[1]Alaa Sabeeh Salim,

[2]Mohamad Mahdi Kassir

[3]Amir Lakizadeh

Managing and Decreasing Power Consumption of Devices in a Smart City Environment

Abstract: - As smart cities continue to grow and the number of connected devices increases, power consumption becomes a critical concern. By offloading computationally intensive tasks from resource-constrained devices to more powerful edge servers, energy efficiency can be significantly improved. The research proposes a framework for managing power consumption in smart cities by offloading computational tasks to edge servers. This approach, considering factors like device capabilities, network conditions, and energy profiles, can improve energy efficiency. The framework's effectiveness is evaluated through real-world data simulations and performance metrics. Results show that offloading tasks to edge servers significantly reduces power consumption, conserving energy and prolonging battery life. The framework's adaptability ensures optimal resource allocation, maximizing energy efficiency without compromising performance. This research offers practical solutions for sustainable and energy-efficient operations in smarter cities.

Keywords: Smart city, Power consumption, Energy management, Edge commuting, Computing offloading

Introduction

1.1 Background

The coming era of computers aims to change the way people interact with technology. The technological revolution of recent decades has revolutionized the way people communicate, work, travel, live, etc., driven by advances and developments in Information and Communication Technologies (ICT). Cities are becoming smart, dynamic infrastructures serving citizens, meeting energy efficiency and sustainability criteria [1]. In these densely populated cities and new urban areas, societies are facing new challenges to minimize the consumption of natural energy resources, promote renewable energy, and reduce atmospheric CO2 emissions. The concept of the Smart City [2] provides an effective means of managing infrastructure and services efficiently and in line with the needs of the city and its inhabitants.

Smart cities face numerous challenges in terms of energy consumption. The deployment of a large number of IoT devices, sensors, and infrastructure components leads to increased power demand. Moreover, the diverse range of applications, such as smart transportation, energy management, and urban planning, further exacerbates the energy challenges. Researchers have emphasized the need for energy-efficient solutions to ensure the sustainable development of smart cities. IoT devices play a pivotal role in smart city environments, but their energy consumption is a significant concern. These devices often operate on battery power and may have limited energy resources. Therefore, reducing the energy consumption of IoT devices is essential to prolong their battery life, minimize maintenance efforts, and enhance the overall sustainability of the smart city ecosystem [3].

Energy consumption is a critical metric that quantifies the amount of energy utilized by a mobile device during the execution of tasks. It serves as a key indicator for evaluating the device's power efficiency and has significant implications for sustainability, battery life, and environmental impact. By measuring and analyzing energy consumption, we gain insights into how effectively the device utilizes its power resources. A device with lower energy consumption is considered more power-efficient, as it accomplishes tasks while minimizing unnecessary energy usage. This efficiency translates into extended battery life, allowing users to use their devices for longer durations without needing frequent recharges [4]. Minimizing energy consumption is not only advantageous for users but also contributes to environmental conservation. As mobile devices have become integral to our daily lives, the collective energy consumption of these devices has increased significantly. By optimizing energy usage,

[1] Department of Computer Engineering, University of Qom, Qom, Iran.

E-mail: Alaasabeeh@gmail.com, m.kaseer@alumni.iut.ac.ir, lakizadeh@qom.ac.ir

Copyright © JES 2024 on-line : journal.esrgroups.org

we can reduce the overall environmental impact associated with mobile device manufacturing, charging, and disposal.

Efficient energy consumption also aligns with the growing trend of sustainable technology. Consumers and organizations are increasingly prioritizing eco-friendly practices, and mobile devices that consume less energy are viewed positively from an environmental standpoint. Furthermore, managing energy consumption is crucial in resource-constrained scenarios, such as remote areas or situations where access to power sources is limited. By designing mobile devices with optimized energy consumption, we can ensure their usability and functionality in diverse environments. Measuring energy consumption provides us with essential information about a device's power efficiency, sustainability, and environmental impact. Minimizing energy consumption not only extends battery life and enhances user experience but also contributes to overall environmental conservation and supports the adoption of sustainable technology practices [5].

Efficient and dynamic computation offloading on the network edge has emerged as a promising solution to manage energy consumption in smart city environments. By offloading computationally intensive tasks to edge servers or cloud infrastructure, the energy burden on individual IoT devices can be reduced, leading to improved energy efficiency. Researchers have explored various offloading techniques and strategies to optimize power consumption while maintaining acceptable performance levels. The literature highlights the significance of energy-aware decision-making algorithms for computation offloading in smart cities [6]. These algorithms consider energy consumption as a critical factor in determining whether a task should be offloaded or processed locally. Researchers have proposed optimization models, machine learning algorithms, and heuristic approaches to make energy-aware offloading decisions, thereby minimizing energy consumption in smart city environments.

1.2 Research Objectives

The research seeks to:

- Analyze the energy consumption patterns in smart city environments.
- Understand the specific challenges related to energy consumption in smart cities such as the deployment of numerous IoT devices, diverse applications, and infrastructure components.
- Explore efficient and dynamic computation offloading techniques as a means to reduce power consumption in smart cities.
- Examine existing offloading strategies and evaluating their effectiveness in terms of energy savings and performance trade-offs.

Literature review

Various studies have focused on exploring innovative approaches to address the energy challenges associated with the growing deployment of IoT devices in urban environments. In this literature review, we will examine the existing research related to efficient and dynamic computation offloading on the network edge for smart city environments, which aligns with the objectives of the current study. One notable study in this field is the work by Chen et al. [7], where they proposed an energy-efficient computation offloading framework for IoT devices in smart cities. They developed an optimization algorithm that dynamically decides whether to offload a task to the network edge or process it locally based on factors like energy consumption, task size, and available resources. Their results demonstrated significant energy savings while maintaining acceptable latency levels.

Another relevant research contribution is the work by Zhang et al. [8], which focused on the energy optimization of IoT devices through dynamic computation offloading in smart cities. They proposed an energy-aware offloading scheme that considers the variations in energy prices and network conditions. Their approach intelligently determines the optimal offloading decision by considering both energy consumption and monetary costs, resulting in substantial power savings. Furthermore, the study by Li et al. [9] investigated the trade-offs between cloud offloading and edge offloading in a smart city context. They developed a hybrid offloading strategy that dynamically selects between cloud and edge resources based on factors such as task characteristics, network

conditions, and energy consumption. Their evaluation showed that dynamic offloading decisions can effectively reduce power consumption and improve response time for IoT applications.

2.1 Edge Computing and Power Efficiency

Edge computing is a paradigm that offers cloud computing capabilities at the network edge, in close proximity to end devices. This approach reduces latency, improves energy efficiency, and offers flexible computing options for computation-intensive tasks. By offloading tasks to nearby edge nodes, devices can reduce reliance on backhaul and cloud network bandwidth, improving efficiency and reducing network congestion. Edge computing is particularly beneficial for time-critical applications like IoT, video streaming, autonomous vehicles, and augmented reality. It also offers energy efficiency by reducing the need for extensive local processing, which can consume significant amounts of energy. Edge nodes, also known as fog nodes, mobile-edge servers, or cloudlets, provide storage and computational resources with minimal delay, enabling efficient execution of tasks requiring substantial computing power or large amounts of data processing [10].

Edge computing refers to the practice of processing and analyzing data closer to the source or at the network edge, rather than sending all the data to a central cloud or data center. This approach offers several benefits, including improved latency, bandwidth optimization, enhanced data privacy, and increased reliability. In the context of power efficiency, edge computing can play a significant role in reducing energy consumption. One of the key advantages of edge computing is the ability to process data locally, closer to where it is generated. By processing and analyzing data at the edge, only relevant and summarized information needs to be transmitted to the central cloud or data center. This reduces the volume of data that needs to be transmitted over the network, resulting in lower energy consumption associated with data transmission [11]. Edge computing reduces network latency by processing data closer to the source. This is particularly beneficial for time-sensitive applications, such as real-time monitoring and control systems in smart cities. By minimizing the time it takes for data to travel back and forth between devices and the cloud, edge computing reduces the need for constant network communication, leading to lower energy consumption.

Edge computing distributes computing resources across various edge devices, such as edge servers, gateways, and IoT devices. This distribution allows for localized processing and reduces the reliance on centralized data centers. By leveraging the computing capabilities of edge devices, the overall energy consumption can be optimized, as the workload is distributed and processed efficiently across the network. Efficient task offloading and resource allocation strategies can further enhance power efficiency in edge computing environments. By intelligently offloading computationally intensive tasks from resource-constrained devices to more powerful edge servers, energy consumption can be optimized. Dynamic resource allocation techniques ensure that resources are allocated based on workload demand, minimizing idle resource consumption and maximizing energy efficiency [12].

Edge computing facilitates real-time decision-making by processing data locally and providing immediate responses. In smart city applications, such as traffic management or emergency response systems, quick decision-making is crucial. By avoiding the latency associated with sending data to a remote cloud, edge computing reduces response times, leading to more efficient operations and optimized energy usage. In traditional cloud-based architectures, data from IoT devices or sensors is typically sent to a central data center for processing and analysis [13]. This constant transmission of data over the network can result in significant energy consumption. Edge computing minimizes network overhead by performing computations locally, thereby reducing the need for continuous data transmission. This reduction in network traffic leads to energy savings and improved overall network efficiency.

Edge computing utilizes distributed computing resources across various edge devices, allowing for efficient resource utilization. Instead of relying solely on centralized data centers, edge devices contribute their computing capabilities to process data locally. This distributed approach ensures that computing resources are used more efficiently and reduces the energy consumption associated with idle resources. Additionally, edge devices can dynamically allocate resources based on workload demands, further optimizing energy usage [14]. Edge computing enables finer-grained power management at the device level. Edge devices can employ power-saving techniques, such as low-power sleep modes or dynamic voltage scaling, to optimize energy consumption based on workload requirements. By tailoring power management strategies to the specific needs of edge devices, power

efficiency can be significantly improved, resulting in extended battery life for battery-powered devices and reduced overall energy consumption.

Edge computing provides scalability and flexibility in managing computational resources. As the number of IoT devices and data sources in smart city environments increases, edge computing can dynamically scale resources to handle the growing workload. By distributing computing tasks across edge devices, the system can adapt to changing demands, ensuring efficient resource utilization and minimizing unnecessary energy expenditure. Energy-aware decision-making algorithms and policies can be designed specifically for edge computing environments. These algorithms consider energy consumption as a crucial factor when making decisions related to task offloading, resource allocation, and workload management. By prioritizing energy efficiency in decision-making processes, edge computing systems can effectively reduce power consumption and maximize the utilization of available energy resources [15].

2.3 Dynamic resource allocation

Dynamic resource allocation in smart cities plays a vital role in optimizing power efficiency. By balancing workloads, making energy-aware decisions, enabling adaptive scaling, facilitating task offloading, leveraging predictive analytics, and integrating with energy management systems, dynamic resource allocation contributes to efficient utilization of computing resources and minimizes energy consumption in smart city environments [16]. Dynamic resource allocation plays a crucial role in optimizing power efficiency in smart cities. It involves intelligently allocating computing resources, such as processing power, memory, and storage, based on the real-time demands of applications and services. Dynamic resource allocation ensures that workloads are evenly distributed across available computing resources. By analyzing the workload patterns and resource utilization, resources can be allocated dynamically to prevent the overloading of certain devices or servers. This balancing of workloads helps avoid resource bottlenecks and reduces the need for additional resources, leading to optimized energy consumption.

Dynamic resource allocation algorithms take into account energy consumption as a critical factor when making resource allocation decisions. By considering the energy efficiency of available resources, the algorithms can intelligently allocate workloads to devices or servers with lower power consumption. This energy-aware decision-making ensures that tasks are assigned to the most energy-efficient resources, thereby reducing overall power consumption in the smart city environment [17].

Dynamic resource allocation enables adaptive scaling of resources based on workload demand. When the workload increases, resources can be scaled up to accommodate the additional processing requirements. Conversely, when the workload decreases, resources can be scaled down or placed in low-power modes to conserve energy. This dynamic scaling ensures that resources are utilized optimally, preventing unnecessary energy consumption during periods of low demand. Dynamic resource allocation facilitates task offloading, where computationally intensive tasks are offloaded from resource-constrained devices to more powerful edge servers or cloud infrastructure. By offloading tasks to devices with higher computing capabilities, energy-constrained devices can conserve energy and operate at lower power levels. Task offloading also enables workload distribution across the network, preventing devices from operating at maximum capacity and reducing overall energy consumption [18].

Dynamic resource allocation can leverage predictive analytics techniques to anticipate future workload patterns and resource requirements. By analyzing historical data and trends, the algorithms can make proactive decisions in allocating resources ahead of time. This predictive approach helps avoid resource shortages or over-provisioning, ensuring efficient resource utilization and minimizing unnecessary energy usage. Dynamic resource allocation continuously monitors the system's performance and adjusts resource allocation in real-time. By considering real-time data on workload, resource utilization, and energy consumption, the algorithms can optimize resource allocation dynamically. This real-time optimization ensures that resources are allocated based on the immediate demand, avoiding energy waste due to idle or underutilized resources [19].

Dynamic resource allocation can be integrated with energy management systems in smart cities. By exchanging information with energy monitoring systems, resource allocation algorithms can consider real-time energy

consumption data and adjust resource allocation accordingly. This integration enables a holistic approach to power efficiency, where resource allocation decisions align with the energy management goals of the smart city, resulting in optimized energy consumption.

Numerous researches have tried to address problems with resource allocation, particularly the allocation of processing resources, by figuring out which device's operational mode is the most efficient. The most important aspect of edge computing is the effective distribution of scarce resources, and the effectiveness of this process directly impacts how well the edge computing paradigm as a whole performs. Due to its enormous success, edge computing is becoming more and more popular, and the research community is paying close attention to it.

To address the problems of resource allocation and management in mobile edge computing (MEC), a broad variety of resource allocation methodologies and algorithms have been put forth. The effectiveness of edge computing jobs depends critically on the effective use of resources. This was discussed in a study published in [20], which offers the alternate direction method of multipliers (ADMM) algorithm to address the problems of resource allocation, content caching tactics, and compute offloading for MEC in wireless cellular networks. The suggested algorithm divides the main problem into smaller ones and finds distributed and efficient solutions for each one. Another effort in the same vein was published in [21], where the authors developed an algorithm for resource allocation in the distributed cloud environment by taking into account the quantity of virtual machines needed for the huge computing activities and the user requests for resources. The proposed work was also suitable for the optimal selection of datacenters in the edge computing environment. The objectives of the algorithm are to decrease the maximum latency and the distance between the selected datacenters.

MiLAN is a piece of software that was first introduced in [22] and is intended to manage and distribute network resources for applications that use data from different sensors and necessitate the choice of the best sensor set. Each sensor's quality, however, is fixed and cannot be altered. Furthermore, MiLAN only considers the network's bandwidth capacity and ignores its processing power. Additionally, because it does not simulate on-board processing, it is unable to accommodate a variety of offloading strategies.

According to the researchers in [23], efficient resource scheduling is essential for achieving resource efficiency. In order to control the distribution of computer resources in edge computing, they created a method called Zenith. The suggested method enables service providers (SPs) to contract with edge infrastructure providers for the pooling of resources. To enable tasks to satisfy their latency requirements and successfully complete, SPs use a latency-aware scheduling and resource provisioning method based on these contracts. For use in the edge/fog computing environment, the Time-Cost aware Scheduling (TCaS) method was put forth in [24]. For efficient resource allocation, the authors suggested a three-layered approach (edge devices layer, edge/fog layer, and cloud layer). The technique was tested using multiple datasets of tasks for edge/fog nodes and the cloud. Its goal is to distribute tasks to edge devices and edge/fog nodes. The trade-off between cost, time, and user comfort was one of the evaluation factors. The technique has a drawback, too, in that it allots resources before task processing and does not offer runtime resource allocation.

An investigation of MEC's energy efficiency and performance guarantee was done by Tao et al. [25]. They developed an optimization problem that reduces energy use while enhancing task performance and used Karush-Kuhn-Tucker criteria to solve it. They also created a request offloading method that details each time slot's bandwidth and energy requirements. This plan aims to lower energy usage while enhancing MEC performance. In multi-access edge computing environments, Mehrabi et al.'s [26] proposal for a heuristic-based approach with low complexity and minimal requirements for parameter adjustment. To demonstrate the effectiveness of the suggested solution and quantify the advantages of network-assisted adaptation over client-based techniques, they tested the optimized solution against two common client-based solutions, namely buffer-based adaptation and rate-based adaptation. The goal of this work is to create a mechanism that can effectively adapt to the shifting conditions of multi-access edge computing environments without requiring a large amount of CPU power or intricate parameter adjustment. To reduce the overall average latency of all mobile devices, Ren et al. [27] addressed the issue of joint computing and communication resource allocation. In terms of normalized cloud computation and backhaul communication capacities, they developed a closed-form optimal job allocation policy. In order to determine the best allocation of compute resources, they also converted the original issue into an

analogous convex optimization problem and solved it using a convex optimization method. In this study, we efficiently allocate computing and communication resources to reduce latency in mobile devices.

The framework presented by Wang et al. [28] addresses the challenge of balancing performance and power consumption in mobile service providers. To maximize system effectiveness, it is necessary to coordinate the scheduling of network resources within a cloud radio access network (C-RAN) and computation resources inside a mobile edge cloud computing (MEC). The authors develop a stochastic resource scheduling problem to address this issue. The decision-making process for distributing network and computing resources in a dynamic and uncertain environment is represented by this problem. Discovery the best resource scheduling method that maximizes system performance while minimizing power usage is the goal.

In [29], Hu and Li put forth an idea for reducing energy usage in mobile communication by optimizing the distribution of transmitting power among mobile users. To address the issue, the authors combine a quasiconvex method with a sub-gradient-based non-cooperative game model. In order to reduce request response time, they also model a simultaneous resource scheduling and request offloading problem. An enhanced fast and elitist multi-objective genetic algorithm is used to tackle this problem after it is first stated as a mixed-integer nonlinear program.

A multiuser mobile edge computing (MEC) system with energy harvesting devices is the subject of a study by Zhang et al. [8] that focuses on power consumption issues. The authors seek to reduce power usage while taking into account battery stability limits and quality of service (QoS) standards. The authors develop a stochastic optimization program to address this issue. Using this program as a mathematical model of the system, it is possible to define an objective function and constraints that account for the trade-off between power consumption, QoS, and battery stability.

To solve the stochastic optimization problem, the authors propose an online algorithm based on Lyapunov optimization. Lyapunov optimization is a technique that utilizes a Lyapunov function, which measures the system's energy and stability, to guide the optimization process. The online algorithm designed by Zhang et al. leverages the Lyapunov optimization framework and considers only the current state of users, making it suitable for real-time decision-making in dynamic environments.

Methodology

3.1 The proposed method

This chapter describes our efforts to create a precise online system for offloading edge computing tasks. The system model for our offloading situation is presented initially with the intention of resolving the prior raised difficulties in this thesis. The method we suggested is then included in a class of honest offloading methods that we define later. Then, we narrow down on our specific problem and look into possible ways to solve its special characteristics using the basic convex optimization techniques. Finally, we show the method we created as our recommended correction.

This research explores the concept of independent tasks for mobile users, characterized by parameters such as CPU cycle per bit, task data size, deadline, and priority value. The study introduces the concept of desired power consumption for each mobile device. The partial offloading process model aims to balance local execution and complete offloading models, focusing on determining the optimal fraction of tasks to be offloaded for each user. This decision impacts the energy efficiency and performance of the Mobile Edge Computing (MEC) system.

The offloading decision problem is an optimization challenge that aims to minimize energy consumption while satisfying latency requirements within the MEC system. This approach is particularly important in resource-constrained environments with limited battery capacity. The offloading decision problem under latency requirement constraints allows for a balance between response times and energy efficiency, making it crucial for applications requiring real-time processing. This approach opens up avenues for further exploration and innovation in mobile computing and edge intelligence.

3.2 Task Model

In this study, we investigate the idea of an independent task for distinct mobile users, designated as T=t_1,t_2,t_3,...,t_j. The parameters c_n, d_n, T_n, and P_n can be used to describe each task, denoted by t, which is submitted by a mobile user, and represented by n. These variables represent, in turn, the size of the task data, the task deadline, the priority value given to task t_n, and the needed CPU cycle per bit of tasks. We also define l_n as the amount of data that a mobile user n offloads, and _n as the percentage of tasks that user offloads. L_n is determined as the sum of d_n and _n, or l_n=d_n._n. In addition, we introduce the idea of desired power consumption for each mobile device, represented as E_(b,n). This figure acts as a starting point for calculating each mobile device's energy needs.

3.3 Offloading Decision Model

The concepts of local execution and complete offloading models are relatively easy to understand. Local execution refers to the scenario where all tasks are processed entirely on the mobile device itself, without involving any external resources. On the other hand, complete offloading involves transferring all tasks to a remote server or cloud for processing. These models have their advantages and limitations, but the partial offloading process model aims to strike a balance between the two.

In the partial offloading process model, the challenge lies in determining the optimal fraction of tasks to be offloaded for each mobile user. This decision is crucial as it directly impacts the energy efficiency and overall performance of the Mobile Edge Computing (MEC) system. By offloading a portion of tasks, the mobile device can leverage the computational capabilities and resources of the MEC infrastructure, thereby reducing energy consumption and potentially improving response times.

To address this challenge, we formulate the offloading decision problem as an optimization challenge that focuses on maximizing energy efficiency while satisfying latency requirements within the MEC system. The goal is to find the optimal offloading fraction that minimizes energy consumption while ensuring that the latency constraints are met. This formulation draws inspiration from Tao's research [46], which provides valuable insights into the energy-latency trade-off in MEC systems.

By considering the energy efficiency aspect, we aim to minimize the power consumption of mobile devices while still meeting the required latency thresholds. This is particularly important in resource-constrained environments where mobile devices have limited battery capacity. By intelligently allocating the tasks between local execution and offloading, we can optimize the energy usage of mobile devices and prolong their battery life.

3.3.1 Optimization Offloading Model

This chapter presents a comprehensive assessment of the proposed approach to analyzing traffic patterns in mobile devices. The approach is evaluated using a simulation environment that accurately replicates real-world scenarios and parameters. A simulator was developed using MatLab software in 2019 to provide a versatile platform for evaluating the methodology across various scenarios and conditions. The simulator offers an intuitive interface, allowing users to customize simulation parameters, fine-tune algorithmic settings, and visualize results.

The proposed method is compared against four alternative approaches: "All Local," "All Offload," "All Offload+Priority," and "Random." The "All Local" strategy focuses on executing tasks locally without offloading them to external servers, leveraging the computing capabilities of mobile devices. This approach reduces reliance on external servers, minimizes network latency, enhances data privacy and security, and distributes computational workload more evenly across the network. However, the effectiveness of this strategy depends on the computational capabilities of individual mobile devices.

The all-local process model focuses on tasks performed locally without task transmission, resulting in energy consumption based on task data size and CPU requirements. The offload process model considers only energy consumption for transmission, while the all-local process model includes both transmission and execution time. The energy consumption of the partial offload process model is divided into local execution consumption and partial offloading tasks transmission consumption. The energy consumption of each mobile device can be divided into local execution consumption and transmission consumption.

The "All Offload" strategy offloads tasks to edge layer servers, leveraging their superior processing power and abundant resources. This strategy allows mobile devices to conserve their limited battery power and computational resources, extending their battery life and enabling users to utilize their devices for longer durations. Edge servers are typically equipped with more powerful processors, larger memory capacities, and faster network connections compared to mobile devices, enhancing efficiency and speed of task execution.

The energy consumption of the all offload process model can be defined as E_o and $t_{n,o}$, respectively.

$$E_o = \frac{d_n \cdot p_n}{r_n} = p_n \, . \, t_n \quad \textbf{(4-3)}$$

$$t_{n,o} = \frac{d_n}{r_n} + \frac{c_n}{h_n^e} \quad \textbf{(4-4)}$$

Let p_n represent the transmission power for mobile n, and let r_n denote the transmission rate of mobile n. The transmission rate of mobile n can be defined as equation **(**4-5**)**.

$$r_n = \beta(1 + \frac{p_n \, g_n^2}{N_o \, \beta}) \quad \textbf{(4-5)}$$

Let β represent the bandwidth of the wireless channel, and let g_n denote the channel gain of the edge server.

The Mobile Layer Algorithm for Energy-Efficient Task Offloading aims to optimize task offloading for each mobile user by iterating over each user and task. The optimal offloading fraction, denoted as α_n, is calculated using equations that consider factors such as network conditions, device capabilities, and task characteristics. Once the optimal offloading fraction is calculated, tasks are offloaded to the edge server, which helps conserve energy and improve the overall performance of the mobile device.

Table 4-1 Description of Energy-Efficient Task Offloading Algorithm

Energy-Efficient Task Offloading Algorithm (*Part A***)**	
Line	***Code***
1:	***for*** *each mobile user n*
2:	***for*** *each task* d_i *in mobile n*
3:	*Calculate the optimal offloading fraction* α_n *by* **(**4-20**)**
4:	*Offload* $d_i \times \alpha_n$ *tasks to edge server;*
5:	*Execute* $d_i \times (1 - \alpha_n)$ *tasks at local mobile*
6:	***end for***
7:	***end for***

3.4 Performance Evaluation

Energy consumption has been chosen as one of the primary measures for our performance assessment. On the other hand, energy consumption measures how much power is used when a task is carried out on a mobile device. It is essential for assessing the device's power effectiveness. We may evaluate a device's sustainability and how effectively it uses its power resources by evaluating its energy usage. In order to increase battery life and lessen the environmental impact of using mobile devices, energy consumption must be kept to a minimum.

Energy Consumption:

The partial offload process model is used to optimize energy efficiency and task execution by dividing computational tasks between mobile devices and edge servers. The energy consumption is quantified using a variable α_n, which represents the offloading fraction to the edge server. The model consists of local execution and transmission consumption, allowing for better decision-making regarding task allocation strategies and resource management techniques. The energy consumption model can be expressed mathematically as equation (4 6), providing valuable insights into the process.

$$E_n = E_o \,.\, \alpha_n + E_l \,.\, (1 - \alpha_n) = \; p_n \,.\, t_n . \alpha_n + f_n \,.\, c_n . (1 - \alpha_n) \qquad \textbf{(}4\text{-}6\textbf{)}$$

In summary, the energy consumption of the partial offload process model considers the interplay between local execution and task offloading to the edge server. By carefully balancing the task offloading fraction and optimizing the energy consumption components, we can achieve better energy efficiency and enhance the overall performance of mobile computing systems.

Results

Within this section, we will present the simulation results obtained from the simulations conducted using the aforementioned settings. These simulations were carefully developed in order to assess the performance and usefulness of the system in question.

We wanted to get deeper understanding of the system's behavior, potential, and limitations by using these models. The results of these simulations provide a detailed understanding of how the system behaves and reacts in various situations.

Evaluation based on the number of tasks

An early experiment was carried out to assess the efficacy of the suggested strategy, with an emphasis on the quantity of tasks generated and carried out on mobile devices. Figures 5-2 show the outcomes of this experiment and their implications for execution time and energy consumption, respectively.

Energy consumption

Figure 5-2, on the other hand, sheds light on the energy consumption associated with the execution of tasks. It quantifies the amount of energy utilized by the mobile devices during the execution process. Understanding the energy consumption is crucial for evaluating the method's power efficiency and sustainability. Lower energy consumption suggests that the proposed method efficiently utilizes the device's power resources, leading to extended battery life and reduced environmental impact.

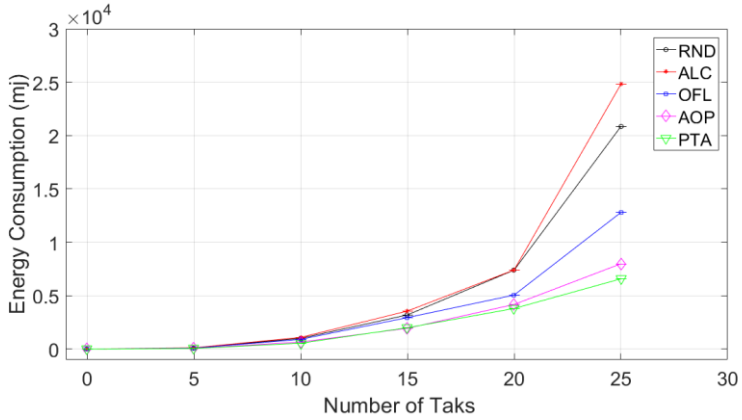

Figure 5-2 Comparison of Energy Consumption based on the Number of Tasks for the proposed method and the benchmarks.

Discussion

The comprehensive analysis of Figure 5- 2 allows us to obtain a holistic understanding of the performance of the proposed method when applied to mobile devices. The method's efficiency, efficacy, and sustainability are all useful information that can be gained from the evaluation based on execution time.

Similarly, Figure 5- 2 provides crucial information about the energy consumption associated with the execution of tasks. We may assess the method's power efficiency and sustainability by looking at the findings of the energy consumption analysis. A strategy that uses the power resources of the device more efficiently will result in a lower energy consumption, longer battery life, and a smaller environmental impact. Making decisions about the method's resource utilization and energy optimization is made easier by these insights.

4.2 Evaluation based on task data size

To further evaluate the effectiveness of the proposed method, a second experiment was conducted, centering on the size of tasks generated and executed on mobile devices. This experiment aimed to investigate how the task size affects the performance of the method. The results of this experiment are presented in Figure 5-4, showcasing the findings regarding execution time and energy consumption, respectively.

Energy consumption

Figure 5-4, on the other hand, focuses on the energy usage related to various-sized tasks. It measures the amount of energy used when carrying out tasks of various complexity. The energy efficiency of the approach can be assessed by examining the energy consumption figures shown in Figure 5-4, and any patterns or trends connected to job size can be found. This knowledge can help us make decisions on how to best manage power, allocate resources, and optimize energy use in order to increase the proposed method's overall effectiveness and sustainability.

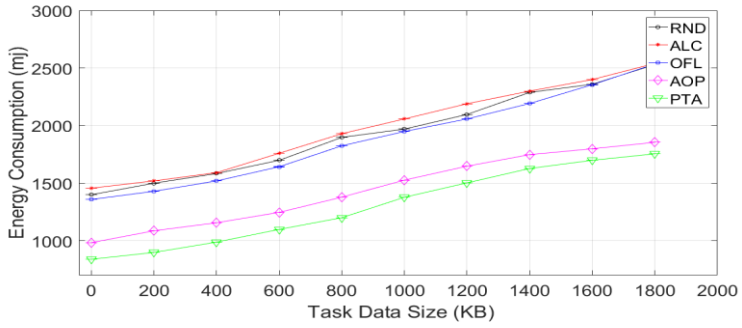

Figure 5-4 Comparison of Energy Consumption based on the Task Data Size for the proposed method and the benchmarks.

Discussion

Indeed, examining Figure 5- 4 in conjunction allows us to acquire valuable insights into the performance characteristics of the proposed method concerning task size. A thorough picture of how the method operates while handling jobs of various complexities or sizes can be gained from the evaluation based on execution time and energy usage.

Figure 5- 4 simultaneously displays the data on energy consumption for various work sizes. We may assess the method's power efficiency and spot any patterns or trends related to job size by looking at the energy consumption figures. This study helps us decide how to best use resources, allocate energy, and control power in order to improve the proposed method's overall effectiveness and sustainability.

4.3 Evaluation based on exponential distributions

A third experiment, which concentrated on the exponential distribution of task size, was conducted to further evaluate the efficacy of the suggested approach. The purpose of this experiment was to examine how well the approach performs when the size of the tasks is distributed exponentially. Figure 5-6 show the data from this experiment, which are connected to execution time and energy use, respectively.

Energy consumption

Figure 5-6 focuses on the energy consumption associated with tasks following an exponential distribution of size. It measures the amount of energy used when jobs of various sizes and distributions are carried out. We can assess the method's power effectiveness and find any trends or patterns associated with the distribution of job sizes by examining the energy consumption figures shown in Figure 5-6. In scenarios with an exponential distribution of task sizes, this knowledge enables us to optimize energy consumption, resource allocation, and power management tactics to increase the overall efficiency and sustainability of the suggested strategy.

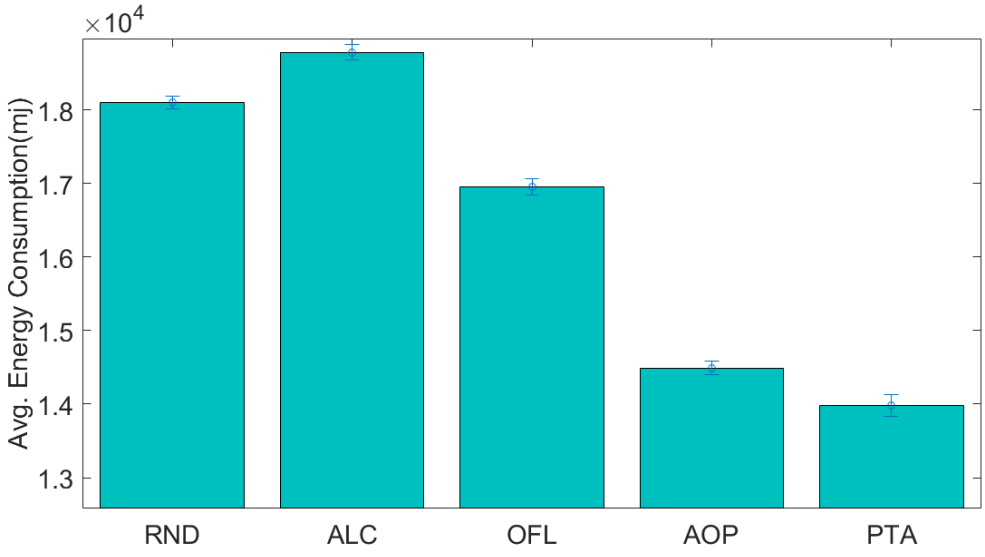

Figure 5-6 Comparison of the proposed method and the benchmarks in terms of Energy consumption.

Discussion

Absolutely, examining Figure 5- 6 allows us to gain valuable insights into the performance characteristics of the proposed method when tasks follow an exponential distribution of size. The evaluation based on execution time and energy consumption provides a comprehensive understanding of how the method performs in such scenarios.

Similar insights into energy usage are provided by Figures 5- 6 for tasks whose sizes follow an exponential distribution. We may assess the method's power efficiency and spot any patterns or trends unique to this distribution by looking at the energy consumption results. In scenarios where task sizes follow an exponential

distribution; this study directs us in optimizing energy utilization, resource allocation, and power management strategies to increase the overall efficiency and sustainability of the suggested method.

Conclusion

The findings demonstrate that the Joint Computation Offloading and Prioritized Scheduling technique successfully reduces task completion time and energy consumption, leading to quicker task execution and shorter wait times. By shifting workloads to the mobile-edge computing infrastructure, it also reduces the amount of energy used by powerful edge servers and resource-constrained devices. The method also improves the mobile-edge computing system's Quality of Service (QoS) metrics, which enables more tasks to be completed in a given amount of time.

The study reveals that task offloading to edge servers significantly reduces energy consumption for mobile devices compared to executing all tasks locally. This is because these servers' utilization of computational resources and capabilities lessens the demand on scarce power resources. Mobile devices operate more effectively and for a longer period of time as a result. Additionally, the results imply that partial offloading, or selective offloading, uses less battery power for mobile devices than full offloading. By utilizing the edge server's processing power while reducing the amount of wireless data transfer that uses a lot of energy, this method maximizes energy consumption. The research highlights the significance of taking task characteristics and related data quantities into account when developing offloading mechanisms.

The study investigates the relationship between changes in task-related delay requirements and energy use. It demonstrates that using the edge server's greater processing capability to offload jobs results in lower energy consumption. However, despite changes in latency requirements, every local execution uses the same amount of energy. In order to optimize task allocation and offloading strategies based on particular application requirements, the study emphasizes the significance of taking latency requirements into account when making decisions on task offloading.

References:

[1] Sinha, B. B. and Dhanalakshmi, R., "Recent advancements and challenges of the Internet of Things in smart agriculture: A survey," Future Generation Computer Systems, vol. 126, pp. 169-184, 2022.

[2] Ketu, S. and Mishra, P. K., "A contemporary survey on IoT based smart cities: architecture, applications, and open issues," Wireless Personal Communications, vol. 125, no. 3, pp. 2319-2367, 2022.

[3] Saeik, F. *et al.*, "Task offloading in Edge and Cloud Computing: A survey on mathematical, artificial intelligence and control theory solutions," *Computer Networks*, vol. 195, p. 108177, 2021.

[4] Mell, P. and Grance, T., "The NIST definition of cloud computing," 2011.

[5] Sadiku, M. N., Musa, S. M., and Momoh, O. D., "Cloud computing: opportunities and challenges," *IEEE potentials*, vol. 33, no. 1, pp. 34-36, 2014.

[6] Hu, S. and Xiao, Y., "Design of cloud computing task offloading algorithm based on dynamic multi-objective evolution," *Future Generation Computer Systems*, vol. 122, pp. 144-148, 2021.

[7] Chen, M., Wang, L., Chen, J., Wei, X., & Lei, L. (2019). A computing and content delivery network in the smart city: Scenario, framework, and analysis. IEEE Network, 33(2), 89-95.

[8] Zhang, T., Xu, Y., Loo, J., Yang, D., and Xiao, L., "Joint computation and communication design for UAV-assisted mobile edge computing in IoT," IEEE Transactions on Industrial Informatics, vol. 16, no. 8, pp. 5505-5516, 2019.

[9] Li, J. (2020). Resource optimization scheduling and allocation for hierarchical distributed cloud service system in smart city. Future Generation Computer Systems, 107, 247-256.

[10] Kai, C., Zhou, H., Yi, Y., and Huang, W., "Collaborative cloud-edge-end task offloading in mobile-edge computing networks with limited communication capability," *IEEE Transactions on Cognitive Communications and Networking*, vol. 7, no. 2, pp. 624-634, 2020.

[11] Hao, Y., Jiang, Y., Chen, T., Cao, D., and Chen, M., "iTaskOffloading: intelligent task offloading for a cloud-edge collaborative system," *IEEE Network*, vol. 33, no. 5, pp. 82-88, 2019.

[12] Intel, I., "and ia-32 architectures software developer's manual, 2011," *Intel order Number*, vol. 64, p. 64.

[13] Pahl, C., "Containerization and the paas cloud," *IEEE Cloud Computing*, vol. 2, no. 3, pp. 24-31, 2015.

[14] Demars, A., "Analysis and simulation of a fair queueing algorithm," *ACM SIGCOM, 1989*, 1989.

[15] Pabla, C. S., "COMPLETELY FAIR SCHEDULER-Linux's latest scheduler makeover," *Linux journal*, no. 184, p. 68, 2009.

[16] Chisnall, D., *The definitive guide to the xen hypervisor*. Pearson Education, 2008.

[17] Xavier, M. G., Neves, M. V., Rossi, F. D., Ferreto, T. C., Lange, T., and De Rose, C. A., "Performance evaluation of container-based virtualization for high performance computing environments," in *2013 21st Euromicro International Conference on Parallel, Distributed, and Network-Based Processing*, 2013, pp. 233-240: IEEE.

[18] Miao, Y., Wu, G., Li, M., Ghoneim, A., Al-Rakhami, M., and Hossain, M. S., "Intelligent task prediction and computation offloading based on mobile-edge cloud computing," *Future Generation Computer Systems*, vol. 102, pp. 925-931, 2020.

[19] Shakarami, A., Shahidinejad, A., and Ghobaei-Arani, M., "An autonomous computation offloading strategy in Mobile Edge Computing: A deep learning-based hybrid approach," *Journal of Network and Computer Applications*, vol. 178, p. 102974, 2021.

[20] 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, vol. 16, no. 8, pp. 4924-4938, 2017.

[21] Ziafat, H. and Babamir, S. M., "A method for the optimum selection of datacenters in geographically distributed clouds," The Journal of Supercomputing, vol. 73, pp. 4042-4081, 2017.

[22] Heinzelman, W. and Murphy, A., "Milan: Middleware Linking Applications and Networks," 2003.

[23] 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, pp. 47-54: IEEE.

[24] 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 9th International Symposium on Information and Communication Technology, 2018, pp. 397-404.

[25] Tao, X., Ota, K., Dong, M., Qi, H., and Li, K., "Performance guaranteed computation offloading for mobile-edge cloud computing," IEEE Wireless Communications Letters, vol. 6, no. 6, pp. 774-777, 2017.

[26] Mehrabi, A., Siekkinen, M., and Ylä-Jääski, A., "Edge computing assisted adaptive mobile video streaming," IEEE Transactions on Mobile Computing, vol. 18, no. 4, pp. 787-800, 2018.

[27] Ren, J., Yu, G., He, Y., and Li, G. Y., "Collaborative cloud and edge computing for latency minimization," IEEE Transactions on Vehicular Technology, vol. 68, no. 5, pp. 5031-5044, 2019.

[28] Wang, X. et al., "Dynamic resource scheduling in mobile edge cloud with cloud radio access network," IEEE Transactions on Parallel and Distributed Systems, vol. 29, no. 11, pp. 2429-2445, 2018.

[29] Hu, S. and Li, G., "Dynamic request scheduling optimization in mobile edge computing for IoT applications," IEEE Internet of Things Journal, vol. 7, no. 2, pp. 1426-1437, 2019.