Journal of Advanced Research in Applied Sciences and Engineering Technology

Journal homepage:
https://semarakilmu.com.my/journals/index.php/applied_sciences_eng_tech/index
ISSN: 2462-1943

Improvement of Energy Consumption in Fog Computing via Task Offloading

Wan Norsyafizan W. Muhamad[1,*], Audrey Celine Ribep[1], Kaharudin Dimyati[2], Azita Laily Yusof[1], Ezmin Abdullah[1]

[1] School of Electrical Engineering, College of Engineering, Universiti Teknologi MARA, 40450 Shah Alam, Selangor, Malaysia
[2] Department of Electrical Engineering, Faculty of Engineering, University Malaya, 50603 Kuala Lumpur, Malaysia

ARTICLE INFO

Article history:
Received 4 August 2023
Received in revised form 20 November 2023
Accepted 2 December 2023
Available online 30 December 2023

Keywords:
Task offloading; Gale Shapley; fog nodes; energy consumption; latency

ABSTRACT

The role of fog computing is to lessen the burden of Cloud and device computing. The purpose of this project is to solve the energy consumption and latency issues in the IoT 6G device as well as propose an approach to match the fog nodes with the IoT devices in task-offloading. This project implemented the Gale Shapley Stable Marriage Matching Game with a task-offloading energy efficiency technique. One-to-one matching is applied, in four fog nodes and four IoT devices according to the size of the task in IoT devices and energy in fog nodes. Simulation results prove that the proposed approach improves energy consumption and latency by 36.15% and 37.5% respectively.

1. Introduction

Fog computing is a viable technique for future 6G networks that can provide storage and computational capabilities [1]. Fog computing will be critical in supporting 6G's enormous Internet of Things (IoT) applications [2]. The primary goal of the 6G network is to link people to all apps, services, and things always and in all places. One of the unique advantages of 6G is intended to handle a higher number of mobile connections than the 5G capacity, which is around 10 x 10^5 per km^2 [3,4]. Many proposals are aimed to cope with the massive energy consumption in the mobile network and cloud provider data centres to attain the high demand for a better Quality of Service (QoS) in the IoT environment [5-10]. Fog computing was proposed as an extension to the cloud for hosting applications as an extension to cloud computing. Fog computing provides enormous, scalable resources that allow computation and storage to be moved to the cloud from a physical location [11]. In addition, fog computing provides massive and scalable resources that enable the physical location of computation and storage to be moved to the cloud. It combines fog nodes and small cell networks which are installed in various areas by different mobile network operators (MNOs) to deliver various data services and applications to IoT devices and subscribers [12]. These IoT devices may select their

[*] *Corresponding author.*
E-mail address: syafizan@uitm.edu.my

https://doi.org/10.37934/araset.36.2.199212

MNOs as well as the associated Fogs Computing to improve their Quality-of-Experience (QoE) while consuming less energy.

IoT consists of a diverse set of devices with different task sizes that are linked to the Internet and collaborate to be responsible for the desired services by creating and sharing data. Today, most mobile and IoT apps use a server-client architecture with front-end smart devices and back-end cloud [13]. Figure 1 depicts fog computing as the focal point of energy efficiency architecture. Cloud computing and device computing are believed to be conventional computing. This is due to the slow transmission speed because cloud servers are typically deployed in remote areas of traditional core networks [14]. The large transmission distance and low transmission rate between IoT devices and cloud servers cause a long transmission delay [15,16]. The second reason is that it causes a burden to the cloud and devices as it will contribute to big data in cloud servers. As the number of IoT devices grows, many tasks are uploaded to cloud servers, resulting in high load and low computing efficiency for the cloud servers [17]. Furthermore, with conventional computing, the energy required to transfer tasks is substantial since IoT devices have limited computing capacity, and tasks are transferred to cloud servers via wireless links in traditional networks.

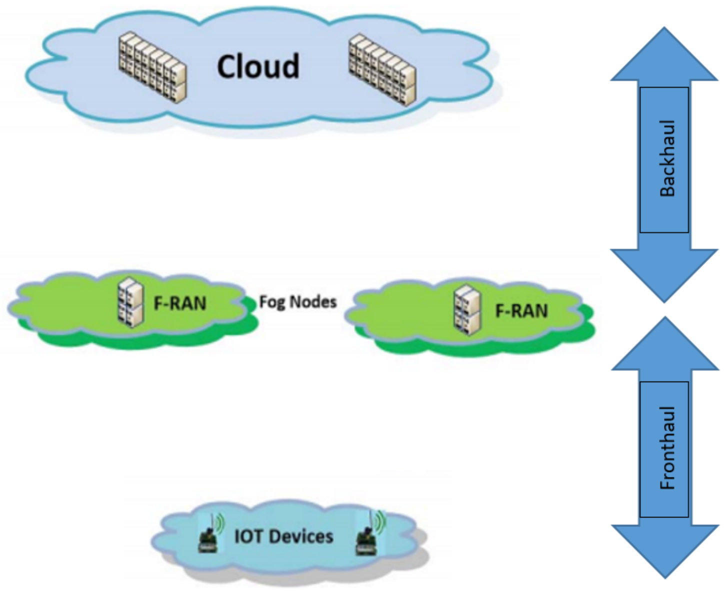

Fig. 1. Structure of fog computing

Fog computing, also known as edge computing, involves distributing computing resources and services closer to the edge of the network, such as at the devices, gateways, or local servers. This decentralized approach offers several benefits in various real-world use cases such as applications of smart cities, industrial Internet of things (IIoT), healthcare and autonomous vehicles. Fog computing plays a crucial role in smart city initiatives. Sensors, cameras, and other IoT devices deployed throughout the city generate massive amounts of data. By implementing fog computing, data processing, and analysis can be performed at the edge, reducing latency and bandwidth usage. For example, in a smart traffic management system, fog nodes placed at intersections can analyze real-time traffic data and control traffic signals locally, improving response times and reducing congestion.

Fog computing is particularly beneficial in industrial settings where low latency and real-time processing is critical. In manufacturing plants, fog nodes placed near production machinery can collect and process sensor data locally, enabling real-time monitoring, predictive maintenance, and immediate response to anomalies. This approach reduces the need for transmitting large volumes of raw data to the cloud, resulting in faster decision-making and lower bandwidth costs. Fog computing enhances healthcare services by bringing computation closer to patients and medical devices. For example, wearable health monitoring devices can collect vital signs and send them to nearby fog nodes for real-time analysis. These nodes can identify critical conditions and trigger immediate alerts, ensuring timely medical intervention. Fog computing also enables local storage and processing of patient data, maintaining privacy and security compliance while reducing reliance on a central cloud infrastructure.

Fog computing is crucial for autonomous vehicles, where split-second decision-making is vital for safety. Fog nodes deployed along roadways can process sensor data from vehicles, analyze traffic patterns, and provide real-time information to autonomous cars. This approach reduces the dependence on cloud connectivity, enabling faster response times and more efficient navigation.

These examples illustrate the wide-ranging applications of fog computing and how it brings computation closer to the edge, providing real-time processing, reduced latency, improved scalability, enhanced security, and cost savings. By leveraging the power of fog computing, organizations can unlock new possibilities and deliver more responsive and efficient services.

In recent years, task offloading for fog computing has received significant attention in research. Accordingly, the following section presents several techniques proposed in recent years, which could assist potential researchers in bridging the research gap. Chen *et al.,* [18] present the multi-user offloading problem in mobile-edge cloud computing and utilise game theory to accomplish effective offloading. Beraldi *et al.,* [19] explore a situation of two eNBs cooperating on computation offloading. The trade-off policy is used to reduce data exchange while minimizing system congestion in this situation, which is stated as an M/M/2-2K queuing model. This cooperative technique, however, cannot be used for the proposed task offloading problem, but it could be applied to the D2D communication environment. Furthermore, task offloading must be implemented using an exact mathematical model in the above two studies.

To reduce the processing time of users' apps, Miao *et al.,* [20] offer a prediction-based compute offloading and task relocation technique. However, in such a dynamic and random MEC environment, it is impossible to accurately forecast future values. Yan *et al.,* [21] investigate the problem of joint task offloading and resource allocation by taking both energy consumption and execution time into account. An effective bisection search policy is employed to compute optimal solutions in order to solve this challenge. It is, however, designed for offloading jobs with dependence linkages and cannot be utilised directly for independent tasks. Li *et al.,* [22] look into the multi-user offloading problem in WPMEC and present a Lyapunov-based online computation rate maximisation (OCRM) technique for achieving optimal bandwidth allocation and low data transmission energy usage. Approaches based on Lyapunov optimization are well adapted to solving the challenge of long-term performance optimization. Each objective function performance, however, must be a time-average metric, such as average energy or power, average delay, and average cost. As a result, it can't be used in optimization problems with deterministic constraints, such as the job deadline constraint. Besides that, energy-aware task offloading also has been discussed by Li *et al.,* in [23]. In their published articles, they proposed a novel energy- aware task offloading framework with MEC in IoT. Random pairing is done in the task offloading via ratio fog nodes and devices that follows 1:100 ratio. An architecture of task offloading also has been proposed according to their method.

In this work, the proposed Gale-Shapley stable marriage matching technique for task offloading differs from other approaches through its preference-based optimization, stable matching, proximity and resource considerations, and scalability, resulting in reduced transmission time delay, energy consumption, and improved QoS in IoT-fog computing environments. Firstly, the Gale-Shapley algorithm is based on preference rankings of both IoT devices and fog nodes. In this case, by considering factors such as the size of tasks and proximity, it optimizes the matching process to achieve desired outcomes. This preference-based approach differs from other methods that may use heuristics, random assignment, or static rules. Secondly, The Gale-Shapley algorithm ensures stable matches, where both IoT devices and fog nodes are mutually satisfied with their assignments. This stability contributes to consistent QoS by preventing frequent task reassignments or resource bottlenecks that could negatively impact performance. Following that, the matching process considers dual factors which are the proximity and resource availability of fog nodes to provide optimal QoS. This dual-factor approach sets it apart from other approaches that may only consider one of these factors, potentially leading to suboptimal QoS. By allocating tasks to nearby fog nodes (based on proximity) with sufficient resources (based on resource availability), the algorithm ensures that tasks experience lower latency, faster response times, and improved QoS. Finally, the Gale-Shapley algorithm is scalable and can handle a large number of IoT devices and fog nodes. This scalability differentiates it from approaches that may struggle to efficiently manage a growing number of devices and nodes.

2. Methodology

2.1 Simulation Setting

Figure 2 shows the architecture of task-offloading via the Internet of Things (IoT) Network. With the aid of fog servers (edge servers), IoT devices downloaded their computing tasks to the fog server to decrease the burden of the IoT devices.

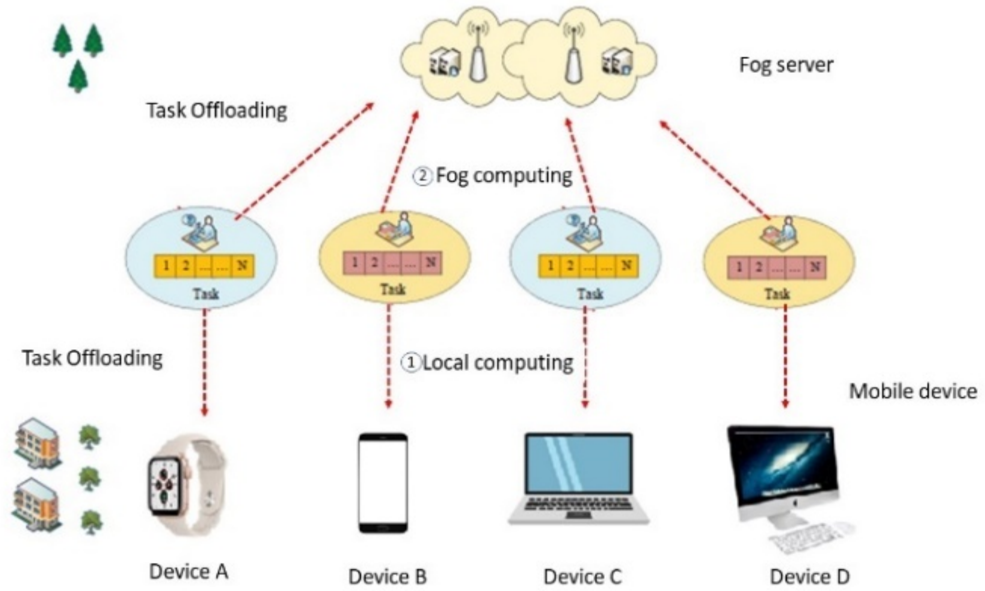

Fig. 2. Architecture of task offloading in fog computing

The following is a summary of each component's roles in the task-offloading approach:

i. IoT Devices: Several IoT devices (e.g., smartphones) surround the edge servers, with each IoT device (i=A, B, C, D…n^{th}). Each IoT device has a limited computing capability, as shown in the diagram (e.g., CPU cycles, battery).
ii. Edge Server (Fog Server): The Fog servers are positioned at the network's edge to schedule task offloading for IoT devices, $(\mathrm{i} = 0,1,2 \ldots\ \mathrm{m}1)$with each edge server having restricted

processing capability, represented by i. When an IoT device offloads tasks, it connects with the fog server over a wireless connection.

iii. Cloud Server: A cloud server may store material supplied by various mobile users and supply computational power to IoT devices. Cloud computing gives internet access to computer services, and the cloud server is hosted in faraway clouds. Wireless connections connect each IoT device and fog server to the cloud server.

iv. Computing Task Delay or Latency: IoT devices produce many different sorts of tasks. In this simulation, the devices are arranged in ascending order according to the number of tasks.

v. Wireless Channel: Wireless channels are used by IoT devices to interact with edge servers. k = (1,..., k,... K) denotes the set of orthogonal channels. Through the use of wireless channels, tasks are offloaded to edge servers. The performance of task offloading is proportional to the wireless channel's transmission bandwidth.

In this simulation, task m conserves energy consumption, so the unit time cost to complete the task m is given by μ_o. Moreover, based on the task division, the m is divided into several devices (Device A,B,C,D). Task m can be completed with low energy by matching the devices to the fog nodes (device-fog-offloading). The matchings are done via Gale Shapley matching. Table 1 displayed all the parameters and the values for simulation purposes.

Table 1
Simulation parameters

Parameter	Definition	Value
S_n	Size of tasks in IoT devices	Device A = 1.1 kbytes
		Device B = 1.2 kbytes
		Device C = 1.3 kbytes
		Device D = 1.4 bytes
μ_2	Transmission bandwidth in fog	4 GHz
μ_1	Transmission bandwidth in 6G IoT devices	0.8 GHz
$w_{n,}\ w_j$	Link bandwidth in fog and devices	40 MHz
μ_o	Unit time cost to complete the task m	1000 mW
P_2	The transmission power of fog	100 mW
P_1	The transmission power of devices	100 mW
α_m^t	Offloading time	α_m^t = 0, without offloading
		α_m^t = 1, with offloading

2.2 Flowchart

Figure 3 illustrates the flowchart of the proposed algorithm. This illustrated design is classified into eight phases. The first phase is the specification of tasks in IoT devices where each device has tasks of different sizes. The tasks are then displayed in ascending order based on their priority. In this project, four devices of varying sizes were applied. The following phase derived the energy consumption in fog nodes according to the size of the tasks. The third phase executes a matching game between fog nodes and IoT devices with Gale Shapley Stable Marriage Matching Algorithm. High-priority tasks of mobile devices are matched to the fog nodes that provide high energy and acceptable latency. Phase 4 involved the Matching condition where if the IoT devices suit the energy in fog nodes, the devices are considered engaged. However, if it does not engage, the process will be repeated until the IoT devices found their perfect stable matches, hence, the device and fog nodes need to go through the process starting from Phase 1 again. Following that, the tasks from the IoT devices are uploaded to the paired fog nodes (Task-offloading energy efficiency technique is applied).

Once the matching game is done, the tasks from the IoT devices are uploaded to the fog nodes, and the fog nodes will be the helper for the IoT devices in computing the given tasks. After the tasks offloading process, performance evaluation is carried out in terms of energy consumption and latency, and performance measurement is performed by displaying the results of energy consumption and latency to prove the effectiveness of the proposed task-offloading approach.

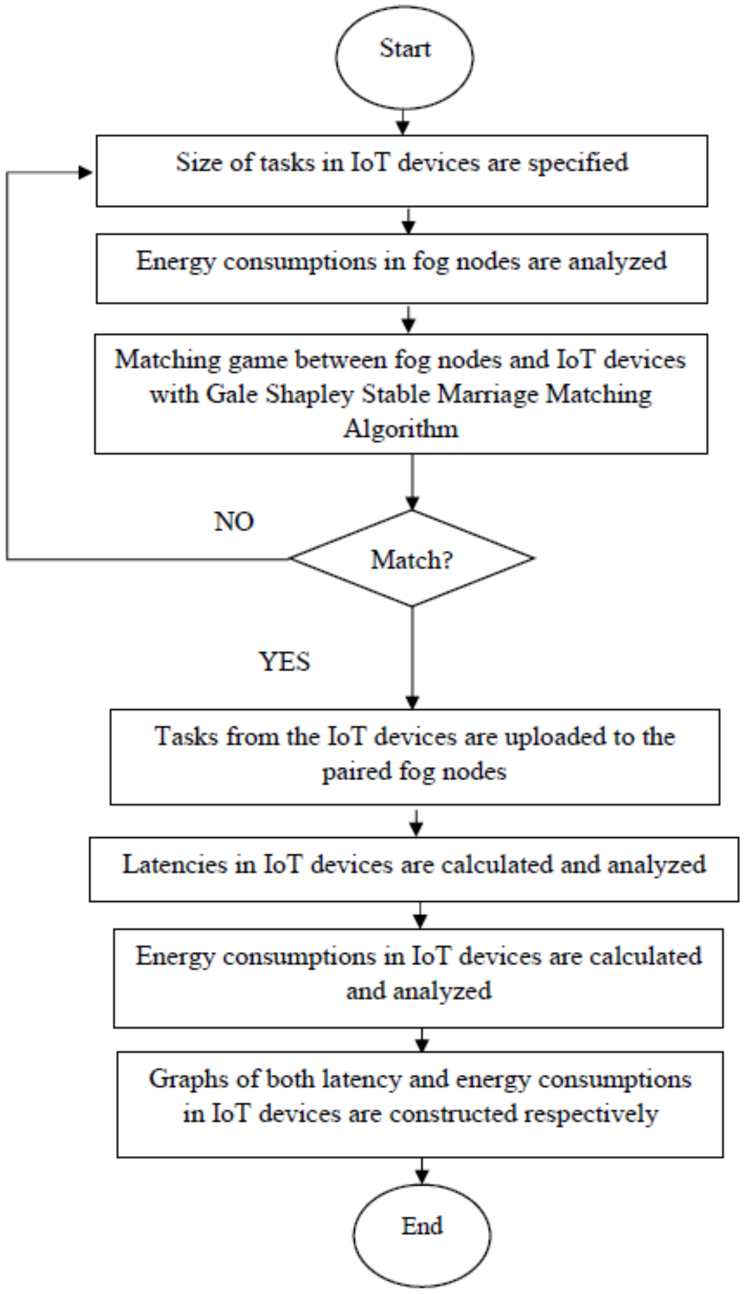
Start
Size of tasks in IoT devices are specified
Energy consumptions in fog nodes are analyzed
Matching game between fog nodes and IoT devices with Gale Shapley Stable Marriage Matching Algorithm
NO
Match?
YES
Tasks from the IoT devices are uploaded to the paired fog nodes
Latencies in IoT devices are calculated and analyzed
Energy consumptions in IoT devices are calculated and analyzed
Graphs of both latency and energy consumptions in IoT devices are constructed respectively
End

Fig. 3. Flowchart of the proposed algorithm

2.3 Details of the Offloading Strategy

The purpose of this research is to construct an ideal task offloading strategy to minimize the consumption of energy. During its Time-to-Live (TTL), the goal is to reduce energy usage and increase the number of tasks accomplished within the time.

2.3.1 Latency in IoT devices and fog nodes analysis

The completion time of task m that executes at the IoT device can be calculated by

$$b_m^1 = \sum_{n=1}^{N}(\frac{s_n}{\omega_n} + \frac{2S_n}{\mu_1}) \tag{1}$$

In addition, the completion task m that is executed at the fog server can be calculated by

$$b_m^2 = \sum_{n=1}^{N}(\frac{s_n}{\omega_j} + \frac{2S_n}{\mu_2}) \tag{2}$$

2.3.2 Energy consumption in IoT devices and fog nodes analysis

The energy consumption in IoT devices with m tasks is expressed by

$$c_m^1 = \sum_{n=1}^{N}(u_o \times \frac{S_n}{\omega_n} + 2s_n \times p_1) \tag{3}$$

In addition, the cost of performing task m at the fog server as

$$c_m^2 = \sum_{n=1}^{N}(u_o \times \frac{S_n}{\omega_j} + 2s_n \times p_2) \tag{4}$$

As a result,

$$E[a_m^t] = a_m^t \times c_m^1 + (1 - a_m^t) \times c_m^2 \tag{5}$$

can be used to calculate the estimated energy consumption.

2.3.3 Device-to-Fog Matching strategy with Gale Shapley

This algorithm has a runtime complexity of $O(n^2)$, where n is the number of fog devices. The runtime is linear in the input size since the input preference lists have a size proportionate to n^2. These iteration orders follow the process of Gale Shapley matching. In this process, the IoT devices are arranged in ascending order according to the task size. Fog nodes with high energy consumption will prefer the IoT devices with high latency and reject the IoT devices with low latency to ensure stable matching. Once the elimination is done, all the fog nodes and the IoT devices are matched according to their preferences and none of them is left unpaired. In this project, the fog and device are defined as a set I and J respectively. Set I is labelled as fog I=$\{f_0, f_{1,} f_{2.} f_3\}$ and set J is labelled as group device J= $\{A, B, C, D\}$. The pseudo-code for the stable matching is as follows:

```
//f=fog
//d=device
Initialize all fog and device to be free
While there exists a device that does not have a partner
{
    f= d's most preferred fog to whom it has not proposed yet
    if f is does not have a partner
        (f.d) become engaged
else some pair (d’, f) already exists
   if d prefers f to f’
        (f, d) become engaged
        d’ become free
   else
        (d’, f) remain engaged
}
```

2.3.4 Task offloading strategy

After the matching game is executed, the task will be uploaded to the fog server. This is to lessen the burden of the device to hold the task. The task will be offloaded at $a_m^t = 1$ and the energy consumption in that process can be derived based on Eq. (3). Meanwhile, the energy consumption without task offloading is denoted as $a_m^t = 0$.

2.4 Limitation and Challenges

Simulating Gale-Shapley's stable marriage matching technique within the context of IoT devices and fog nodes introduces additional limitations and challenges such as resource constraints, dynamic environment, heterogeneous preferences and constraints, scalability and network size, real-time constraints and latency. IoT devices and fog nodes typically have limited computational power, memory, and energy resources. Simulating the matching process within these constraints can be challenging, as the algorithm's execution and resource requirements need to be optimized to ensure feasibility on resource-constrained devices. IoT devices and fog nodes operate in dynamic and unpredictable environments. The availability and resource status of fog nodes may change dynamically due to device mobility, network conditions, or varying workloads. Simulating the matching algorithm in such dynamic environments and capturing the real-time changes can be challenging, requiring dynamic updates and adjustments to the matching process.

IoT devices and fog nodes may have diverse preferences, capabilities, and constraints. Simulating and representing these heterogeneous preferences and constraints accurately can be complex. Assigning appropriate preference rankings and modelling constraints based on the characteristics of the IoT devices and fog nodes is essential for realistic simulations. The number of IoT devices and fog nodes in a network can be vast, leading to scalability challenges. The execution time and computational requirements of the matching algorithm may increase significantly as the network size grows. Efficient simulation techniques, such as parallelization or approximation algorithms, may be necessary to handle large-scale IoT deployments. Some IoT applications require real-time decision-making and low-latency interactions. Simulating the matching process in real time and analyzing the impact of latency on the algorithm's performance becomes crucial. Accounting for real-time constraints and ensuring timely and efficient matching decisions can be challenging in IoT and fog computing environments.

Addressing these limitations and challenges within the simulation of Gale-Shapley's stable marriage matching technique in IoT devices and fog nodes requires careful consideration of resource constraints, communication overhead, dynamic environments, heterogeneous preferences, scalability, privacy, security, and real-time constraints. It necessitates specialized simulation frameworks and techniques that account for the unique characteristics and constraints of IoT and fog computing environments.

2.5 Summary of Methodology

The utilization of the Gale-Shapley Stable Marriage Matching Game for matching IoT devices with fog nodes is proposed that can contribute to improving end-to-end performance in terms of energy efficiency and transmission time. The flow of the approach starts with preference-based matching where each IoT device and fog node rank their preferences for potential matches based on specific criteria, in this work, it is based on the size of each task and proximity within the fog nodes and IoT devices. The following procedure is an iterative matching process. The Gale-Shapley algorithm proceeds iteratively, starting with each IoT device proposing its most preferred fog node. Fog nodes then evaluate their proposals and accept or reject them based on their own preferences and capacity constraints. Rejections prompt IoT devices to propose to their next preferred fog node, and this process continues until a stable matching is reached.

This proposed matching game contributes to improving end-to-end performance such as energy efficiency and latency. By employing the Gale-Shapley algorithm, IoT devices can be efficiently matched with fog nodes that best meet their requirements. This reduces unnecessary energy consumption and communication overhead that would occur if devices were randomly assigned or inefficiently matched. Optimized device-node assignments help minimize energy expenditure, leading to improved overall energy efficiency in the IoT-fog computing environment. Matching IoT devices with nearby fog nodes can significantly reduce transmission distance and latency. By considering proximity as a preference criterion, the algorithm can allocate devices to nearby fog nodes, enabling faster data transmission and reduced communication delays. This reduction in transmission time contributes to lower latency, improved responsiveness, and enhanced real-time processing capabilities. By leveraging the Gale-Shapley Stable Marriage Matching Game, the process of matching IoT devices with fog nodes can be optimized, resulting in improved energy efficiency, reduced transmission time, better resource utilization, and enhanced system scalability. These benefits contribute to the overall performance and effectiveness of IoT-fog computing environments.

3. Results

3.1 Performances of Energy Consumption and Latency

This section presents the performance analysis of energy consumption and latency. The transmission loss is considered negligence. During the offloading process, stable matching is required for the IoT devices and fog nodes. The stable matching is executed based on the IoT device and fog nodes' energy value. Based on the stable matching, the IoT device with high latency will be paired with the fog with a high level of energy. The matching is considered stable due to the ability of the fog nodes' energy to compute the tasks. Firstly, the latency of all the IoT devices is derived with Eq. (1). Table 2 shows the latency of the IoT device before offloading process.

Table 2
Latency before implementation of task offloading

IoT devices	Latency (in µs)
A	30.25
B	63.25
C	99.50
D	137.5

Secondly, the energy consumption in fog nodes before task offloading is derived and tabulated in Table 3.

Table 3
Energy consumption before implementation of task offloading

Fog	Tasks in kbytes	Energy in kJ
0	1.1	1.1
1	1.2	2.3
2	1.3	3.6
3	1.4	5.0

Thirdly, the fog nodes and the IoT devices will be paired according to the Gale Shapley Stable Marriage technique. The details of pairing are displayed in Table 4.

Table 4
Pairing order based on the device latency and fog nodes' energy based on Gale Shapley matching

Fog	Device	Pair order
0	A	1st
1	B	2nd
2	C	3rd
3	D	4th

In this pairing and task offloading process, the tasks in the IoT devices and fog are shared based on a 3:2 ratio, which causes the reduction of tasks in IoT devices by 40%. Table 5 shows the new latency of the tasks in IoT devices after offloading process is carried out.

Table 5
Latency after implementation of task offloading

IoT devices	Latency (in µs)
A	18.15
B	37.95
C	59.40
D	82.50

Next, the energy in devices without task offloading is also measured and summarized in Table 6.

Table 6
The energy consumption in devices without task offloading

IoT Devices	Energy consumption (J)
A	220
B	460
C	700
D	1060

After the Stable Matching game of the fog nodes with the devices, the energy consumption in the IoT is reduced because some of the tasks are transferred to the fog nodes. The total energy consumption in IoT devices after stable matching and task offloading is displayed in Table 7.

Table 7
Energy consumption before implementation of task offloading

IoT Devices	Energy consumption (J)
A	132
B	276
C	432
D	600

Based on the results gained, it can be concluded that the Gale Shapley Stable Marriage Matching game that is implemented in task offloading has contributed to low latency and energy consumption in IoT devices. In addition, based on the ratio division, the latency and energy consumption in IoT devices are reduced up to 37.5% and 36.15% respectively after the task offloading method. This is due to the task being transferred to the fog nodes from the devices. Before the task offloading approach is implemented, the IoT devices only go through local computing, which is also known as the conventional method of computing. As a result, the IoT devices will have the burden to process all the data and it is simply inefficient because it will contribute to high energy consumption and latency in the task computing; thus, it gives a low QoS in the IoT devices.

With the implementation of fog computing, much of the processing in fog computing happens in a data hub on a smart mobile device or at the network's fog in a smart router or other gateway devices. Hence, some of the data and tasks are transferred to fog nodes and this will reduce the burden of the IoT devices to compute the tasks and this process will contribute to minimising energy consumption. In order to evaluate and verify the results, the graph of latency and energy consumption are displayed in Figure 4 and 5 respectively. The red line legends show the energy and latency in IoT devices before the offloading process and the blue line legends show the vice versa.

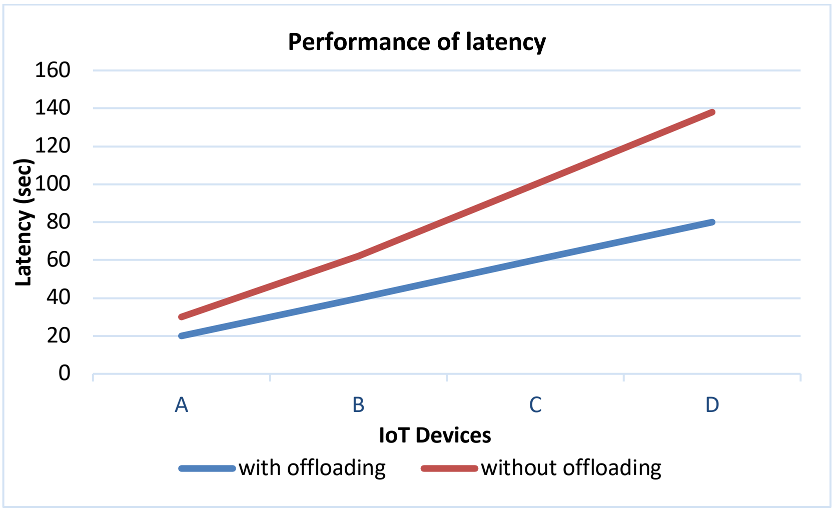

Fig. 4. Performance evaluation of latency

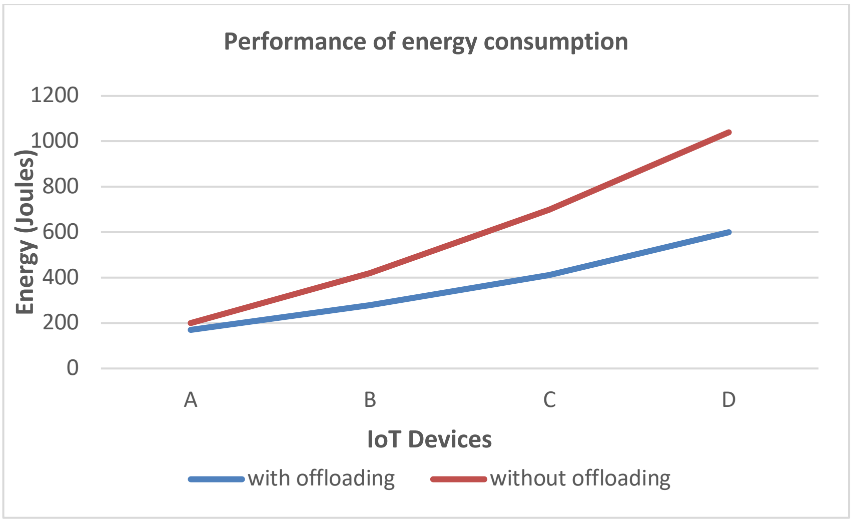

Fig. 5. Performance evaluation of energy consumption

4. Conclusions

By using the Gale-Shapley stable marriage matching technique, the proposed approach optimizes the allocation of tasks to fog nodes based on proximity and resource availability. This reduces latency since tasks are processed closer to the devices, resulting in faster response times. Moreover, this technique also helps in minimizing energy consumption. By offloading tasks to nearby fog nodes instead of transmitting data to distant cloud servers, the energy required for data transmission is reduced. This is particularly beneficial for IoT devices with limited energy resources, as it helps extend their battery life. Overall, the Gale-Shapley stable marriage matching technique applied to task offloading in fog computing within IoT devices and fog nodes optimizes resource allocation, reduces latency, and minimizes energy consumption, resulting in improved performance and efficiency in IoT deployments. Based on simulation results, The proposed approach reduces energy usage and delay by 36.15% and 37.5%, respectively, according to simulation results.

For future works, more comprehensive and rigorous evaluations of the proposed Gale-Shapley Stable Marriage Matching task offloading technique will be performed. The proposed algorithm will be further enhanced to evaluate the performance of the technique in terms of energy consumption and latency with different percentages of partial offloading based on task sizes which is a crucial step in understanding its effectiveness and efficiency. This type of analysis will provide valuable insights into the behaviour of the technique under varying conditions and workload characteristics. We are planning such evaluations as part of our future work.

Acknowledgement
This research was supported under College of Engineering, UiTM Shah Alam and the Fundamental Research Grant Scheme Malaysia Higher Education: FRGS/1/2019/ICT03/UITM/03/1.

References
[1] Malik, Usman Mahmood, Muhammad Awais Javed, Sherali Zeadally, and Saif ul Islam. "Energy-efficient fog computing for 6G-enabled massive IoT: Recent trends and future opportunities." *IEEE Internet of Things Journal* 9, no. 16 (2021): 14572-14594. https://doi.org/10.1109/JIOT.2021.3068056
[2] Ajibola, Opeyemi O., Taisir EH El-Gorashi, and Jaafar MH Elmirghani. "Disaggregation for energy efficient fog in future 6G networks." *IEEE Transactions on Green Communications and Networking* 6, no. 3 (2022): 1697-1722. https://doi.org/10.1109/TGCN.2022.3160397

[3] Keshari, Niharika, Dinesh Singh, and Ashish Kumar Maurya. "A survey on Vehicular Fog Computing: Current state-of-the-art and future directions." *Vehicular Communications* 38 (2022): 100512. https://doi.org/10.1016/j.vehcom.2022.100512

[4] Cheng, Zhipeng, Minghui Liwang, Ning Chen, Lianfen Huang, Xiaojiang Du, and Mohsen Guizani. "Deep reinforcement learning-based joint task and energy offloading in UAV-aided 6G intelligent edge networks." *Computer Communications* 192 (2022): 234-244. https://doi.org/10.1016/j.comcom.2022.06.017

[5] Swain, Chittaranjan, Manmath Narayan Sahoo, Anurag Satpathy, Khan Muhammad, Sambit Bakshi, Joel JPC Rodrigues, and Victor Hugo C. de Albuquerque. "METO: Matching-theory-based efficient task offloading in IoT-fog interconnection networks." *IEEE Internet of Things Journal* 8, no. 16 (2020): 12705-12715. https://doi.org/10.1109/JIOT.2020.3025631

[6] Cao, Kun, Junlong Zhou, Guo Xu, Tongquan Wei, and Shiyan Hu. "Exploring renewable-adaptive computation offloading for hierarchical QoS optimization in fog computing." *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 39, no. 10 (2019): 2095-2108. https://doi.org/10.1109/TCAD.2019.2957374

[7] Tran-Dang, Hoa, and Dong-Seong Kim. "A many-to-one matching based task offloading (mato) scheme for fog computing-enabled iot systems." In *2022 International Conference on Advanced Technologies for Communications (ATC)*, pp. 239-244. IEEE, 2022. https://doi.org/10.1109/ATC55345.2022.9942992

[8] Yousefpour, Ashkan, Genya Ishigaki, Riti Gour, and Jason P. Jue. "On reducing IoT service delay via fog offloading." *IEEE Internet of things Journal* 5, no. 2 (2018): 998-1010. https://doi.org/10.1109/JIOT.2017.2788802

[9] Tran-Dang, Hoa, and Dong-Seong Kim. "Dynamic task offloading approach for task delay reduction in the IoT-enabled fog computing systems." In *2022 IEEE 20th International Conference on Industrial Informatics (INDIN)*, pp. 61-66. IEEE, 2022. https://doi.org/10.1109/INDIN51773.2022.9976147

[10] Huang, Xiaoge, Yifan Cui, Qianbin Chen, and Jie Zhang. "Joint task offloading and QoS-aware resource allocation in fog-enabled Internet-of-Things networks." *IEEE Internet of Things Journal* 7, no. 8 (2020): 7194-7206. https://doi.org/10.1109/JIOT.2020.2982670

[11] Pakhrudin, Nor Syazwani Mohd, Murizah Kassim, and Azlina Idris. "A review on orchestration distributed systems for IoT smart services in fog computing." *International Journal of Electrical and Computer Engineering* 11, no. 2 (2021): 1812. https://doi.org/10.11591/ijece.v11i2.pp1812-1822

[12] Singh, Jagdeep, Parminder Singh, and Sukhpal Singh Gill. "Fog computing: A taxonomy, systematic review, current trends and research challenges." *Journal of Parallel and Distributed Computing* 157 (2021): 56-85. https://doi.org/10.1016/j.jpdc.2021.06.005

[13] Rho, Seungmin, and Yu Chen. "Social Internet of Things: Applications, architectures and protocols." *Future Generation Computer Systems* 82 (2018): 667-668. https://doi.org/10.1016/j.future.2018.01.035

[14] Rajkumar, K., and U. Hariharan. "Moving to the cloud, fog, and edge computing paradigms: Convergences and future research direction." In *Artificial Intelligence and Machine Learning for EDGE Computing*, pp. 425-442. Academic Press, 2022. https://doi.org/10.1016/B978-0-12-824054-0.00018-6

[15] Sopin, Eduard S., Anastasia V. Daraseliya, and Luís M. Correia. "Performance analysis of the offloading scheme in a fog computing system." In *2018 10th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, pp. 1-5. IEEE, 2018. https://doi.org/10.1109/ICUMT.2018.8631245

[16] Iftikhar, Sundas, Sukhpal Singh Gill, Chenghao Song, Minxian Xu, Mohammad Sadegh Aslanpour, Adel N. Toosi, Junhui Du *et al.,* "AI-based fog and edge computing: A systematic review, taxonomy and future directions." *Internet of Things* (2022): 100674. https://doi.org/10.1016/j.iot.2022.100674

[17] Douch, Salmane, Mohamed Riduan Abid, Khalid Zine-Dine, Driss Bouzidi, and Driss Benhaddou. "Edge computing technology enablers: A systematic lecture study." *IEEE Access* 10 (2022): 69264-69302. https://doi.org/10.1109/ACCESS.2022.3183634

[18] Jameel, Furqan, Zara Hamid, Farhana Jabeen, Sherali Zeadally, and Muhammad Awais Javed. "A survey of device-to-device communications: Research issues and challenges." *IEEE Communications Surveys & Tutorials* 20, no. 3 (2018): 2133-2168. https://doi.org/10.1109/COMST.2018.2828120

[19] Benhong, Zhang, Han Liang, and Bi Xiang. "Task Offloading and Resource Allocation Scheme based on Cooperative Game." In *2022 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA)*, pp. 327-333. IEEE, 2022. https://doi.org/10.1109/ICAICA54878.2022.9844531

[20] Miao, Yiming, Gaoxiang Wu, Miao Li, Ahmed Ghoneim, Mabrook Al-Rakhami, and M. Shamim Hossain. "Intelligent task prediction and computation offloading based on mobile-edge cloud computing." *Future Generation Computer Systems* 102 (2020): 925-931. https://doi.org/10.1016/j.future.2019.09.035

[21] Yan, Jia, Suzhi Bi, Ying Jun Zhang, and Meixia Tao. "Optimal task offloading and resource allocation in mobile-edge computing with inter-user task dependency." *IEEE Transactions on Wireless Communications* 19, no. 1 (2019): 235-250. https://doi.org/10.1109/TWC.2019.2943563

[22] Li, Chunlin, Jianhang Tang, and Youlong Luo. "Dynamic multi-user computation offloading for wireless powered mobile edge computing." *Journal of Network and Computer Applications* 131 (2019): 1-15. https://doi.org/10.1016/j.jnca.2019.01.020

[23] Li, Jiliang, Minghui Dai, and Zhou Su. "Energy-aware task offloading in the Internet of Things." *IEEE Wireless Communications* 27, no. 5 (2020): 112-117. https://doi.org/10.1109/MWC.001.1900495