



## Paper

# Latency Focused Load Balancing Method between Cloudlets using Game theory

Yuki Yokota <sup>1</sup> and Sumiko Miyata <sup>1</sup>

<sup>1</sup> Shibaura Institute of Technology,  
3-7-5 Toyosu, Koto-ku, Tokyo, 135-8548 Japan

Received October 27, 2023; Revised December 29, 2023; Published July 1, 2024

**Abstract:** A cloudlet is a cluster of computers that exists within the same Local Area Network (LAN) and can be accessed by all users connected to the network in a single hop. Compared to the traditional edge computing system, where servers are located at base stations, cloudlets enable real-time communication with less network latency. On the other hand, the lack of computing power is another problem to be solved. Conventional research examines offloading jobs between cloudlets in order to reduce workload variance whilst ensuring that the total latency of each cloudlet remains below an acceptable level. However, the calculation of total latency is dependent on the fraction of offloaded jobs, which results in an unfair decision regarding latency for a smaller fraction of jobs within the system. It is necessary to explore fairer solutions for offloading decisions in cloudlet systems. In this paper, we present a method for load balancing that prioritises reducing latency, utilising game theory. We assess the method's effectiveness by comparing the differences in utilisation between cloudlets.

**Key Words:** Cloud Computing, Cloudlet, Offloading, Queuing Theory, Game Theory

## 1. Introduction

In order to compensate for the problems faced by mobile devices, such as limited power, storage capacity and computing power, traditional mobile cloud computing technologies have been developed. In this context, the increasing number of applications requiring an average latency of about 10-100 ms, such as virtual reality, online gaming and tele-operation, has led to the need for solutions using edge computing techniques to reduce latency [1]. The cloudlet proposed by Mahadev et al. [2] is a cluster of computers that reside on the same LAN as users and can be connected via a single hop. Unlike edge computing in a large network, where a computing server is installed at a base station, a cloudlet can communicate with users with lower latency by installing it at a router or other location within the LAN.

One problem with cloudlets is that they are small and have much less processing power than traditional cloud computing or edge computing in a larger network. Therefore, the limited resources of cloudlets need to be used efficiently. Conventional research in wireless access networks has focused on the problem of cloudlet placement to distribute resources efficiently, such as the [3] method. In distributed computing systems, it is necessary to determine which computer will process each incoming job in order to distribute the load across the system [4]. For this purpose, methods such as [5] and



[6] have been proposed to determine the allocation of job requests from mobile terminals to cloudlets based on average latency. However, given the complex movement patterns of mobile terminal users and the randomness of arrival rates, it is inevitable to have both highly concentrated cloudlets and empty cloudlets. It is therefore necessary to consider a method of distributing jobs across cloudlets to enable efficient use of resources and reduce load dispersion [7].

Sourav et al. [8] proposed a method for offloading jobs between cloudlets, noting that user satisfaction does not change as long as the latency is below a certain level. Each fraction of jobs to be offloaded is determined so that each cloudlet processes as many jobs as possible while keeping the average latency of user jobs below the set acceptable latency. This is expected to maximize the use of the limited cloudlet resources while maintaining user satisfaction. On the other hand, this method calculates the latency weighted by the fraction of offloaded jobs to meet the acceptable latency, which results in the latency of jobs with low fraction not effectively affecting the results. Therefore, no load balancing is performed to limit the latency of offloaded jobs.

In the research of Yokota and Miyata [9], to consider the latency of offloaded jobs, they proposed an alternative utility function that directly evaluates the latency of each fraction of jobs. This allows to calculate the real latency of each fraction of jobs instead of the weighted latency in the conventional method, and could partially achieve a better load balancing compared to the method of Sourav et al. [8]. However, full load balancing is not yet achieved because the main focus of the proposed utility function is to increase utilization below acceptable latency.

In this paper, we introduce another utility function that focuses on limiting latency below an acceptable threshold to achieve complete load balancing of cloudlets by offloading. We will evaluate the effectiveness of our proposed method of load balancing by measuring the difference in actual utilization among cloudlets that can be derived using the optimal offload fraction. The contributions of this paper are:

1. The optimal offload fraction for each arriving job at cloudlet is proposed.
2. Building a non-cooperative load balancing game which calculates the optimal offload fraction with acceptable latency.
3. The numerical experiments indicate that the suggested method accomplishes absolute load balancing of cloudlets.

This paper is organized as follows. Section 2 describes related studies on load balancing of cloudlet resources. Section 3 models the system and constructs a method for determining the offload fraction of jobs based on game theory. After clarifying the existence of a Nash equilibrium for the game, we present an algorithm for deriving Nash equilibrium of the proposed game. Section 4 shows the results of the numerical analysis, and Section 5 summarizes the conclusions and issues in this paper.

## 2. Related Research

A conventional method for distributing the load of cloudlets [3] determines access points where cloudlets are installed according to the density of users in the area, and the allocation decisions are made so that the average latency for each user is the smallest. This affects the cost and number of cloudlets installed due to the trade-offs. In the studies [5], [6], it only determines the allocation of job requests from mobile terminals to cloudlets based on average latency. Many of the above studies, such as the cloudlet placement decisions [3], and job allocation [6], have only been analysed in limited environments because it is difficult to account for the randomness of job arrival rates due to the complex mobility patterns of real users.

Recent research by [10], and [11] has focused on differentiating user jobs to effectively offload and balance the load between users as well as in the servers. It is important to focus on what types of jobs are offloaded, as the difference in user jobs is directly related to the processing time at the servers, but these cases have narrow scopes and lack scalability.

To find the optimal solution to the problem, mixed integer programming is used in [3], and game theory based approach is used in [5], [8], [10], and [11]. In this paper, we assume that there are multiple

providers maintaining each cloudlet. Thus, cloudlets will not cooperate with each other to achieve a globally optimal solution, but will act selfishly to maximize their individual utility. Therefore, the use of non-cooperative game theory to find the optimal solution is appropriate for this topic.

Sourav et al. [8] focuses on offloading jobs between cloudlets and uses queueing theory to theoretically model the randomness of users. Sourav et al. also set a bound on the average latency for processing each job in terms of acceptable latency rather than minimising it, and use non-cooperative game theory to determine the offload fraction so that the utilisation at each cloudlet is uniformly maximized. However, the latency is weighted according to the offload fraction of each job, and the latency of smaller fractions of jobs is not fully reflected in the results. This raises a problem of fairness between offload fractions. Research by Yokota and Miyata [9] proposed an alternative utility function to value the latency of offloaded jobs more than the conventional method [8] to balance the risk of each offload fractions. As a result, this balanced offloading method achieved partial improvements in load balancing. In this paper, we further investigate the effect of the latency focused utility function based on the balanced offloading method [9] to improve in the load balancing aspect.

### 3. Proposed method

#### 3.1 System model

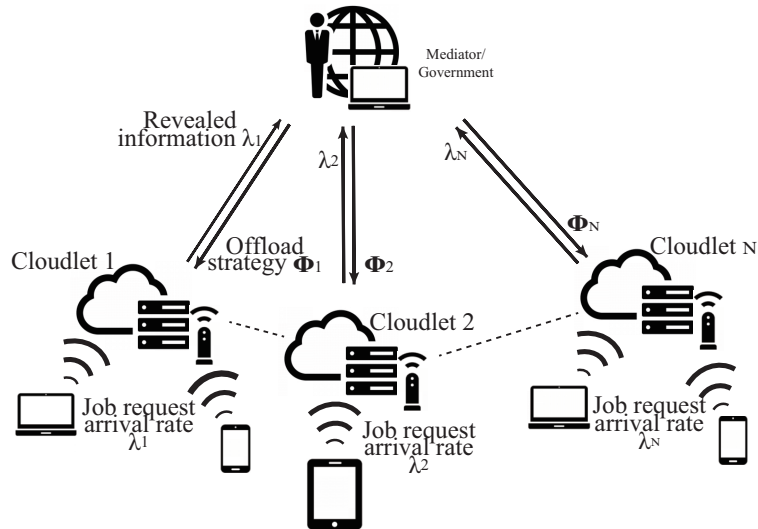


Fig. 1. System model.

When a user's job arrives at cloudlet  $i \in \{1, 2, \dots, N\} = \mathcal{C}$ ,  $N \geq 2$ , it is first determined whether it is offloaded or not. Jobs that are not offloaded are processed as they arrive at the cloudlet  $i$  and the results are returned to the user. If offloaded, the job is sent to another cloudlet where it is processed and returned to the user via the original arriving cloudlet  $i$ . We assume that cloudlets receive some profit from processing jobs. We also assume that all cloudlets have known information about each other's processing power and are aware of each other's existence as competitors [9].

Based on the general cluster data, we can assume that the arrival rate of jobs sent by users to servers follows a Poisson distribution and the service rate of servers follows an exponential distribution, so their behavior is a Poisson process. Therefore, we model the job arrival system using the  $M/M/1$  model for each cloudlet.

Let  $\lambda_i$ ,  $i \in \mathcal{C}$  [jobs/s] be the average arrival rate of jobs in cloudlet  $i$  following a Poisson distribution, and let  $\mu$  [jobs/s] be the average service rate following an exponential distribution. To determine the offload destination for jobs arriving in cloudlet  $i$ , we define the fraction vector  $\varphi_i = (\varphi_{i1}, \varphi_{i2}, \dots, \varphi_{iN}) \in \mathbb{R}^N$ ,  $\varphi_{ij} = [0, 1] \subset \mathbb{R}$ ,  $\sum_{j=1}^N \varphi_{ij} = 1$ ,  $i, j \in \mathcal{C}$ . In this case,  $\varphi_{ii}$  is the fraction of jobs that the  $i$ -th cloudlet offloads to the  $j$ -th cloudlet. Thus, jobs in  $\varphi_{ii}\lambda_i$  are processed in cloudlet  $i$  without being offloaded, and jobs in  $\varphi_{ij}\lambda_i$  are offloaded to cloudlet  $j$ . This gives the actual arrival rate  $\lambda_i^* = \varphi_{ii}\lambda_i + \sum_{j=1, j \neq i}^N \varphi_{ji}\lambda_j$  at cloudlet  $i$  after all the jobs have been offloaded. Fig.1 shows

the system model of how user's job requests are transferred among each cloudlet. Information of  $\lambda_i$  for all the cloudlet is collected at the mediator, and the offloading strategy  $\varphi_i$  for each cloudlet is determined based on this information.

The average round-trip time between users and cloudlets is defined as  $t_{\text{prop}}$ , and the average round-trip time between individual cloudlets is defined as  $t_{ij}, i \in \mathcal{C}, j \in \mathcal{C} \setminus \{i\}$ . Using the parameters  $\lambda_i, \mu, \varphi_i$  the average processing latency at each cloudlet can be calculated using Little's formula, which derives the average latency for each fraction of jobs  $T_{ij}, i, j \in \mathcal{C}$ .

The average latency  $T_{ii}$  for jobs  $\varphi_{ii}\lambda_i$  that are not offloaded is defined as

$$T_{ii} = t_{\text{prop}} + \frac{1}{\mu - \varphi_{ii}\lambda_i - \sum_{j=1, j \neq i}^N \varphi_{ji}\lambda_j}. \quad (1)$$

The average latency of offloaded jobs  $\varphi_{ij}, j \neq i$  can be expressed as follows.

$$T_{ij} = t_{\text{prop}} + t_{ij} + \frac{1}{\mu - \varphi_{ij}\lambda_i - \sum_{k=1, k \neq i}^N \varphi_{kj}\lambda_k}. \quad (2)$$

### 3.2 Non-Cooperative game theory

In this paper, the offload fraction  $\varphi_i$  of cloudlet  $i$  and the offload fraction of other cloudlets  $\varphi_{-i} = (\varphi_1, \dots, \varphi_{i-1}, \varphi_{i+1}, \dots, \varphi_i, \varphi_{-i})$ ,  $i \in \mathcal{C}$  are the strategy of each cloudlet participating as a player in the load balancing game,  $\mathcal{C}, i \in \mathcal{C}$ . From the strategy set, the utility function  $U_i(\varphi_i, \varphi_{-i})$ ,  $i \in \mathcal{C}$  is defined and the each cloudlet maximizes their utility non-cooperatively by choosing its best strategy. The optimal offload fraction  $\varphi_i^*$  that maximizes the utility is determined using game theory.

The utility of a cloudlet  $i$  is assumed to depend on the initial arrival rate  $\lambda_i$ , multiplied by the price value  $\Omega_1$ . At the same time, for every  $\sum_{j=1}^N \varphi_{ji}\lambda_j$  of jobs offloaded by other cloudlets and for every offloaded job  $\sum_{j=1}^N \varphi_{ij}\lambda_i$  cloudlet  $i$  receives a reward of  $\Omega_2$  and gives away the same amount as a penalty for each offloading job  $\sum_{j=1}^N \varphi_{ij}\lambda_i$ . We also define  $\zeta$  and  $\eta$  as the cost weights required to process jobs in cloudlets in order to reproduce reasonable offloading from cloudlets.

If  $T_{ij}$  does not satisfy the acceptable latency  $D_{ij}$ ,  $i, j \in \mathcal{C}$ , penalty of  $\Omega_3$  is given accordingly, to make the computed offloading strategy satisfy the latency.

Using the above and by considering that the diagonal component can be replaced by  $\varphi_{ii} = (1 - \sum_{j=1, j \neq i}^N \varphi_{ij})$  from the definition of the offload fraction  $\varphi_{ij}$ , the utility function for cloudlet  $i$  is given as follows [9].

$$\begin{aligned} U_i(\varphi_i, \varphi_{-i}) = & \Omega_1 \frac{\lambda_i}{\mu} + \Omega_2 \sum_{j=1, j \neq i}^N \varphi_{ji} \frac{\lambda_j}{\mu} \\ & - \Omega_2 \sum_{j=1, j \neq i}^N \varphi_{ij} \frac{\lambda_i}{\mu} - \left\{ \zeta \left[ \frac{(1 - \sum_{j=1, j \neq i}^N \varphi_{ij})\lambda_i + \sum_{j=1, j \neq i}^N \varphi_{ji}\lambda_j}{\mu} \right] + \eta \right\} \\ & - \frac{\Omega_3}{N} \frac{\lambda_i}{\mu} \left\{ \left( t_{\text{prop}} + \frac{1}{\mu - (1 - \sum_{j=1, j \neq i}^N \varphi_{ij})\lambda_i - \sum_{j=1, j \neq i}^N \varphi_{ji}\lambda_j} - D_{ii} \right) \right. \\ & \left. + \left( \sum_{j=1, j \neq i}^N \left[ t_{\text{prop}} + t_{ij} + \frac{1}{\mu - \varphi_{ij}\lambda_i - \sum_{k=1, k \neq i}^N \varphi_{kj}\lambda_k} - D_{ij} \right] \right) \right\}. \end{aligned} \quad (3)$$

Each cloudlet strategically determines its offload fraction to optimize its utility, aiming for incremental utility gains through the offloading process. Therefore, it is not necessary to consider that offloading gives a lower utility than no offloading at all. If the utility without any offloading is  $U_i^0$ ,  $i \in \mathcal{C}$ , the following inequality condition can be defined.

$$\begin{aligned} U_i(\varphi_i, \varphi_{-i}) \geq U_i^0 = & \Omega_1 \frac{\lambda_i}{\mu} - \left\{ \zeta \left[ \frac{\lambda_i}{\mu} \right] + \eta \right\} \\ & - \Omega_3 \frac{\lambda_i}{\mu} \left\{ t_{\text{prop}} + \frac{1}{\mu - \lambda_i} - D_{ii} \right\}. \end{aligned} \quad (4)$$

Moreover, as the anticipated systems are in a steady-state, the solution can only be derived when the utility function meets the condition  $[\mu - (1 - \sum_{j=1, j \neq i}^N \varphi_{ij})\lambda_i - \sum_{j=1, j \neq i}^N \varphi_{ji}\lambda_j] \geq 0, i \in \mathcal{C}, j \in \mathcal{C} \setminus \{i\}$ .

Since the aim of  $U_i$  proposed in the balanced offloading method [9] is to limit latency within acceptable levels and maximize the utilization of cloudlets participating, we suggest an alternative utility function  $U_{\text{latency},i}$  focusing solely on the latency aspect to provide an optimal load balancing solution.

$$U_{\text{latency},i}(\varphi_i, \varphi_{-i}) = - \left\{ (t_{\text{prop}} + \frac{1}{\mu - (1 - \sum_{j=1, j \neq i}^N \varphi_{ij})\lambda_i - \sum_{j=1, j \neq i}^N \varphi_{ji}\lambda_j} - D_{ii}) + (\sum_{j=1, j \neq i}^N \left[ t_{\text{prop}} + t_{ij} + \frac{1}{\mu - \varphi_{ij}\lambda_i - \sum_{k=1, k \neq i}^N \varphi_{kj}\lambda_k} - D_{ij} \right]) \right\}. \quad (5)$$

The minimum condition for  $U_{\text{latency},i}$  will be derived as follows.

$$U_{\text{latency},i}(\varphi_i, \varphi_{-i}) \geq U_{\text{latency},i}^0 = -\{t_{\text{prop}} + \frac{1}{\mu - \lambda_i} - D_{ii}\}. \quad (6)$$

Other conditions of  $U_{\text{latency},i}$  will remain unchanged from the conditions of utility function  $U_i$ . This indicates that the solution for  $U_{\text{latency},i}$  can be derived using the same algorithm as the calculation for  $U_i$ .

Using game theory, the load balancing game proposed in this work can be modelled as  $\Gamma = (\mathcal{C}, \{\theta_i\}_{i \in \mathcal{C}}, \{U_i(\varphi_i, \varphi_{-i})\}_{i \in \mathcal{C}})$ . The cloudlet strategies set is symbolised by  $\Theta = (\theta_i = \{\varphi_i : U_i(\varphi_i, \varphi_{-i}) \geq U_i^0\}, i \in \mathcal{C})$ . Cloudlet  $i$  selects a strategy from the set that yields higher than  $U_i^0$ . Hence, the game  $\Gamma$  is considered as a Generalized Nash Equilibrium (GNE) problem as each player's strategy influences others' set of strategies. The computation of a solution for GNE problem is challenging. Under specific conditions, GNE problem can be substituted with a Variational Inequality (VI) problem. If the VI problem has a unique solution, then the solution corresponds with the Nash equilibrium of the substituted GNE [8].

### 3.3 Proof of VI problem and Nash Equilibrium

In this section, we aim to prove that the game  $\Gamma$  can be transformed into a VI problem. We assume a VI problem expressed as  $\mathbf{VI}(\Psi, F)$ , where  $F = \nabla_{\varphi_i}(U_i(\varphi_i, \varphi_{-i}))$ . Here,  $\Psi$  represents the solution set of  $\mathbf{VI}(\Psi, F)$ , denoted as  $\Psi = \Theta \subseteq [0, 1]^{N \times N}$ . Let  $\Psi$  be a bounded closed set, making it a compact set in the finite-dimensional space  $\mathbb{R}^{N \times N}$ . Additionally, it is trivially a convex set. If we can prove that the solution space of the game  $\Gamma$  expressed as  $\Theta$  is equivalent to  $\Psi$ , then the game  $\Gamma$  can be transformed into  $\mathbf{VI}(\Psi, F)$ .

First, we show that  $U_i(\varphi_i, \varphi_{-i})$  is convex over  $\varphi_i$ . To demonstrate the convexity of  $U_i(\varphi_i, \varphi_{-i})$  under steady-state condition of  $[\mu - (1 - \sum_{j \neq i} \varphi_{ij})\lambda_i - \sum_{j \neq i} \varphi_{ji}\lambda_j] \geq 0, i \in \mathcal{C}, j \in \mathcal{C} \setminus \{i\}$ , we calculate the first-order partial derivatives.

$$\begin{aligned} \frac{\partial U_i}{\partial \varphi_{ij}} &= -\Omega_2 \frac{\lambda_i}{\mu} + \zeta \frac{\lambda_i}{\mu} \\ &\quad - \frac{\Omega_3}{N} \frac{\lambda_i}{\mu} \left\{ \left( \frac{-\lambda_i}{(\mu - (1 - \sum_{j=1, j \neq i}^N \varphi_{ij})\lambda_i - \sum_{j=1, j \neq i}^N \varphi_{ji}\lambda_j)^2} \right) \right. \\ &\quad \left. + \left( \frac{\lambda_i}{(\mu - \varphi_{ij}\lambda_i - \sum_{k=1, k \neq i}^N \varphi_{kj}\lambda_k)^2} \right) \right\}, \end{aligned} \quad (7)$$

$$\begin{aligned} \frac{\partial U_i}{\partial \varphi_{ji}} &= \Omega_2 \frac{\lambda_j}{\mu} - \zeta \frac{\lambda_j}{\mu} \\ &\quad - \frac{\Omega_3}{N} \frac{\lambda_i}{\mu} \left\{ \left( \frac{\lambda_j}{(\mu - (1 - \sum_{j=1, j \neq i}^N \varphi_{ij}) \lambda_i - \sum_{j=1, j \neq i}^N \varphi_{ji} \lambda_j)^2} \right) \right. \\ &\quad \left. + \left( \frac{-\lambda_j}{(\mu - \varphi_{ij} \lambda_i - \sum_{k=1, k \neq i}^N \varphi_{kj} \lambda_k)^2} \right) \right\}, \end{aligned} \quad (8)$$

$$\frac{\partial U_i}{\partial \varphi_{jk}} = \frac{\Omega_3}{N} \frac{\lambda_i}{\mu} \left\{ \frac{\lambda_j}{(\mu - \varphi_{ij} \lambda_i - \sum_{k=1, k \neq i}^N \varphi_{kj} \lambda_k)^2} \right\}. \quad (9)$$

Subsequently, the Hessian matrix elements can be derived using variables  $\varphi_{ij}$ ,  $\varphi_{ji}$ ,  $\varphi_{jk}$ ,  $i \in \mathcal{C}$ ,  $j \in \mathcal{C} \setminus \{i\}$ ,  $k \in \mathcal{C} \setminus \{i, j\}$  as follows.

$$\begin{aligned} \frac{\partial^2 U_i}{\partial \varphi_{ij}^2} &= -2 \frac{\Omega_3}{N} \frac{\lambda_i}{\mu} \left\{ \left( \frac{\lambda_i^2}{(\mu - (1 - \sum_{j=1, j \neq i}^N \varphi_{ij}) \lambda_i - \sum_{j=1, j \neq i}^N \varphi_{ji} \lambda_j)^3} \right) \right. \\ &\quad \left. + \left( \frac{\lambda_i^2}{(\mu - \varphi_{ij} \lambda_i - \sum_{k=1, k \neq i}^N \varphi_{kj} \lambda_k)^3} \right) \right\} < 0. \end{aligned} \quad (10)$$

$$\begin{aligned} \frac{\partial^2 U_i}{\partial \varphi_{ij} \partial \varphi_{ik}} &= -2 \frac{\Omega_3}{N} \frac{\lambda_i}{\mu} \left\{ \left( \frac{\lambda_i^2}{(\mu - (1 - \sum_{j=1, j \neq i}^N \varphi_{ij}) \lambda_i - \sum_{j=1, j \neq i}^N \varphi_{ji} \lambda_j)^3} \right) \right\} \\ &< 0. \end{aligned} \quad (11)$$

$$\begin{aligned} \frac{\partial^2 U_i}{\partial \varphi_{ji}^2} &= -2 \frac{\Omega_3}{N} \frac{\lambda_i \lambda_j}{\mu} \left\{ \left( \frac{\lambda_j}{(\mu - (1 - \sum_{j=1, j \neq i}^N \varphi_{ij}) \lambda_i - \sum_{j=1, j \neq i}^N \varphi_{ji} \lambda_j)^3} \right) \right. \\ &\quad \left. + \left( \frac{\lambda_j}{(\mu - \varphi_{ij} \lambda_i - \sum_{k=1, k \neq i}^N \varphi_{kj} \lambda_k)^3} \right) \right\} < 0. \end{aligned} \quad (12)$$

From Eq.(10)-(12), it is confirmed that at steady-state, each component of the Hessian matrix is always negative. Consequently, the utility function  $U_i(\varphi_i, \varphi_{-i})$  is concave upwards, and the solution space  $\Theta$  of the game  $\Gamma$  is also a convex set. Therefore, we can conclude that the game  $\Gamma$  can be transformed into  $\mathbf{VI}(\Psi, F)$  because both solution space  $\Phi$  and  $\Theta$  are closed and compact set.

Next, we prove that the replaced VI problem has a unique solution. The optimal response functions for cloudlet  $i$ ,  $\frac{\partial U_i}{\partial \varphi_{ij}} = 0$  are complex due to the interdependence of variables  $\varphi_{ij}$ ,  $i \in \mathcal{C}$ ,  $j \in \mathcal{C} \setminus \{i\}$ . Therefore, we initially examine the monotonicity for the case of  $N = 2$ .

The second-order derivatives of the utility functions  $U_1$  and  $U_2$  for the case of  $N = 2$  are shown as follows.

$$\begin{aligned} \frac{\partial^2 U_1}{\partial \varphi_{12}^2} &= -2 \frac{\Omega_3}{N} \frac{\lambda_1}{\mu} \left\{ \left( \frac{\lambda_1^2}{(\mu - (1 - \varphi_{12}) \lambda_1 - \lambda_2)^3} \right) \right. \\ &\quad \left. + \left( \frac{\lambda_1^2}{(\mu - \varphi_{12} \lambda_1 - (1 - \varphi_{21}) \lambda_2)^3} \right) \right\} < 0. \end{aligned} \quad (13)$$

$$\begin{aligned} \frac{\partial^2 U_2}{\partial \varphi_{21}^2} &= -2 \frac{\Omega_3}{N} \frac{\lambda_1 \lambda_2}{\mu} \left\{ \left( \frac{\lambda_2}{(\mu - (1 - \varphi_{12}) \lambda_1 - \varphi_{21} \lambda_2)^3} \right) \right. \\ &\quad \left. + \left( \frac{\lambda_2}{(\mu - \varphi_{12} \lambda_1 - (1 - \varphi_{21}) \lambda_2)^3} \right) \right\} < 0. \end{aligned} \quad (14)$$

$$\begin{aligned} \frac{\partial^2 U_1}{\partial \varphi_{12} \partial \varphi_{21}} &= \frac{\Omega_3}{N} \frac{\lambda_1 \lambda_2}{\mu} \left\{ \left( \frac{\lambda_1}{(\mu - (1 - \varphi_{12}) \lambda_1 - \varphi_{21} \lambda_2)^3} \right) \right. \\ &\quad \left. + \left( \frac{\lambda_1}{(\mu - \varphi_{12} \lambda_1 - (1 - \varphi_{21}) \lambda_2)^3} \right) \right\} > 0. \end{aligned} \quad (15)$$

---

**Algorithm 1** Projection Algorithm with Constant Step Size

---

- 1: **Initialization:** Choose any Lagrange multipliers  $\alpha^0, \beta^0, \xi^0 \geq 0$ , step size  $\omega > 0$ , and tolerance limit  $\epsilon > 0$ . Set the index  $t = 0$ .
- 2: **Output:** NE of the computation offload game  $\varphi^*$ .
- 3: If  $\alpha^t, \beta^t$ , and  $\xi^t$  satisfies a desirable tolerance limit: STOP
- 4: Given  $\alpha^t, \beta^t$ , and  $\xi^t$ , compute  $\varphi^t(\alpha^t, \beta^t, \xi^t)$  as the NE of the GNE problem (19)-(22) with fixed Lagrange multipliers  $\alpha = \alpha^t, \beta = \beta^t$ , and  $\xi = \xi^t$ .
- 5: Update all the Lagrange multipliers: for all  $i, j \in \mathcal{C}$ , compute

$$\begin{aligned}\alpha_{ij}^{t+1} &= [\alpha_{ij}^t - \omega(\varphi_{ij})]^+, \quad i \neq j \\ \beta_{ij}^{t+1} &= [\beta_{ij}^t - \omega(1 - (\varphi_{ij}))]^+, \quad i \neq j \\ \xi_{ij}^{t+1} &= [\xi_{ij}^t - \omega(U_i - U_i^0)]^+, \end{aligned}$$

where  $[z]^+ = \max\{0, z\}$ .

- 6: Set  $t \leftarrow t + 1$ ; go to Step 3.
- 

$$\begin{aligned}\frac{\partial^2 U_2}{\partial \varphi_{21} \varphi_{12}} &= \frac{\Omega_3}{N} \frac{\lambda_1 \lambda_2}{\mu} \left\{ \left( \frac{\lambda_2}{(\mu - (1 - \varphi_{12})\lambda_1 - \varphi_{21}\lambda_2)^3} \right) \right. \\ &\quad \left. + \left( \frac{\lambda_2}{(\mu - \varphi_{12}\lambda_1 - (1 - \varphi_{21})\lambda_2)^3} \right) \right\} > 0. \end{aligned} \quad (16)$$

Using the implicit function theorem and Eq.(13)-(16), we demonstrate the monotonicity of  $\frac{\varphi_{12}}{\varphi_{21}}$  and  $\frac{\varphi_{21}}{\varphi_{12}}$  as follows.

$$\frac{\varphi_{12}}{\varphi_{21}} = - \frac{\left( \frac{\partial^2 U_2}{\partial \varphi_{21} \varphi_{12}} \right)}{\left( \frac{\partial^2 U_1}{\partial \varphi_{12}^2} \right)} > 0. \quad (17)$$

$$\frac{\varphi_{21}}{\varphi_{12}} = - \frac{\left( \frac{\partial^2 U_1}{\partial \varphi_{12} \varphi_{21}} \right)}{\left( \frac{\partial^2 U_2}{\partial \varphi_{21}^2} \right)} > 0. \quad (18)$$

Thus, we conclude that the optimal response functions for the two Cloudlets are monotonically increasing. For the case with  $N > 2$ , the utility functions can be considered as a linear extension of the utility functions for  $N = 2$  case. Therefore, the optimal response functions for all Cloudlets in the game  $\Gamma$  are monotonically increasing. Consequently,  $\mathbf{VI}(\Psi, F)$  has a unique solution which corresponds to the Nash equilibrium of the game  $\Gamma$  with optimal offload fraction  $\varphi^*_i$  [12], [13].

### 3.4 Solving for optimal solution

To solve for the optimal solution  $\varphi^*_i$ , using the Lagrange multipliers  $\alpha_i \in \mathbb{R}^{N-1}$ ,  $\beta_i \in \mathbb{R}^{N-1}$ ,  $\xi_i \in \mathbb{R}^N$ ,  $i \in \mathcal{C}$  the Karush-Kuhn-Tucker(KKT) conditions for cloudlet  $i$  can be defined as follows.

$$\nabla_{\varphi_i} U + \nabla_{\varphi_i} \left( \alpha_i^T \varphi_i + \beta_i^T (1 - \varphi_i) + \xi_i^T (U_i - U_i^0)_{i \in \mathcal{C}} \right), \quad (19)$$

$$\alpha_i^T \varphi_i = 0, \quad (20)$$

$$\beta_i^T (1 - \varphi_i) = 0, \quad (21)$$

$$\xi_i^T (U_i - U_i^0) = 0. \quad (22)$$

Using the conditions above the optimal solution  $\varphi^*_i$  can be derived using gradient projection algorithm described in Algorithm 1 [8].

As previously mentioned, utility function  $U_i$  can be replaced by  $U_{\text{latency},i}$  considering its mathematical conditions. Therefore, its solution can also be derived by Algorithm 1.



## 4. Numerical Analysis

### 4.1 Parameters

In order to demonstrate the effectiveness of the proposed method, the following numerical analysis is performed on Matlab. Each weight parameter is based on the conventional method [8] with  $\Omega_1 = 5 \times 10^2$ ,  $\Omega_2 = 1 \times 10^6$ ,  $\Omega_3 = 5 \times 10^8$ ,  $\zeta = 300$ ,  $\eta = 700$ . The processing rate of the cloudlet is set to  $\mu = 10000$  jobs/s, and  $t_{\text{prop}} = 2$  ms,  $t_{12} = 0.75$  ms, and  $D_{ij} = 5$  ms are defined.

We evaluate the the optimal offloading fraction between  $N = 2$  cloudlets with the arrival rate of cloudlet 2,  $\lambda_2 = 9000$  jobs/s and the arrival rate of cloudlet 1, from  $\lambda = 0$  jobs/s to  $\lambda = 9000$  jobs/s. Since  $\mu$  is fixed, by setting the initial arrival rate of cloudlet 2 to  $\lambda_2 = 9000$  jobs/s the utilization before offloading is fixed at 0.9. Likewise, the initial arrival rate  $\lambda_1$  is set between 0 to 9000 jobs/s which varies the utilization of cloudlet 1 before offloading from 0 to 0.9. Since the processing rate remains constant, the initial utilization is determined by the initial arrival rate. In other words, analyzing the change in the initial utilization rate is the same as analyzing the change in the initial arrival rate. By observing changes in the initial arrival rate, the effectiveness of the method for various user environments can be obtained. Under such conditions, we evaluate the results in terms of difference in actual utilization  $\mu_{\text{diff}}$  after offloading which is calculated from the actual arrival rate  $\lambda_1^*$ , and  $\lambda_2^*$  after offloading.

$$\mu_{\text{diff}} = \left| \frac{\lambda_2^* - \lambda_1^*}{\mu} \right|. \quad (23)$$

By analyzing the dependence of  $\mu_{\text{diff}}$  on the changes of initial utilization of cloudlet 1, we will be able to evaluate the level of load balancing of different methods in various system environments.

First, we show the performance of  $U_i$  proposed by Yokota and Miyata. [9] to show its need of improvement. Next, we calculate the difference in actual utilization after offloading using proposed utility function  $U_{\text{latency},i}$  shown in equation (5) to evaluate the effectiveness of our method in terms of load balancing. After that, we evaluate the optimal offloading fraction  $\varphi_{12}^*$ ,  $\varphi_{21}^*$  for each initial utilization of cloudlet 1 to identify the difference between proposed method and conventional method [8] in terms of offloading. Each case is derived using Algorithm 1, and the results are compared to the conventional method [8].

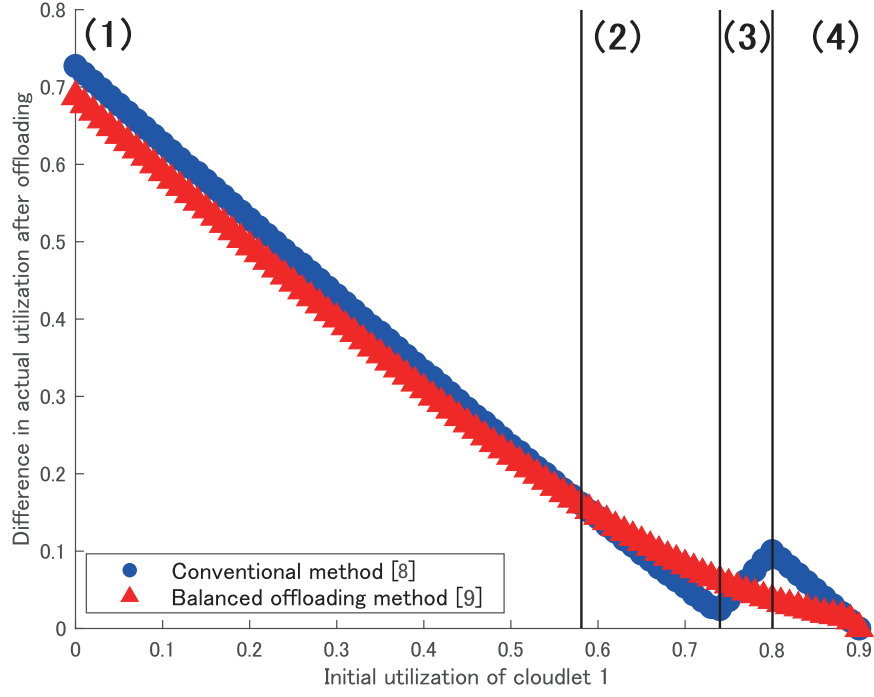
Although the general scenario assumes a multiple cloudlet environment, this paper focuses on evaluating the effectiveness of both methods through numerical analysis of offloading between two cloudlets with  $N = 2$ . The degree of load distribution can be assessed by the difference in the load of each cloudlet. As discussed previously, the game  $\Gamma$  between multiple cloudlets is a linear extension of the game between two cloudlets. Although the solution of each game is unique, it is likely to vary linearly. Accordingly, a correlation between the difference in load among two cloudlets and the difference in load among multiple cloudlets is expected to be similar. Therefore, this analysis only compares the results between two cloudlets.

### 4.2 Results

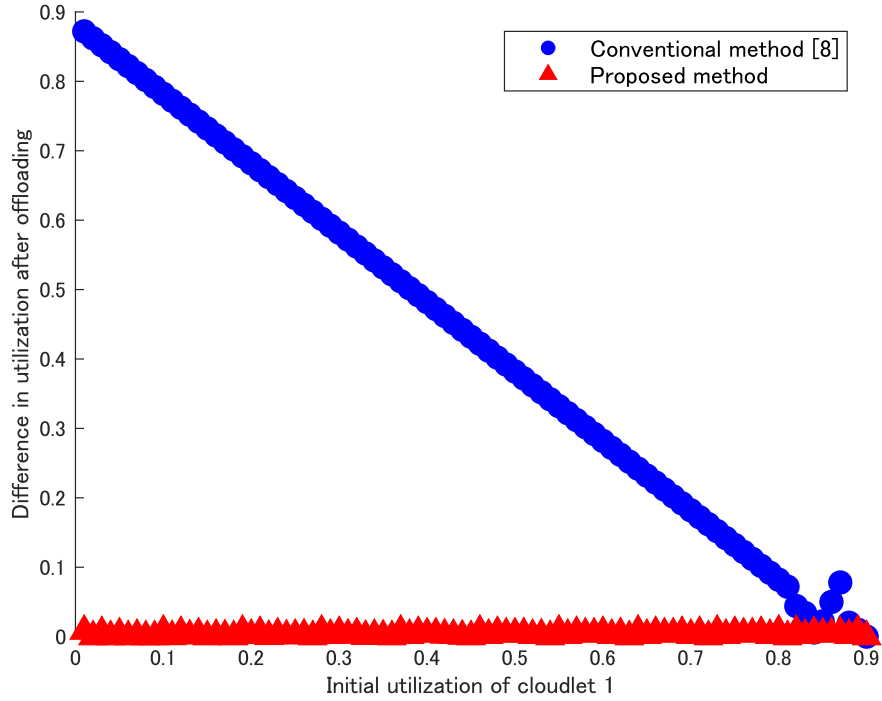
Fig.2 shows the comparison of the conventional method [8] and the balanced offloading method [9] in terms of  $\mu_{\text{diff}}$  between two cloudlets.

In the area (1) shown on Fig.2 the proposed method performs better in terms of load balancing. This is because the proposed method evaluates the latency of each fraction of jobs individually, which balances the load for both cloudlets. In the area (2), the conventional method performs slightly better due to the difference in the amount of acceptable latency applied to the utility function. However, in the area (3) the results of the conventional method starts to increase. The exponential increase in the latency of offloaded jobs is directly reflected on the utility function of conventional method which decreases the offloading fraction until it reaches to a point where no offloading will occur. Consequently, in the area (4) no offloading will take place for the conventional method and the proposed method will start to outperform because it continues to offload even in highly crowded environment. Overall, the latency of all the jobs in the system are kept below acceptable latency.



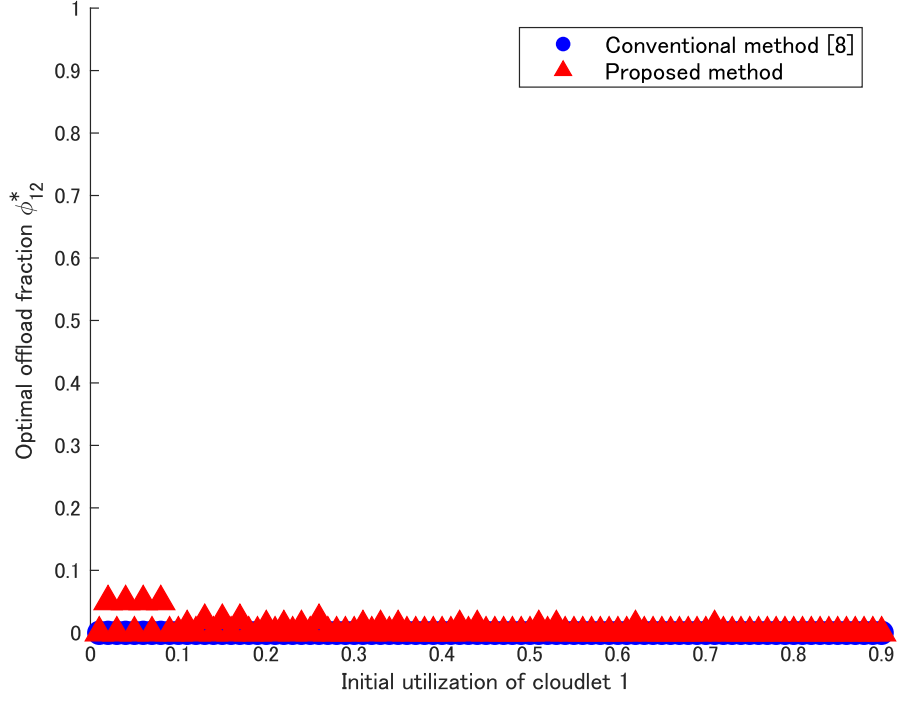


**Fig. 2.** Difference in actual utilization after offloading for different initial utilization of cloudlet 1

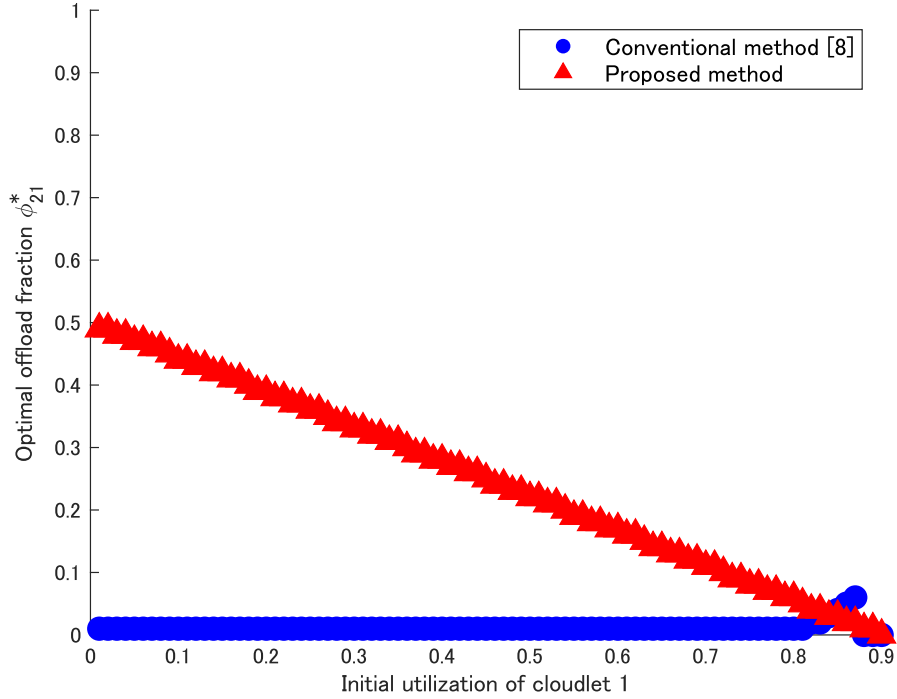


**Fig. 3.** Difference in actual utilization after offloading  $\mu_{diff}$  for different initial utilization of cloudlet 1  $U_{latency,i}$

Fig.3 shows the comparison of the conventional method and the proposed method using  $U_{latency,i}$  in terms of utilization difference after offloading between two cloudlets. From this result, it is obvious that our proposed utility function that only focused on the latency achieves a complete load balancing



**Fig. 4.** Optimal offload strategy  $\phi_{12}^*$



**Fig. 5.** Optimal offload strategy  $\phi_{21}^*$

as its difference between utilization is maintained around 0 at any load of cloudlet 1. Game theoretic solution achieves a compromised point where the utility differences made by the changes of strategy become the least for all players. This means that for  $U_{\text{latency},i}$  the changes of how much the total latency does not exceed the acceptable latency is at its least value in the result. Therefore, since we model the arrival process in  $M/M/1$  queuing model and assume only the steady-state environment, if

we only focus on the latency its optimal strategy will be the point where all the cloudlets have equal amount of arriving jobs, which results in a complete load balancing shown in Fig.3.

Fig.4 displays the optimal offloading fraction  $\varphi_{12}^*$  in Fig.3 for different initial utilization rates of cloudlet 1. As shown in the graph, the optimal offload fraction is almost zero for both the proposed and conventional method[8]. At low values of initial utilization, the proposed method shows a slight deviation in results, which is attributed to a computational error caused by the use of Matlab. The result can be concluded that both the conventional method[8] and the proposed method do not offload the cloudlet with the lower initial utilization rate.

Figure5 illustrates the optimal offload fraction  $\varphi_{21}^*$  in Fig.3 for different initial utilization rates of cloudlet 1. The results indicate that the conventional method [8] only offloads to a limited extent and eventually offloads in environments with uniformly high initial utilization rates. In contrast, the proposed method exhibits a linear decrease in the offload fraction from 0.5 and, conversely, more offloading in environments with varying initial utilization rates.

We can conclude that our proposed method in this paper can offload more jobs than conventional methods [8]. This results in a smaller difference between the utilization of the two cloudlets while still meeting the acceptable latency.

## 5. Conclusion

This paper has contributed on the optimal load balancing of cloudlets below acceptable latency by modelling the offloading decision making system using queuing theory and game theory. Compared to the conventional research [8] we focused on evaluating the latency of each fraction of jobs individually to improve the load balancing between cloudlets. We modified the balanced offloading method [9] to focus only on the latency of cloudlets. The result showed that the proposed method has managed to determine the offloading strategy in various system environment and outperforms the conventional method, and the modified utility function can be used to achieve a complete load balancing between cloudlets by offloading.

In the future work, we plan to extend the queuing model used to demonstrate the offloading mechanism in the current game to real-life scenarios. Moreover, investigating the difference between cooperative and non-cooperative games in cloudlets could help to further develop our proposed method in terms of offloading efficiency.

## Acknowledgments

These research results were obtained from the commissioned reserach(No.JPJ012368C05601) by National Institute of Information and Communications Technology (NICT), Japan. This work was supported by JSPS KAKENHI Grant Numbers 19K11947, 22K12015, 22H00247.

## References

- [1] E. Wong, M. Pubudini Imali Dias, and L. Ruan, "Predictive resource allocation for tactile internet capable passive optical lans," *Journal of Lightwave Technology*, vol.35, no.13, pp.2629–2641, 2017.
- [2] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case for VM-based cloudlets in mobile computing," *IEEE Pervasive Computing*, vol.8, no.4, pp.14–23, 2009.
- [3] Q. Fan and N. Ansari, "Cost aware cloudlet placement for big data processing at the edge," *IEEE International Conference on Communications* pp.1–6, 2017.
- [4] A. Yousefpour, C. Fung, T. Nguyen, K. Kadiyala, F. Jalali, A. Niakanlahiji, J. Kong, and J. P. Jue, "All one needs to know about fog computing and related edge computing paradigms: A complete survey," *Journal of Systems Architecture*, vol.98, pp.289–330, 2019.
- [5] H. Cao and J. Cai, "Distributed multiuser computation offloading for cloudlet-based mobile cloud computing: A game-theoretic machine learning approach," *IEEE Transactions on Vehicular Technology*, vol.67, no.1, pp.752–764, 2018.
- [6] Q. Fan and N. Ansari, "Application aware workload allocation for edge computing-based iot," *IEEE Internet of Things Journal*, vol.5, no.3, pp.2146–2153, 2018.

- [7] Y. Jiang, "A survey of task allocation and load balancing in distributed systems," *Transactions on Parallel and Distributed Systems*, vol.27, no.2, pp.585–599, 2016.
- [8] S. Mondal, G. Das, and E. Wong, "A game-theoretic approach for non-cooperative load balancing among competing cloudlets," *IEEE Open Journal of the Communications Society*, vol.1, pp.226–241, 2020.
- [9] Y. Yokota and S. Miyata, "Game Theoretic Approach for Non-Cooperative Load Balancing between Local Cloudlets" *Proc. NOLTA'23*, paper ID (6207), September 2023.
- [10] D. Madeo, S. Mazumdar, C. Mocenni, and R. Zingone, "Evolutionary game for task mapping in resource constrained heterogeneous environments," *Future Generation Computer Systems*, vol.108, pp.762–776, 2020.
- [11] T. Fang, J. Chen, and Y. Zhang, "Content-Aware Multi-Subtask Offloading: A Coalititon Formation Game-Theoretic Approach," *IEEE Communications Letters*, vol.25, no.8, pp.2664–2668, 2021.
- [12] F. Facchinei, A. Fischer, V. Piccialli, "On generalized Nash games and variational inequalities," *Operations Research Letters*, vol.35, pp.19–164, March 2007.
- [13] G. Scutari, D. Palomar, F. Facchinei, J. Pang, "Convex Optimization, Game Theory, and Variational Inequality Theory," *IEEE Signal Processing Magazine*, vol.27, pp.35–49, 2010.