rise in the cost of waiting time as it looks to find a perfect match between the demands and the availability of resources, but not on the priority of the request. Conversely, the scenario is different under the proposed PBRAS method as the management of resources and task scheduling is performed based on the nature and priority of each request, which eventually reduces the cost of processing and waiting time.

4.5. Energy Consumption

Fig. 8 depicts the evaluation performance of our proposed PBRAS by considering the energy consumption with respect to the number of receiving requests. From the figure, when the number of receiving requests is raised, the energy consumption also increases for the methods (because it needs more processing and more switching thus needing extra energy).

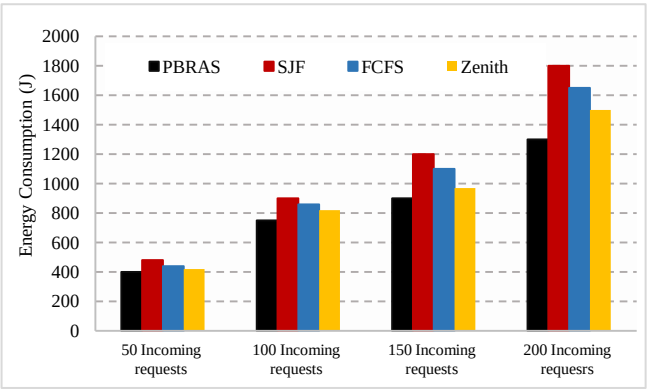

Fig. 8 Energy consumption in kilojoules with reference to incoming requests

The energy consumption is nearly equal when there are only 50 requests demanding for resources, but when this demand for resources is increasing, our proposed approach consumes less energy comparatively to others. This is because, unlike SJF and FCFS, it assigns resources to incoming requests and then manages the load on those resources, which makes it overall more efficient. Compared to Zenith, FCFS, and SJF, our proposed method has a lower energy consumption level for all sets of incoming requests.

5. Conclusion

In this paper, an effective priority-based resource allocation scheme (PBRAS) is proposed. The novelty of the proposed technique is that the received requests from end devices for resources are divided into two categories (i.e., priority-based requests and normal type requests). Three possibilities exist to deal with the priority-based requests. The first scenario is when all the demanded resources are available, and these are immediately allocated. The second and third situations are complicated when there are either insufficient or no resources available. Under these circumstances, the required resources can be taken back from the ordinary type of request(s) and allocated to the new priority request(s). Resulted of this, incomplete ordinary-type requests will be directed to the centralized cloud. The proposed approach is also capable when managing new requests from an ordinary request type or category and can be completed with three scenarios but with lesser modifications. The simulation results reveal that the proposed PBRAS enhances the usage of resources at the EC node and decreases the response time for the end devices due to its adaptive resources allocation ability.

Acknowledgment

This research was conducted in PETRONAS University of Technology under the YUTP grant scheme with the reference code of #RG2022-0754, the cost center of 015LC0-413, and the project title of "Edge Computing Oriented IoT Operation and IoT Resources Optimization."

References

[1] SHARIF Z., JUNG L.T., and AYAZ M. Priority-based Resource Allocation Scheme for Mobile Edge Computing. In: *2022 2nd International Conference on Computing and Information Technology (ICCIT)*, 2022: 138-143: IEEE.

[2] HASSAN N., GILLANI S., AHMED E., YAQOOB I., and IMRAN M. The role of edge computing in internet of things. *IEEE Communications Magazine*, 2018, 56(11): 110-115.

[3] AYAZ M., AMMAD-UDDIN M., SHARIF Z., MANSOUR A., and AGGOUNE E.-H.M. Internet-of-Things (IoT)-based smart agriculture: Toward making the fields talk. *IEEE Access*, 2019, 7: 129551-129583.

[4] JUNG L.T. IoT underwater wireless sensor network monitoring. In: *Role of IoT in Green Energy Systems*. IGI Global, 2021: 38-58.

[5] WANG X., HAN Y., LEUNG V.C., NIYATO D., YAN X., and CHEN X. Convergence of edge computing and deep learning: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 2020, 22(2): 869-904.

[6] SHI W., CAO J., ZHANG Q., LI Y., and XU L. Edge computing: Vision and challenges. *IEEE Internet of Things Journal*, 2016, 3(5): 637-646.

[7] SHARIF Z., JUNG L.T., RAZZAK I., and ALAZAB M. Adaptive and Priority-based Resource Allocation for Efficient Resources Utilization in Mobile Edge Computing. *IEEE Internet of Things Journal*, 2021. Ahead of print. https://doi.org/10.1109/JIOT.2021.3111838

[8] JUNG L.T., and HARUNA A.A. Incentive-Based Scheduling for Green Computational Grid. In: *Role of IoT in Green Energy Systems*. IGI Global, 2021: 272-293.

[9] COUGHLIN T. *175 Zettabytes By 2025*. 2021. https://www.forbes.com/sites/tomcoughlin/?sh=461818ad4498

[10] IQBAL S., SHARIF Z., SHAHID M.A., and ABBAS M.Z. Internet-of-Things based Home Automation System using Smart Phone. *Sir Syed University Research Journal of Engineering Technology*, 2021, 11(2).

[11] KAUR K., GARG S., KADDOUM G., AHMED S.H., and ATIQUZZAMAN M. KEIDS: Kubernetes-based energy and interference driven scheduler for industrial IoT in edge-cloud ecosystem. *IEEE Internet of Things Journal*, 2019, 7(5): 4228-4237.

[12] REINSEL D., GANTZ J., and RYDNING J. *The digitization of the world from edge to core*. Data Age 2025. 2018. https://www.readkong.com/page/the-digitization-of-the-world-from-edge-to-core-8666239

[13] SHARIF Z., JUNG L.T., AYAZ M., YAHYA M., and KHAN D. Smart Home Automation by Internet-of-Things Edge Computing Platform. *International Journal of Advanced Computer Science Applications*, 2022,13(4).

[14] ABBAS M.Z., BAKAR K.A., AYAZ M., MOHAMED M.H., and TARIQ M. Hop-by-hop dynamic addressing based routing protocol for monitoring of long range underwater pipeline. *KSII Transactions on Internet Information Systems*, 2017, 11(2): 731-763.

[15] RAHMAN A., HOSSAIN M.S., MUHAMMAD G., KUNDU D., DEBNATH T., RAHMAN M., KHAN M.S.I., TIWARI P., and BAND S.S. Federated learning-based AI approaches in smart healthcare: concepts, taxonomies, challenges and open issues. *Cluster Computing*, 2022: 1-41. Online ahead of print. DOI: 10.1007/s10586-022-03658-4.

[16] ELGENDY I.A., ZHANG W., TIAN Y.-C., and LI K. Resource allocation and computation offloading with data security for mobile edge computing. *Future Generation Computer Systems*, 2019, 100: 531-541.

[17] ABBAS M.Z., BAKAR K.A., AYAZ M., and MOHAMED M.H. An overview of routing techniques for road and pipeline monitoring in linear sensor networks. *Wireless Networks*, 2018, 24(6): 2133-2143.

[18] UDDIN M.A., AYAZ M., AGGOUNE E.-H.M., MANSOUR A., and LE JEUNE D. Affordable broad agile farming system for rural and remote area. *IEEE Access*, 2019, 7: 127098-127116.

[19] TRINH H., CHEMODANOV D., YAO S., LEI Q., ZHANG B., GAO F., CALYAM P., and PALANIAPPAN K. Energy-aware mobile edge computing and routing for low-latency visual data processing. *IEEE Transactions on Multimedia*, 2018, 20(10): 2562-2577.

[20] JAIN K., and MOHAPATRA S. Taxonomy of Edge Computing: Challenges, Opportunities, and Data Reduction Methods. In: *Edge Computing*. Springer, 2019: 51-69.

[21] SHAHZADI S., IQBAL M., DAGIUKLAS T., and QAYYUM Z.U. Multi-access edge computing: open issues, challenges and future perspectives. *Journal of Cloud Computing*, 2017, 6(1): 30.

[22] SODHRO A.H., PIRBHULAL S., and DE ALBUQUERQUE V.H.C. Artificial intelligence-driven mechanism for edge computing-based industrial applications. *IEEE Transactions on Industrial Informatics*, 2019, 15(7): 4235-4243.

[23] BAIG I., FAROOQ U., UL HASAN N., ZGHAIBEH M., RANA U., IMRAN M., and AYAZ M. On the PAPR reduction: A novel filtering based hadamard transform precoded uplink MC-NOMA scheme for 5G cellular networks. In: *2018 1st International Conference on Computer Applications & Information Security (ICCAIS)*, 2018: 1-4.

[24] AHMAD N., SHARIF Z., BUKHARI S., and AZIZ O. Insights Into Functional and Structural Impacts of nsSNPs in XPA-DNA Repairing Gene. *International Journal of Applied Research in Bioinformatics*, 2022, 12(1): 1-12.

[25] DUAN Z., TIAN C., ZHANG N., ZHOU M., YU B., WANG X., GUO J., and WU Y. A novel load balancing scheme for mobile edge computing. *Journal of Systems Software*, 2022, 186: 111195.

[26] KHAN M.A. A survey of security issues for cloud computing. *Journal of Network Computer Applications*, 2016, 71: 11-29.

[27] UDDIN M., AYAZ M., MANSOUR A., AGGOUNE E.-H.M., SHARIF Z., and RAZZAK I. Cloud-connected flying edge computing for smart agriculture. *Peer-to-Peer Networking Applications*, 2021,14(6): 3405-3415.

[28] YIMAM D.F., and EDUARDO B. A survey of compliance issues in cloud computing. *Journal of Internet Services Applications*, 2016, 7(1): 1-12.

[29] PELTONEN E., BENNIS M., CAPOBIANCO M., DEBBAH M., DING A., GIL-CASTIÑEIRA F., JURMU M., KARVONEN T., KELANTI M., KLIKS A., LEPPÄNEN T., LOVÉN L., MIKKONEN T., RAO A., SAMARAKOON S., SEPPÄNEN K., SROKA P., TARKOMA S., and YANG T. 6G White Paper on Edge Intelligence. *arXiv:2004.14850 [cs.DC]*, 2020. https://doi.org/10.48550/arXiv.2004.14850

[30] CHIANG M., and ZHANG T. Fog and IoT: An overview of research opportunities. *IEEE Internet of Things Journal*, 2016, 3(6): 854-864.

[31] BAIG I., AYAZ M., and JEOTI V. On the peak-to-average power ratio reduction in mobile WiMAX: A discrete cosine transform matrix precoding based random-interleaved orthogonal frequency division multiple access uplink system. *Journal of Network Computer Applications*, 2013, 36(1): 466-475.

[32] WANG C., LIANG C., YU F.R., CHEN Q., and TANG L. Computation Offloading and Resource Allocation in Wireless Cellular Networks With Mobile Edge Computing. *IEEE Transactions on Wireless Communications*, 2017, 16(8): 4924-4938.

[33] ZIAFAT H., and BABAMIR S.M. A method for the optimum selection of datacenters in geographically distributed clouds. *The Journal of Supercomputing*, 2017, 73(9): 4042-4081.

[34] ZHOU Y., YU F.R., CHEN J., and KUO Y. Resource Allocation for Information-Centric Virtualized Heterogeneous Networks With In-Network Caching and Mobile Edge Computing. *IEEE Transactions on Vehicular Technology*, 2017, 66(12): 11339-11351.

[35] NGUYEN D.T., LE L.B., and BHARGAVA V. Price-based Resource Allocation for Edge Computing: A Market Equilibrium Approach. *IEEE Transactions on Cloud Computing*, 2018, 9(1): 302-317.

[36] ZHANG K., MAO Y., LENG S., ZHAO Q., LI O., PENG X., PAN L., MAHARJAN S., and ZHANG Y. Energy-efficient offloading for mobile edge computing in 5G heterogeneous networks. *IEEE Access*, 2016, 4: 5896-5907.

[37] MAO Y., ZHANG J., SONG S.H., and LETAIEF K.B. Stochastic Joint Radio and Computational Resource Management for Multi-User Mobile-Edge Computing Systems. *IEEE Transactions on Wireless Communications*, 2017, 16(9): 5994-6009.

[38] KWAK J., KIM Y., LEE J., and CHONG S. DREAM: Dynamic resource and task allocation for energy minimization in mobile cloud systems. *IEEE Journal on Selected Areas in Communications*, 2015, 33(12): 2510-2523.

[39] XU J., PALANISAMY B., LUDWIG H., and WANG Q. Zenith: Utility-aware resource allocation for edge computing. In: *2017 IEEE international conference on edge computing (EDGE)*, 2017: 47-54.

[40] BINH H.T.T., ANH T.T., SON D.B., DUC P.A., and NGUYEN B.M. An evolutionary algorithm for solving task scheduling problem in cloud-fog computing environment. In: *Proceedings of the ninth international symposium on information and communication technology*, 2018: 397-404.

[41] REN J., YU G., CAI Y., and HE Y. Latency Optimization for Resource Allocation in Mobile-Edge Computation Offloading. *IEEE Transactions on Wireless Communications*, 2018, 17(8): 5506-5519.

[42] BITAM S., ZEADALLY S., and MELLOUK A. Fog computing job scheduling optimization based on bees swarm. *Enterprise Information Systems*, 2018, 12(4): 373-397.

[43] KAO Y.-H., KRISHNAMACHARI B., RA M.-R., and BAI F. Hermes: Latency optimal task assignment for resource-constrained mobile computing. *IEEE Transactions on Mobile Computing*, 2017, 16(11): 3056-3069.

[44] MAHMUD R., and BUYYA R. Modelling and simulation of fog and edge computing environments using iFogSim toolkit. In: *Fog and edge computing: Principles and paradigms*, 2019: 1-35.

[45] RAFIQUE H., SHAH M.A., ISLAM S.U., MAQSOOD T., KHAN S., and MAPLE C. A novel bio-inspired hybrid algorithm (NBIHA) for efficient resource management in fog computing. *IEEE Access*, 2019, 7: 115760-115773.

参考文:

[1] SHARIF Z.、JUNG L.T. 和 AYAZ M. 用於移動邊緣計算的基於優先級的資源分配方案。在：2022 年第二屆計算與信息技術國際會議，2022：138-143：電氣和電子工程師學會。

[2] HASSAN N.、GILLANI S.、AHMED E.、YAQOOB I. 和 IMRAN M。邊緣計算在物聯網中的作用。電氣和電子工程師學會 通訊雜誌, 2018, 56(11): 110-115.

[3] AYAZ M.、AMMAD-UDDIN M.、SHARIF Z.、MANSOUR A. 和 AGGOUNE E.-H.M. 基於物聯網的智能農業：讓田野對話。電氣和電子工程師學會 訪問, 2019, 7: 129551-129583.

[4] JUNG L.T. 物聯網水下無線傳感器網絡監測。在：物聯網在綠色能源系統中的作用。IGI 全球，2021 年：38-58。

[5] WANG X.、HAN Y.、LEUNG V.C.、NIYATO D.、YAN X. 和 CHEN X. 邊緣計算與深度學習的融合：綜合調查。電氣和電子工程師學會 通信調查與教程，2020 年，22(2)：869-904。

[6] SHI W.、CAO J.、ZHANG Q.、LI Y. 和 XU L. 邊緣計算：願景與挑戰。電氣和電子工程師學會 物聯網雜誌, 2016, 3(5): 637-646.

[7] SHARIF Z.、JUNG L.T.、RAZZAK I. 和 ALAZAB M. 移動邊緣計算中高效資源利用的自適應和基於優先級的資源分配。電氣和電子工程師學會 物聯網雜誌，2021 年。印刷前。https://doi.org/10.1109/JIOT.2021.3111838

[8] JUNG L.T. 和 HARUNA A.A. 綠色計算網格的基於激勵的調度。在：物聯網在綠色能源系統中的作用。IGI 全球，2021 年：272-293。

[9] COUGHLIN T. 175 到 2025 年為澤字節. 2021. https://www.forbes.com/sites/tomcoughlin/?sh=461818ad4498

[10] IQBAL S.、SHARIF Z.、SHAHID M.A. 和 ABBAS M.Z. 使用智能手機的基於物聯網的家庭自動化系統。賽義德爵士大學工程技術研究雜誌，2021 年，11(2)。

[11] KAUR K.、GARG S.、KADDOUM G.、AHMED S.H. 和 ATIQUZZAMAN M. 用於邊緣雲生態系統中工業物聯網的基於庫伯內斯的能源和乾擾驅動調度程序。電氣和電子工程師學會 物聯網雜誌, 2019, 7(5): 4228-4237.

[12] REINSEL D.、GANTZ J. 和 RYDNING J. 世界從邊緣到核心的數字化。數據時代 2025. 2018. https://www.readkong.com/page/the-digitization-of-the-world-from-edge-to-core-8666239

[13] SHARIF Z.、JUNG L.T.、AYAZ M.、YAHYA M. 和 KHAN D. 通過物聯網邊緣計算平台實現智能家居自動化。國際高級計算機科學應用雜誌, 2022,13(4).

[14] ABBAS M.Z.、BAKAR K.A.、AYAZ M.、MOHAMED M.H. 和 TARIQ M. 基於逐跳動態尋址的遠程水下管道監控路由協議。韓國互聯網信息學會互聯網信息系統交易，2017 年，11(2)：731-763。

[15] RAHMAN A.、HOSSAIN M.S.、MUHAMMAD G.、KUNDU D.、DEBNATH T.、RAHMAN M.、KHAN M.S.I.、TIWARI P. 和 BAND S.S. 智能醫療中基於聯邦

學習的人工智能方法：概念、分類法、挑戰和未解決的問題。集群計算，2022：1-41。先於印刷在線。DOI：10.1007/s10586-022-03658-4。
[16] ELGENDY I.A.、ZHANG W.、TIAN Y.-C. 和 LI K. 用於移動邊緣計算的具有數據安全性的資源分配和計算卸載。下一代計算機系統，2019，100：531-541。
[17] ABBAS M.Z. 、 BAKAR K.A. 、 AYAZ M. 和 MOHAMED M.H. 線性傳感器網絡中道路和管道監控的路由技術概述。無線網絡, 2018, 24(6): 2133-2143.
[18] UDDIN M.A.、AYAZ M.、AGGOUNE E.-H.M.、MANSOUR A. 和 LE JEUNE D. 適用於農村和偏遠地區的負擔得起的廣泛敏捷農業系統。電氣和電子工程師學會 訪問, 2019, 7: 127098-127116.
[19] TRINH H.、CHEMODANOV D.、YAO S.、LEI Q.、ZHANG B.、GAO F.、CALYAM P. 和 PALANIAPPAN K. 用於低延遲視覺數據處理的能量感知移動邊緣計算和路由. 電氣和電子工程師學會 多媒體彙刊，2018 年，20(10)：2562-2577。
[20] JAIN K. 和 MOHAPATRA S. 邊緣計算分類：挑戰、機遇和數據縮減方法。在：邊緣計算。施普林格，2019：51-69。
[21] SHAHZADI S.、IQBAL M.、DAGIUKLAS T. 和 QAYYUM Z.U. 多接入邊緣計算：未解決的問題、挑戰和未來前景。雲計算學報, 2017, 6(1): 30.
[22] SODHRO A.H. 、 PIRBHULAL S. 和 DE ALBUQUERQUE V.H.C. 基於邊緣計算的工業應用的人工智能驅動機制。電氣和電子工程師學會 工業信息學彙刊，2019，15(7)：4235-4243。
[23] BAIG I. 、 FAROOQ U. 、 UL HASAN N. 、 ZGHAIBEH M.、RANA U.、IMRAN M. 和 AYAZ M. 關於降低 PAPR：一種基於濾波的新型哈達瑪變換預編碼上行鏈路多載波非正交多址方案 5G 蜂窩網絡。在：2018 年第一屆計算機應用與信息安全國際會議，2018：1-4。
[24] AHMAD N.、SHARIF Z.、BUKHARI S. 和 AZIZ O. 洞察非同義單核苷酸多態性在著色性乾皮病-脫氧核糖核酸修復基因中的功能和結構影響。國際生物信息學應用研究雜誌, 2022, 12(1): 1-12.
[25] DUAN Z., TIAN C., ZHANG N., ZHOU M., YU B., WANG X., GUO J. 和 WU Y. 一種用於移動邊緣計算的新型負載均衡方案。系統軟件學報, 2022, 186: 111195.
[26] KHAN M.A. 雲計算安全問題調查。網絡計算機應用, 2016, 71: 11-29.
[27] UDDIN M. 、 AYAZ M. 、 MANSOUR A. 、 AGGOUNE E.-H.M.、SHARIF Z. 和 RAZZAK I. 用於智能農業的雲連接飛行邊緣計算。點對點網絡應用, 2021,14(6): 3405-3415.
[28] YIMAM D.F. 和 EDUARDO B. 雲計算合規性問題調查。互聯網服務應用雜誌, 2016, 7(1): 1-12.
[29] PELTONEN E., BENNIS M., CAPOBIANCO M., DEBBAH M., DING A., GIL-CASTIÑEIRA F., JURMU M., KARVONEN T., KELANTI M., KLIKS A., LEPPÄNEN T., LOVÉN L.、MIKKONEN T.、RAO A.、SAMARAKOON S. 、 SEPPÄNEN K. 、 SROKA P. 、 TARKOMA S. 和 YANG T. 6G 邊緣智能白皮書。arXiv:2004.14850 [cs.DC], 2020. https://doi.org/10.48550/arXiv.2004.14850
[30] CHIANG M. 和 ZHANG T. 霧與物聯網：研究機會概述。電氣和電子工程師學會 物聯網雜誌，2016 年，3(6)：854-864。
[31] BAIG I.、AYAZ M. 和 JEOTI V. 關於移動全球微波接入互操作性中峰均功率比的降低：基於隨機交錯正交頻分多址上行鏈路系統的離散餘弦變換矩陣預編碼。網絡計算機應用, 2013, 36(1): 466-475.
[32] WANG C.、LIANG C.、YU F.R.、CHEN Q. 和 TANG L. 具有移動邊緣計算的無線蜂窩網絡中的計算卸載和資源分配。電氣和電子工程師學會 無線通信彙刊，2017 年，16(8)：4924-4938。
[33] ZIAFAT H. 和 BABAMIR S.M. 一種在地理分佈的雲中優化選擇數據中心的方法。超級計算學報, 2017, 73(9): 4042-4081.
[34] ZHOU Y.、YU F.R.、CHEN J. 和 KUO Y. 具有網絡緩存和移動邊緣計算的以信息為中心的虛擬化異構網絡的資源分配。電氣和電子工程師學會車輛技術交易, 2017, 66(12): 11339-11351.
[35] NGUYEN D.T.、LE L.B. 和 BHARGAVA V. 基於價格的邊緣計算資源分配：市場均衡方法。電氣和電子工程師學會 雲計算彙刊，2018 年，9(1)：302-317。
[36] ZHANG K.、MAO Y.、LENG S.、ZHAO Q.、LI O.、PENG X.、PAN L.、MAHARJAN S. 和 ZHANG Y. 5G 異構移動邊緣計算的節能卸載 網絡。電氣和電子工程師學會 訪問，2016 年，4：5896-5907。
[37] MAO Y., ZHANG J., SONG S.H. 和 LETAIEF K.B. 多用戶移動邊緣計算系統的隨機聯合無線電和計算資源管理。電氣和電子工程師學會 無線通信彙刊，2017 年，16(9)：5994-6009。
[38] KWAK J.、KIM Y.、LEE J. 和 CHONG S. 移動雲系統中能量最小化的動態資源和任務分配。電氣和電子工程師學會通信選定領域期刊, 2015, 33(12): 2510-2523.
[39] XU J.、PALANISAMY B.、LUDWIG H. 和 WANG Q. 天頂：用於邊緣計算的效用感知資源分配。在：2017 年電氣和電子工程師學會 邊緣計算國際會議，2017：47-54。
[40] BINH H.T.T.、ANH T.T.、SON D.B.、DUC P.A. 和 NGUYEN B.M. 一種解決雲霧計算環境下任務調度問題的進化算法。見：第九屆信息與通信技術國際研討會論文集，2018：397-404。
[41] REN J.、YU G.、CAI Y. 和 HE Y. 移動邊緣計算卸載中資源分配的延遲優化。電氣和電子工程師學會 無線通信彙刊，2018 年，17(8)：5506-5519。
[42] BITAM S.、ZEADALLY S. 和 MELLOUK A. 基於蜂群的霧計算作業調度優化。企業信息系統, 2018, 12(4): 373-397.
[43] KAO Y.-H.、KRISHNAMACHARI B.、RA M.-R. 和 BAI F. 愛馬仕：資源受限移動計算的延遲最優任務分配。電氣和電子工程師學會 移動計算彙刊, 2017, 16(11): 3056-3069.
[44] MAHMUD R. 和 BUYYA R. 使用基於爪哇的開源模擬霧計算場景的工具工具包對霧和邊緣計算環境進行建模和仿真。在：霧和邊緣計算：原則和範例，2019：1-35。
[45] RAFIQUE H. 、 SHAH M.A. 、 ISLAM S.U. 、 MAQSOOD T.、KHAN S. 和 MAPLE C. 一種新型仿生混合算法，用於霧計算中的高效資源管理。電氣和電子工程師學會 訪問, 2019, 7: 115760-115773.