# Blockchain based energy efficient multi-tasking optimistic scenario for mobile edge computing

PeerJ Computer Science

# Blockchain based energy efficient multi-tasking optimistic scenario for mobile edge computing

Jianbin Wu[1], Sami Ahmed Haider[2], Mukesh Soni[3], Ashima Kalra[4] and Nabamita Deb[5]

[1] Computer Science and Engineering Department, Zhejiang Normal University, Jinhua, Zhejiang, China
[2] Computing Department, University of Worcester, Worcester, United Kingdom
[3] Changdigarh University, Department of CSE, University Center for Research and Development, Mohali, Punjab, India
[4] ECE Department, Chandigarh Engineering College, Landran, Mohali, India
[5] Department of Information Technology, Gauhati University, Assam, India

## ABSTRACT

Mobile edge computational power faces the difficulty of balancing the energy consumption of many devices and workloads as science and technology advance. Most related research focuses on exploiting edge server computing performance to reduce mobile device energy consumption and task execution time during task processing. Existing research, however, shows that there is no adequate answer to the energy consumption balances between multi-device and multitasking. The present edge computing system model has been updated to address this energy consumption balance problem. We present a blockchain-based analytical method for the energy utilization balance optimization problem of multi-mobile devices and multitasking and an optimistic scenario on this foundation. An investigation of the corresponding approximation ratio is performed. Compared to the total energy demand optimization method and the random algorithm, many simulation studies have been carried out. Compared to the random process, the testing findings demonstrate that the suggested greedy algorithm can improve average performance by 66.59 percent in terms of energy balance. Furthermore, when the minimum transmission power of the mobile device is between five and six dBm, the greedy algorithm nearly achieves the best solution when compared to the brute force technique under the classical task topology.

**Subjects** Artificial Intelligence, Computer Networks and Communications, Mobile and Ubiquitous Computing, Security and Privacy, Blockchain

**Keywords** Mobile edge computing, Energy balance, Greedy algorithm, Blockchain, Task offloading

Submitted 22 March 2022
Accepted 5 September 2022
Published 17 October 2022

Corresponding author
Sami Ahmed Haider,
s.ahmedhaider@worc.ac.uk

Academic editor
Lisu Yu

Additional Information and Declarations can be found on page 18

DOI **10.7717/peerj-cs.1118**



## INTRODUCTION

With advancements in science and technology, time-sensitive applications such as augmented reality, virtual reality, and real-time gaming have exploded in popularity in recent years. The creation of such apps often necessitates the use of high-performance machine assistance. However, as mobile devices become more widespread, users are more likely to utilise them to do a variety of jobs. When a user wants to do many tasks on a mobile device, including time-sensitive apps, the device's computational power and battery

life must be adequate. However, a mobile device's volume is frequently limited owing to its mobility, and its computational power and durability cannot be guaranteed. To address this issue, the European Telecommunications Standards Institute (ETSI) proposed mobile edge computing (MEC). MEC has advanced quickly in recent years, and it now has critical applications in video content transmission, autonomous automobiles, and other industries.

Furthermore, data sharing on the Internet has expanded due to the fast growth of the Internet of Things and fifth-generation communication technologies. According to a Cisco white paper, the number of mobile devices *per capita* will reach 1.5 by 2022, with mobile devices accounting for 20% of worldwide IP traffic (*De Nitto Personè & Grassi, 2019*). When all of the data created by these mobile devices is delivered to and finished by the central cloud server, network congestion is inevitable, however its use MEC to overcome these issues (*Reiter, Prünster & Zefferer, 2017*; *Li et al., 2016*).

By supporting mobile devices with their duties, MEC can lower their energy usage and time to execute these activities. However, owing to the restricted processing capacity of mobile devices, improving their computational capability while preserving energy has become a difficulty (*Wei et al., 2017*). As a result, most current research focuses on employing edge servers or distant clouds to boost mobile device processing capability (*Li & Wang, 2018*; *Ren et al., 2018*), such as using edge servers and cloud servers to reduce overall energy consumption (*Subramanya et al., 2017*; *Muniswamaiah, Agerwala & Tappert, 2021*; *Dolui & Datta, 2017*; *Li et al., 2018*) or task completion time (*Li et al., 2018*; *Kim & Hong, 2019*). However, the majority of current investigations ignore the issue of balancing mobile device energy consumption in the context of computing job offloading in the presence of many mobile devices (*Zeng et al., 2020*; *Kim et al., 2020*; *Marjanović, Antonić & Žarko, 2018*; *Takahashi, Tanaka & Kawamura, 2015*).

The goal of multi-device multi-energy tasking's balancing challenge is to reduce the maximum energy usage of numerous devices. Consider the following scenario: e is an edge server, and a and b is two mobile devices. For the sake of simplicity, both mobile devices a and b are supposed to have two jobs that must be done within two time periods. Consider that edge servers and mobile devices can only conduct one task each period to simplify the description. Furthermore, the energy consumption cost is reduced when the job is given to the edge server for execution. Assume, however, that e is given both subtasks in a. In such instance, b can no longer access the edge server, causing device b's energy consumption to be much greater than that of device a, resulting in an imbalanced energy consumption. As a result, in order to attain energy balance, the work scheduling method must be modified (*Badri et al., 2018*; *Hung, Hsieh & Wang, 2017*; *Li et al., 2021*; *Dolui & Datta, 2017*; *Abbas et al., 2018*; *Zhao & Yang, 2021*; *Li et al., 2021*).

*Zeng et al. (2020)* and *Kim et al. (2020)* are thought to be the most comparable to this research. The literary structure (*Zeng et al., 2020*) is similarly made up of three layers. It takes into account task dependencies, but its purpose is to reduce overall energy consumption rather than the balance of mobile device energy usage. The device, the edge, and the cloud are the three components of this article, and the usage of relay devices to improve the system is not explored. Multiple original devices and a wireless access point

(AP) are merged by an edge server in the literature framework (*Kim et al., 2020*). Under two users, the study minimises the weighted total of energy use. However, this document does not employ relay devices, and the jobs in this document may be split arbitrarily, with no regard for task interdependence.

The majority of related research focuses on making use of edge server computing performance to cut down on mobile device energy utilization and task processing time. However, existing research demonstrates that the energy consumption balances between using many devices and multitasking are intractable. This energy consumption balancing issue has been addressed by updating the model for the current edge computing system. Based on this, we provide a blockchain-based analytical solution for the energy usage balance optimization problem of many mobile devices and multitasking, along with an optimistic scenario.

As a result, based on the aforementioned articles, this study improves the current work by removing the cloud server with higher energy consumption from the original structure and adding the relay device node as the task's transit node, resulting in a mobile device. Machines and edge servers make into a three-tier structure. The following are the contributions of this article:

1. A multi-device and multi-task energy balancing greedy algorithm (GA) MMG (min-max greedy algorithm) that meets task dependence is created based on the above structure. The MMG method outperforms current minimal energy consumption algorithms while solving the multi-device multi-task energy balance issue with task dependencies.
2. The energy balancing problem is recast as a problem of minimising maximum energy consumption, and the total energy consumption optimization method and the random algorithm are compared to the MMG algorithm.
3. A significant number of comparison tests were conducted to prove the superiority of the MMG algorithm. The testing findings reveal that the MMG algorithm's average performance in terms of energy balance may be enhanced by 66.59 percent.

## RELATED WORK

In the research of MEC, most of the teams focus on how to make full use of server resources to improve the performance of mobile devices. The goals generally fall into two categories: saving energy costs and reducing task completion time.

In terms of saving energy costs, *Marjanović, Antonić & Žarko (2018)* designed a dynamic computing offloading algorithm and a corresponding online offloading algorithm to solve the emotional computing offloading problem and keep the energy consumption of mobile devices. *Takahashi, Tanaka & Kawamura (2015)* proposed a scheduling algorithm to allocate wireless bandwidth to minimize the total cost for all users. *Badri et al. (2018)* offers two offloading strategies to reduce energy consumption under delay constraints. *Hung, Hsieh & Wang (2017)* proposed edge cloud architecture and designed a greedy algorithm to minimize the total energy consumption of mobile devices. *Li et al. (2021)*

developed a distributed algorithm that combines "0-1" programming and coalition games and minimizes the total energy consumption of mobile devices by sharing computing results among mobile users. *Samanta & Li (2018)* studies the problem of reducing the total execution cost of an application under time constraints in an architecture consisting of a remote cloud and heterogeneous local processors. *Abbas et al. (2018)* proposed a distributed linear relaxation heuristic and a greedy heuristic to minimize the total energy consumption of mobile users. *Zhao & Yang (2021)* considers the energy consumption problem and the scheduling strategy of sensitive tasks and designs a distributed dynamic offloading and resource scheduling strategy. The goal is to minimize the energy consumption of mobile devices, but the energy consumption between devices is not considered—an equilibrium problem. None of those mentioned above articles has fully considered the task completion time, nor have they thought of the cooperative role of relay devices to optimize the energy consumption of mobile devices better.

To reduce the task completion time, *Li et al. (2021)* proposes an efficient one-dimensional search algorithm by using the Markov decision method to minimize the average delay of each computing task. *Guo et al. (2020)* designed a centralized, distributed, and greedy maximum scheduling algorithm on the multi-user multi-task problem to solve the multi-user multitask computing offload problem. Still, the article did not consider the task dependencies. *Jiang, Li & Wu (2019)* proposed a "dependency-aware" offloading scheme for "edge-cloud" collaboration and designed two algorithms to minimize the task completion time of the device under task dependency constraints and a given budget. *Asheralieva & Niyato (2021)* assumes that the resources of the edge server are infinite, and researches the problem of minimizing the total delay of mobile users under any given computing offloading strategy under the heterogeneous delay-sensitive computing task environment where different mobile users arrive randomly. The problem is modeled as a dynamic priority queue, and a priority transmission scheduling strategy is designed to solve it. However, none of the above articles has considered the use of relay devices (*Guo et al., 2020*; *Jiang, Li & Wu, 2019*; *Asheralieva & Niyato, 2021*; *Wu et al., 2021*).

In addition to energy consumption and task completion time, some research is devoted to solving other intractable problems encountered by edge computing. *Wu et al. (2021)* designs a distributed computing offload algorithm using game theory and studies the multi-user offload problem in the multi-channel wireless interference environment. *Queralta et al. (2020)* uses evolutionary game strategy to study the problem of multi-user computing offloading in dynamic environments. Finally, *Xu et al. (2020)*, *Huang et al. (2022)*, and *Hou et al. (2020)* exploits the mobility of UAVs to solve the low mobility problem of edge servers (*Queralta et al., 2020*; *Xu et al., 2020*; *Huang et al., 2022*; *Hou et al., 2020*).

## CONSENSUS MECHANISM FOR BLOCK CHAIN

The blockchain is a decentralised storage system with no administrators, in which each node owns all data. Due to its unique trust building method, blockchain is widely employed in the global deployment of the Internet of Vehicles (*Reiter, Prünster & Zefferer, 2017*), Internet of Things (*Li et al., 2016*), financial services (*Wei et al., 2017*; *Li & Wang,*

2018), smart grid (Ren et al., 2018) and other industries as a new computing paradigm and Collaboration mode (De Nitto Personè & Grassi, 2019). Several key directions for the development of the contemporary burgeoning digital sector include blockchain (Dolui & Datta, 2017), big data (Li et al., 2018), artificial intelligence (Kim & Hong, 2019), cloud computing, and network security.

The consensus mechanism refers to making nodes agree on the content in the distributed database in the process of dynamic transactions. The blockchain uses the consensus mechanism to make nodes reach a consensus on transactions, thereby weakening the function of the centralized supervision system. From the initial proof of work (PoW), practical Byzantine fault tolerance (PBFT), proof of stake (PoS) to later delegated proof of stake (DPoS), a series of consensus mechanisms such as proof of authority (PoA) (Marjanović, Antonić & Žarko, 2018). The consensus mechanism has been continuously improved, and it has evolved in different directions corresponding to additional field requirements. However, as the core technology of the blockchain, the consensus mechanism can effectively reach a consensus on the data of each node in the blockchain, complete the transaction data processing quickly, and ensure the consistency and reliability of the data. Its typical consensus mechanism is analyzed in detail as follows.

**(A) Proof of Work (PoW):** This work first proposed the idea of Proof of Work to increase the cost of spammers by calculating a specific mathematical problem (Takahashi, Tanaka & Kawamura, 2015). The PoW consensus mechanism was first introduced in the article in 1999, which also laid the foundation for the consensus mechanism used in Bitcoin proposed by Satoshi Nakamoto in later generations (Hou et al., 2020). Proof of work is one of the most typical algorithms in blockchain consensus mechanisms, such as the mining process in Bitcoin. "Miners" obtain a particular Bitcoin reward by continuously trying to calculate a random number N that meets the mining difficulty (Difficulty), as shown in Formula (1):

$$N(BlockHeader) \leq target \tag{1}$$

The difficulty value belongs to a tiny part of the target value range (Target) in the $2^{256}$ input space, as shown in Formula (2):

$$Targe = target * \frac{actualtime}{expectedtime} \tag{2}$$

In the blockchain system, the block will dynamically adjust the difficulty of the threshold in a certain period (every 2 016 blocks, about 2 weeks), as shown in Formula (3):

$$Difficulty = \frac{difficulty//target}{Target} \tag{3}$$

when the system's mining difficulty (target) remains unchanged, the actual mining difficulty is greater than the expected mining difficulty. The right side of the equal sign is greater than 1, the target threshold increases, and the mining difficulty decreases simultaneously. The mining difficulty is proportional to the target threshold difficulty.

When the number of miners increases and the speed of block generation is significantly accelerated, the system will increase the difficulty of mining. The rate of block generation tends to be balanced (generally, a block is generated every 10 min). *Li et al. (2018)* proposed the use of the Byzantine fault tolerance algorithm (practical Byzantine fault tolerance, PBFT), which was applied to the digital asset platform with small throughput but a large number of events, which improved the efficiency based on the original Byzantine algorithm (*Hung, Hsieh & Wang, 2017*). The PBFT algorithm stipulates that at least 3f + 1 node need to be deployed in the whole network, which can tolerate up to If there are f malicious nodes, if a Byzantine failure occurs, the state of the entire system is determined by 2f + 1 nodes, that is, on the premise of ensuring system activity and security, a consensus is reached when the number of malicious nodes in the entire network is less than 1/3. However, the famous scientist Professor Eric proposed that distributed systems can only satisfy two of the three aspects of consistency, security and partition fault tolerance, and the PBFT algorithm could not fulfil the ecosystem at that time. Recently, Proof of Stake (PoS) first appeared in Peercoin. It uses the concept of "coinage" to control the amount of currency in the hands of miners and stipulates that currency holders must have a specific period. The longer, as shown in Formula (4) (*Samanta & Li, 2018*):

$$Coin\ age = currency\ amount \times holding\ time \tag{4}$$

To ensure the system's fairness, the higher the coin age of miners, the lower the difficulty of mining, which can reduce the possibility of users being attacked to a certain extent. Furthermore, the emergence of PoS has improved the phenomenon of excessive computing power consumption in PoW, and to a certain extent, has alleviated the previous inefficiency caused by the slow block generation time, increasing the throughput and speeding up the processing speed, but if the system The phenomenon of the richest man is prominent, which will cause the problem of centralization (*Marjanović, Antonić & Žarko, 2018*).

In Formulas (1)–(3), actual-time and expected time are the actual mining difficulty and expected mining difficulty, respectively; target and difficulty are the target threshold difficulty and mining difficulty of the system, respectively; Difficulty||Target is the actual mining setting for the system Difficulty value, the minimum is 1, and Target is the target threshold.

**(B) Authorized Proof of Stake (DPoS) (*Takahashi, Tanaka & Kawamura, 2015*):** DPoS is based on the form of democratic voting. Nodes elect N members to become the "delegation" in the system, and the node with more tokens has a higher probability of becoming a "representative". The "representative" node in the group is responsible for collecting information, packaging transactions, and verifying transactions and newly produced blocks. The time slice is used to allocate time to the "representative" node to process things. If there is a malicious "representative", the node will be revoked to block and cancel the "representative" resource.

Grid, and then select a new "representative". The emergence of DPoS reduces the waste of computing power and electricity. Also, it improves the transaction processing speed and

blocks throughput, but at the same time, it inevitably weakens the ability of the decentralized working model.

**(C) Proof of Authority (PoA):** Gavin, the founder of Ethereum, first proposed the proof-of-authority consensus mechanism in 2017. The PoA consensus mechanism is mainly used for reputation accumulation, and the validator needs to verify the user's identity, not the currency held by the user (*Zeng et al., 2020*). A user who wants to verify a transaction first confirms their identity, links it to the verification performed, and stores it on the blockchain. When a transaction is verified, the validator's identity is confirmed on-chain through some protocol. This identity is only determined by a small group of validators, increasing the efficiency and security of the consensus protocol. PoA does not require high computational costs or accumulation of large amounts of tokens, but it only works on private blockchains and consortium blockchain networks. In July of the same year, the Hyperledger community officially released Fabric 1.0. The emerging consensus mechanism broke the last impression of a proof-based consensus mechanism and formed an endorsing peer, orderers, and committing peers. Fabric consensus mechanism based on three types of nodes (*Li et al., 2016*). The Hyperledger consensus process contain following steps as:

**Step 1** The client (client SDK) creates a proposal and sends the proposal to the corresponding endorsement node according to the selected endorsement policy. The proposal contains information such as the user ID (ClientID) and the called chaincode function (BlockchainCoin function) and its parameters, timestamp (Timestamp) and client signature (ClientSig) (*Wei et al., 2017*).

**Step 2** The endorsing node 1erifies the client signature to ensure that an authenticated client sent the proposal. Then, the transaction request simulated by the endorsement node is executed according to the chain code in the proposal, and the endorsement node's signature (sign TX endorsed) is appended to the generated execution result, that is, the endorsement process. The development of the simulation execution is a set of readset and writeset set based on the current world state (state database), and the endorsement signature and writeset set are sent to the client after endorsement.

**Step 3** The client verifies the signature of the endorsement result to ensure that it comes from a legitimate endorsement node. The client can check the validity of the endorsement node's signature and the consistency between the read and write sets received from different endorsement nodes. The result generates a transaction and broadcasts it to the ordering nodes. The client can enforce mandatory checks later in the validation phase to help detect transaction failures earlier in the transaction flow to reduce overhead.

**Step 4** Order transactions using the Kafka schema in Hyperledger Fabric. The sorting node hands the received transactions to the Kafka cluster for sorting and reads a certain number of ordered transactions from the Kafka cluster according to specific rules, and packs them into blocks. After the ordering service signs the league, it distributes the block to submitting nodes.

**Table 1** Symbols used in this article.

| Symbol | Mobile device set |
|---|---|
| N | Subtask set |
| M | Relay Device Set |
| H | Edge server |
| h | The workload size of the jth subtask for the ith mobile device |
| $X_{i,j}$ | The uplink data rate between the ith mobile device and the kth relay and the uplink data rate between the kth relay and the edge server |
| $S_{ik}$, $S_k$ | The data size of the jth subtask of the ith mobile device |
| $E_{i,j}$ | The computing time and local energy consumption of the i-th mobile device itself to process the j-th subtask |
| $u_{i,j}^{Local}$, $F_{i,j}^{Local}$ | While transferring from the ith mobile device to the kth relay device, the jth subtask requires time and energy. |
| $u_{i,j,k}^{Send}$, $F_{i,j,k}^{Send}$ | Time and energy used by the ith mobile device's jth subtask to send data from the kth relay mechanism to the network edge |
| $u_{i,j,k}^{Send2}$, $F_{i,j,k}^{Send2}$ | The time required to compute the jth subtask of the ith mobile device on e |
| $u_{i,j}^{edge}$ | Total energy consumption of the jth subtask of the ith mobile device |
| $F_{i,j}$ | The time it takes for the jth subtask of the ith mobile device to complete the computation |
| $u_{i,j}^{g}$ | Completion time for all subtasks of the ith mobile device |
| $U_i$ | The energy consumption of all subtasks of the i-th mobile device to complete the calculation |

**Step 5** After the submitting node receives the block, it can verify the league, mainly to confirm whether the read-write data set in the transaction is consistent with the data version of the world state. The submitting node uses the read moulded part of the read-write set to check the transaction's validity and then writes the writing part of the read-write set of all verified transactions into the world state. At the same time, the submit node will use the write set to update the state database, that is, the ledger. In the event of validation failure, a validation block for aborted and committed transactions will be appended to the log. Then each transaction's commit or abort status will be recorded.

# MODELS

This chapter will first introduce the model of the system in this article. Then, the symbols used in this article and their meanings are shown in Table 1.

## System model

As shown in Fig. 1, the system is a mobile edge computing model with a three-tier architecture. The top layer of the system resides on an edge server with computing power, typically at a base station or remote access point. The middle layer of the system is relay devices, which are only responsible for task transmission but not task execution. The bottom layer of the system is the mobile device, each mobile device has a running application, and each application consists of multiple subtasks with task dependencies. The mobile device communicates with the relay device through a wireless connection, such as Wi-Fi. The relay device also communicates with the edge server through a wireless connection. There are multiple mobile devices in the model, represented by the set $N = \{n_1, n_2, \cdots, n_n\}$, where N is the number of mobile devices.

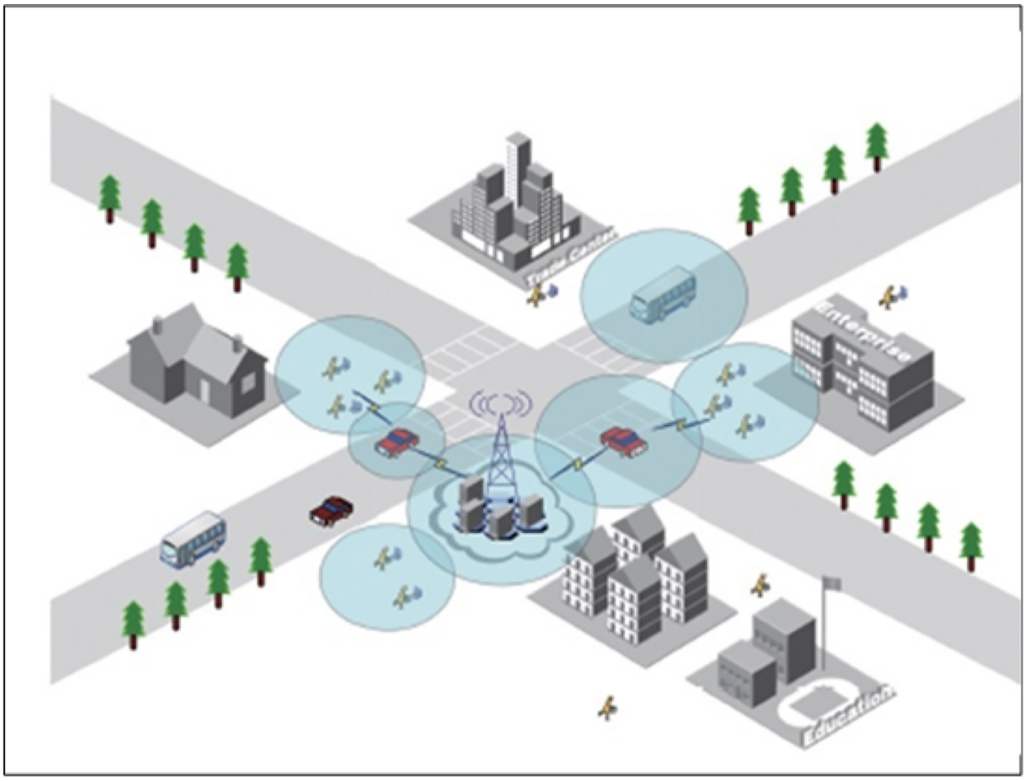


**Figure 1** System model. Full-size DOI: 10.7717/peerj-cs.1118/fig-1

Furthermore, the relays in the model are represented by the set G = {$g_1$, $g_2$, ⋯, $g_G$}, where G is the number of relays. For each mobile device, it contains M subtasks with dependencies, denoted by M = {$m_1$, $m_2$, ⋯, $m_M$}. In the following, this article's communication model and computing model will be introduced in detail.

## Wireless communication model

Wireless communication exists between mobile devices, relay devices, and edge servers. According to Shannon's formula, the uplink data rate from the ith mobile device to the kth relay device can be expressed as:

$$S_{i,k} = C_{i,k} .Oc \left[1 + \frac{Q_i H_{i,k}}{\sigma_k^2}\right] \tag{5}$$

Similarly, the uplink data rate between the kth relay device and the edge server e is:

$$S'_k = C'_k .Oc \left[1 + \frac{Q'_k H_{k,e}}{\sigma'^2}\right] \tag{6}$$

Among them, $C_{i,k}$ and $C'_k$ are the channel bandwidths between the ith mobile device and the kth relay device, and between the kth relay device and the edge server e, respectively; σk and σ′ represent the kth environmental noise when the relay device and edge server e are the receivers; $Q_i$ and $Q'_k$ are the transmission power of the ith mobile device and the

transmission power of the kth relay device, respectively; $H_{i,k}$ and $H_{k,e}$ are the channel gain between the i mobile device and the kth relay device and between the kth relay device and the edge server e is calculated as:

$$H_{a,b} = (e_{a,b})^{-\alpha} \tag{7}$$

where $e_{a,b}$ is the Euclidean distance between nodes a and b, α is the path loss component, and $a, b \in \{N \cup G \cup \{e\}\}$.

## Computational model

When j in i is executed locally on the mobile device, its execution time can be expressed as:

$$u_{i,j}^{local} = \frac{X_{i,j}}{d_i} \tag{8}$$

Amongst them, X (i, j) is the activity necessary to finish the $j^{th}$ subtask in the $i^{th}$ smart phone, $d_i$ is the $i^{th}$ mobile device's CPU clock cycle frequency, the unit is cycle/s, and the associated power consumption is:

$$F_{i,j}^{local} = \rho_i \cdot X_{i,j} \cdot d_i^2 \tag{9}$$

If the jth subtask in the ith smart phone is released to the kth relay device, the number of communication at this phase may be represented as: Where I is a constant that relies on the architecture of the mobile device, the number of communication at this point can be expressed as:

$$u_{i,j,k}^{send_1} = \frac{E_{i,j}}{S_{i,k}} \tag{10}$$

Among them, $E_{i,j}$ is the data size of the jth subtask of the $i^{th}$ mobile device. The energy consumption of this process can be expressed as:

$$E_{i,j,k}^{send_1} = q_i \cdot u_{i,j,k}^{send_1} \tag{11}$$

If the jth subtask in the ith mobile device is offloaded from the kth relay device to the edge server e, the transmission time of this process can be expressed as:

$$u_{i,j,k}^{send_2} = \frac{E_{i,j}}{S_k} \tag{12}$$

The corresponding energy consumption is:

$$E_{i,j,k}^{send_2} = q_i' \cdot u_{i,j,k}^{send_2} \tag{13}$$

The computation time to complete the jth subtask of the ith mobile device on the edge server e is:

$$u_{i,j}^{edge} = \frac{X_{i,j}}{d_e} \tag{14}$$

Among them, $d_e$ is the CPU clock cycle frequency of e, and the unit is cycle/s.

Therefore, the energy consumption for completing the jth subtask of the ith mobile device is:

$$E_{i,j} = E_{i,j,k}^{Local} \cdot \left(1 - \sum_{k=1}^{G} A_{i,j,k}\right) + \sum_{k=1}^{G} \left(F_{i,j,k}^{send_1} \cdot A_{i,j,k}\right) \tag{15}$$

where $\sum_{k=1}^{G} A_{i,j,k} \in \{0,1\}$, $A_{i,j,k}$ is the assignment strategy of the jth subtask of the ith mobile device. When $\sum_{k=1}^{G} A_{i,j,k} = 0$, it means that the jth subtask in the ith mobile device is offloaded from the ith mobile device to the kth relay device; when $\sum_{k=1}^{G} A_{i,j,k} = 1$, it means that the jth subtask of the ith mobile device performs computation locally.

## Task dependencies

This section defines dependencies between subtasks. The completion time of the subtask is divided into the following parts.

Let T ready$_1$ i,j be the time when the jth subtask of the ith mobile device is ready to be processed locally, then:

$$U_{i,j}^{ready_1} = max_{s \in \pi_J} \cdot \left\{ U\ G_{i,s}^{Local} \right\} \tag{16}$$

where $\pi_J$ is the set of predecessors to the jth subtask, s $\in \pi_J$. TF local, s is the local completion time of the sth subtask of the ith mobile device, which can be calculated as:

$$U\ G_{i,s}^{Local} = u_{i,j}^{edge} + U_{i,j}^{ready_1} \tag{17}$$

Similarly, $U\ G_{i,s}^{Local}$ is the completion time of the jth subtask of the ith mobile device on edge server e. It can be represented as:

$$U\ G_{i,j}^{Local} = u_{i,j}^{edge} + U_{i,j}^{ready_1} \tag{18}$$

where $U_{i,j}^{ready_1}$ j is the time when the j$^{th}$ subtask of the ith mobile device is ready to be processed on the edge server e, which can be calculated as:

$$U_{i,j}^{ready_2} = max_{s \in \pi_J} \cdot \left\{ u_{i,s}^{d} \right\} + u_{i,j,k}^{send_1} + u_{i,j,k}^{send_2} \tag{19}$$

Let $u_{i,s}^{d}$ be the completion time of the j$^{th}$ subtask of the ith mobile device, which can be calculated as:

$$u_{i,s}^{d} = U\ G_{i,j}^{Local} \cdot \left(1 - \sum_{k=1}^{G} B_{i,j}(k)\right) + U\ G_{i,j}^{edge} \cdot \left(1 - \sum_{k=1}^{G} B_{i,j}(k)\right) \tag{20}$$

Therefore, the time required for the i$^{th}$ mobile device to complete all subtasks is:

$$U_i = max_{j \in N} \left\{ u_{i,s}^{d} \right\} \tag{21}$$

To sum up, the total energy consumption of the i[th] mobile device can be expressed as:

$$F_i = \sum_{j=1}^{G} F_{i,j} \tag{22}$$

### Problem definition

This section will give a formulaic definition of minimizing the maximum energy consumption. The problem of reducing the maximum energy consumption is described as follows:

$$\text{Obj: } \min\{\max(Fi)\} \tag{23}$$

$$\text{U}i \leq U, \quad \forall i \in \text{N} \tag{24}$$

$$max_{j\in N}\left\{u_{i,s}^{d}\right\} \leq U_{i,j}^{ready_1}, \ \forall i \in \text{N}, \ \forall j \in \text{M} \tag{25}$$

$$max_{j\in N}\left\{u_{i,s}^{d}\right\} \leq U_{i,j}^{ready_2}, \ \forall i \in \text{N}, \ \forall j \in \text{M} \tag{26}$$

$$\sum_{k=1}^{G} A_{i,j,k} \cdot \left(1 - \sum_{k=1}^{G} A_{i,j,k}\right) = 0 \ , \ \forall i \in N, \forall j \in \text{M}, \forall k \in \text{G} \tag{27}$$

where U is the budget time for the given ith mobile device to complete all subtasks Eq. (24) is the constraint on the completion time of the ith mobile device. Equations (25) and (26) are the dependency constraints of subtasks, ensuring that the jth subtask of the ith mobile device can only start executing after all its predecessor subtasks are completed. Equation (27) indicates that the jth subtask of the ith mobile device can only be committed locally or at the edge server, e.

According to the existing research, the problem of minimizing the maximum energy consumption is an optimization problem of finding a scheduling strategy, which can be reduced to the minimum, full-time problem in the literature, which reduces the minimum, full-time pain to NP-complex integer programming problem. Therefore, the issue of minimizing the maximum energy consumption is NP-hard.

## PERFORMANCE EVALUATION

In this chapter, experiments are designed to investigate the performance of the proposed algorithm. This article uses Matlab r 2016a to conduct many experiments and get the experimental results. This article sets the coverage as 50 m × 50 m (*Hou et al., 2020*). By default, the number of subtasks, mobile devices, and relay devices is set to 4, 3, and 3, respectively, and other parameters in the experiments (*Jiang, Li & Wu, 2019*; *Asheralieva & Niyato, 2021*; *Wu et al., 2021*) are shown in Table 2. In the whole experiment process, the article assumes that the data size and workload size of computing tasks follow a normal distribution in the range of (25 Kbit, 1,024 Kbit), with a mean of 512 and a standard deviation of 256. In addition, this article also designs an experimental comparison algorithm for the total energy consumption minimization algorithm (TECM).

**Table 2** Experimental parameters.

| Parameter | Numerical value | Parameter | Numerical value |
|---|---|---|---|
| alpha | 0.1 | Bk′/MHz | 5 |
| Bi,k/MHz | 5 | pk′/dBm | 5 |
| pi/dBm | 5 | fe/GHz | 4 |
| fi/GHz | 0.8 | σ′/dBm | −70 |
| ρi | $0.3 \times 10^{-27}$ | σk/dBm | −70 |

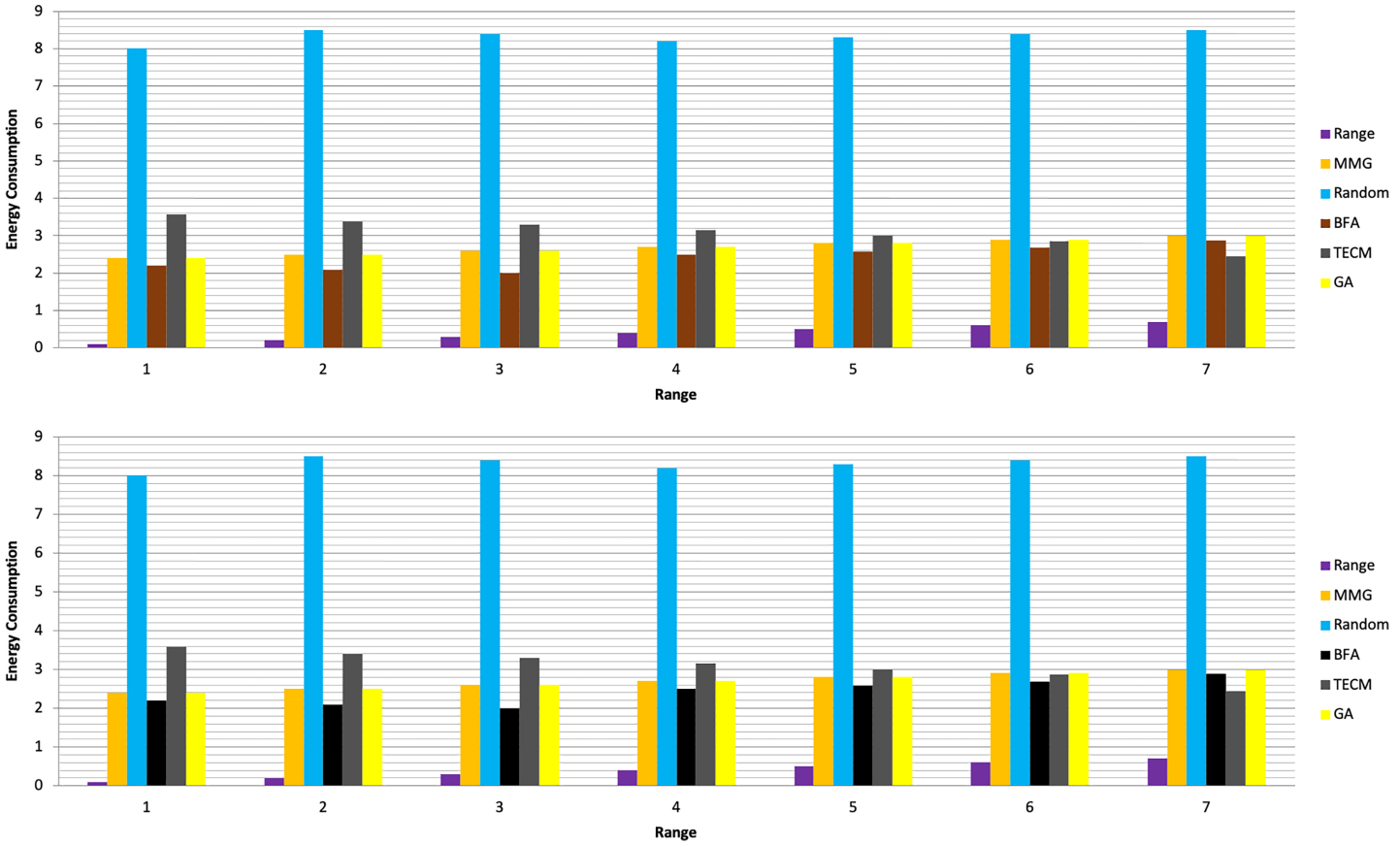


**Figure 2 Maximum energy consumption.** (A) Transmission Power = 5 dbm. (B) Transmission Power = 6 dbm.
Full-size DOI: 10.7717/peerj-cs.1118/fig-2

## Experiment 1

In Experiment 1, the article studies the change of the relationship between the path loss component and the maximum energy consumption of the subtask. Figure 2 depicts that the maximum energy consumption of the subtask increases with the path loss component. From Eqs. (5)–(7), (10), (12), the maximum energy consumption of subtasks is proportional to the path loss component, which explains the shape of the curves in the experimental Fig. 2 with Table 3. Experiments show that when the transmission power of the mobile device is five dBm and six dBm, the performance of the MMG algorithm in

**Table 3** Evaluation of experiment 1.

| Range | MMG | Random | BFA | TECM | GA | MMG | Random | Brute force | TECM | GA |
|---|---|---|---|---|---|---|---|---|---|---|
| | Transmission power = 5 dBm | | | | | Transmission power = 6 dBm | | | | |
| 0.1 | 2.4 | 8 | 2.2 | 3.58 | 2.4 | 3.4 | 8.5 | 2.5 | 4.58 | 3.4 |
| 0.2 | 2.5 | 8.5 | 2.1 | 3.39 | 2.5 | 3.5 | 7.5 | 2.67 | 4.39 | 3.5 |
| 0.3 | 2.6 | 8.4 | 2 | 3.29 | 2.6 | 3.6 | 7.3 | 2.78 | 4.29 | 3.6 |
| 0.4 | 2.7 | 8.2 | 2.5 | 3.15 | 2.7 | 3.7 | 7.9 | 2.2 | 4.15 | 3.7 |
| 0.5 | 2.8 | 8.3 | 2.58 | 3 | 2.8 | 3.8 | 8.6 | 2.296 | 4 | 3.8 |
| 0.6 | 2.9 | 8.4 | 2.69 | 2.86 | 2.9 | 3.9 | 8.4 | 2.45 | 3.86 | 3.9 |
| 0.7 | 3 | 8.5 | 2.88 | 2.45 | 3 | 4 | 8 | 2.78 | 3.45 | 4 |

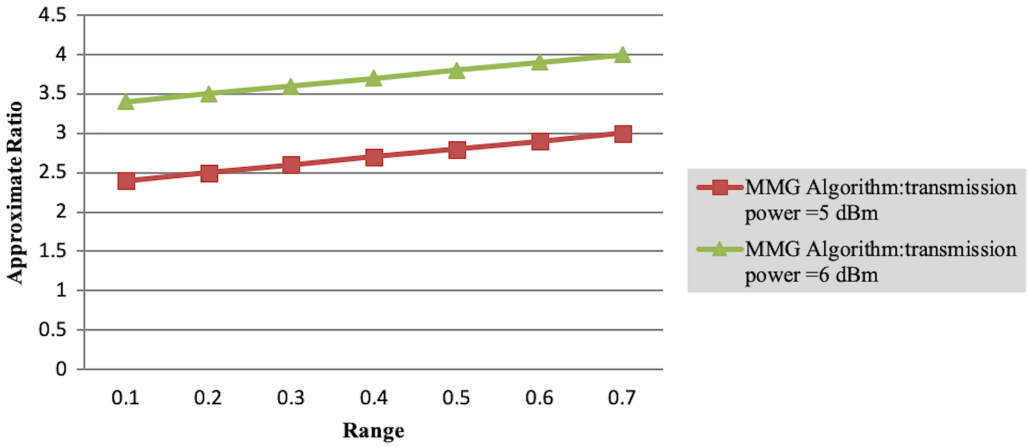


**Figure 3** Approximation ratio of greedy algorithm. Full-size DOI: 10.7717/peerj-cs.1118/fig-3

minimizing the maximum energy consumption of subtasks is improved by 66.59% and 61.87% on average compared with the random algorithm, which is close to the maximum obtained by the brute force algorithm (BFA) optimal solution. Figure 3 is based on the results of Figs. 2A and 2B, which show that the approximate ratios of the MMG algorithm are 1.096 9 and 1.098 4 when the minimum transmit power of the mobile device is five dBm and six dBm, respectively.

## Experiment 2

Experiment 2 investigates the maximum energy consumption of subtasks under changing time constraints. In Figs. 3 and 4 the minimum transmits the power of the mobile device is five dBm. The results show that the greedy algorithm outperforms the total energy consumption optimization and random algorithms.

Finally, in the case where the path loss component is 0.1, we observe the variation of the maximum energy consumption of subtasks with the number of task layers. As shown in Table 4, as the number of tasks increases, the completion time of each subtask decreases accordingly, resulting in jobs that need to be executed in a shorter time. This leads to a rise in the energy consumption required to complete the task. However, the MMG algorithm is

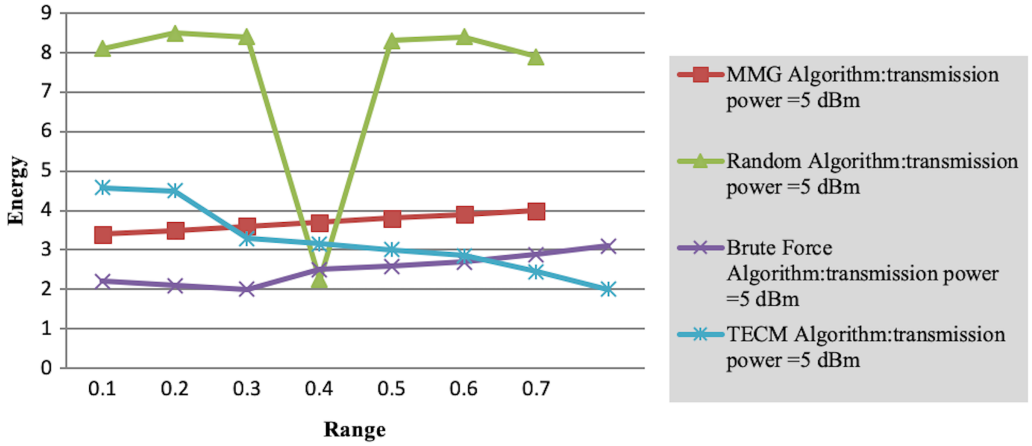


**Figure 4** **Maximum energy consumption of subtask *vs* time constraint.**
Full-size DOI: 10.7717/peerj-cs.1118/fig-4

**Table 4** **Maximum energy consumption of subtask V.S *time* constraint (Transmission power = 5 Dbm).**

| Range | MMG algorithm | Random algorithm | Brute force algorithm | TECM algorithm | Time constraint |
|---|---|---|---|---|---|
| 0.1 | 3.4 | 8.1 | 2.2 | 4.58 | 2.4 |
| 0.2 | 3.5 | 8.5 | 2.1 | 4.49 | 2.5 |
| 0.3 | 3.6 | 8.4 | 2 | 3.29 | 2.6 |
| 0.4 | 3.7 | 2.25 | 2.5 | 3.15 | 2.7 |
| 0.5 | 3.8 | 8.3 | 2.58 | 3 | 2.8 |
| 0.6 | 3.9 | 8.4 | 2.69 | 2.86 | 2.9 |
| 0.7 | 4 | 7.9 | 2.88 | 2.45 | 3 |

better than the TECM algorithm and the random algorithm in the maximum energy consumption of the minimum subtask. As shown in Table 4, as the number of subtasks increases, the leading energy consumption of subtasks increases, for the transmission power of 5 dBm, the approximate ratio of MG is 1.353 7, 1.353 8.

## Experiment 3

Accomplished inside the confines of the room As a result, the amount of energy needed to execute the work increases. However, in terms of maximum energy consumption of the smallest subtask, the MMG method outperforms both the TECM and the random algorithms. As shown in Table 5 and Figure 5, the leading power consumption of subtasks rises as the quantity of subtasks grows, and when the portable device's signal strength is five dBm and six dBm, respectively, the approximate MMG ratios are 1.353 7, 1.353 8.

This article provides a detailed description of the proposed MMG algorithm. This article divides U into several parts according to the number of subtasks, and each subtask of each layer should end in the interval after the division is completed. First, each subtask's computation model is determined based on time and energy consumption conditions.

**Table 5** Maximum energy consumption of subtask V.S *the nuber of subtask.*

| Range | MMG | Random | Brute force | TECM | Subtask | MMG | Random | Brute force | TECM | Subtask |
|---|---|---|---|---|---|---|---|---|---|---|
| | Transmission power = 5 dBm | | | | | Transmission power = 6 dBm | | | | |
| 0.1 | 3.4 | 9 | 3.2 | 3.58 | 3.4 | 3.4 | 9.5 | 3.5 | 5.58 | 3.4 |
| 0.2 | 3.5 | 9.5 | 3.1 | 3.39 | 3.5 | 3.5 | 8.5 | 3.67 | 5.39 | 3.5 |
| 0.3 | 3.6 | 9.4 | 3 | 3.29 | 3.6 | 3.6 | 9.3 | 3.78 | 5.29 | 3.6 |
| 0.4 | 3.7 | 9.2 | 3.5 | 3.15 | 3.7 | 3.7 | 9.9 | 3.2 | 5.15 | 3.7 |
| 0.5 | 3.8 | 9.3 | 3.58 | 3 | 3.8 | 3.8 | 9.6 | 3.9 | 5 | 3.8 |
| 0.6 | 3.9 | 9.4 | 3.56 | 2.86 | 3.9 | 3.9 | 9.4 | 3.45 | 5.86 | 3.9 |
| 0.7 | 4 | 9.5 | 3.88 | 2.45 | 4 | 4 | 9 | 3.78 | 4.56 | 4 |

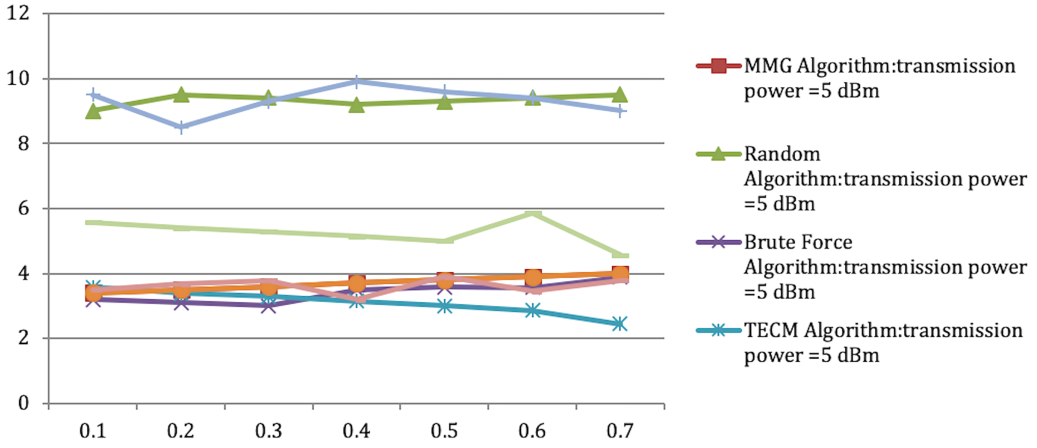


**Figure 5** Approximation ratio of MMG algorithm. Full-size DOI: 10.7717/peerj-cs.1118/fig-5

Then, for each subtask, an allocation scheme is obtained by a min-max algorithm. In addition, it is then judged whether the allocation scheme satisfies the time limit.

## VALIDITY OF ALGORITHM DESIGN AND RESULT ANALYSIS

Based on the above analysis, the energy consumption of different subtasks calculated locally and migrated to the edge server is obtained through calculation, and an energy consumption matrix is formed. Then, a relay device node is randomly assigned for each mobile device source node. Finally, assuming $F_{max}$ is the maximum energy consumption from node a to relay b, find $F_{max}$, set all relays to "unused", and set to "used" when the relay device is occupied.

Get allocation policy. It is judged whether there are other relay device nodes available except the relay device nodes allocated by the maximum energy consumption. If available, replace the relay device node and determine the new $F_{max}$ (adjust the energy consumption so that the total energy consumption obtained by the allocation strategy is as small as possible, and the energy consumption difference is more minor). If no other available relay device node is found, it starts to judge whether the allocation strategy satisfies the time constraint. If not, the process is re-allocated; the feasible allocation strategy is obtained if it is met.

**Theorem 1:** Set the number of mobile devices to be M, the number of subtasks of each mobile device to be N, and the number of relay devices to be G. The total time complexity of the algorithm is $P(NMG + NMG^2)$.

Proof First, the algorithm proposed in this article is divided into three levels of nested loops to calculate the time consumption and energy consumption, and its time complexity is P(NMG). Then, the time complexity of the min-max function is analyzed. The complexity of each iteration is P(G), and there are at most NG pairs per iteration, so the total min-max time complexity is $P(NMG^2)$. In terms of time checking, this is a sum function with only one level of loops, including two groups of nested. While loops, each while loop does not exceed H times, so its time complexity is $P(NMG^2)$. Also, the second step is executed under different subtask layers, so the total time complexity of the second step is O(MNH2). The whole time complexity of this algorithm is $P(NMG + NMG^2)$.

**Theorem 2:** The bipartite graph of any mobile edge computing structure can be transformed into an energy consumption matrix. Denote ε and N as an arbitrarily small positive constant and the number of subtasks, respectively, the approximate ratio of this greedy algorithm can reach $(1 + \varepsilon)^N$.

Prove In Theorem 1, the algorithm's time complexity is $P(NMG + NMG^2)$, and the maximum number of iterations is G. For a subtask, according to Formulas (11) and (13), let its transmission energy ratio be σ and ψ, respectively; according to Formula (9), let its calculated energy ratio be γ. The energy consumption of each "device-relay" pair can be expressed as $F_{i,j} = \min(\sigma_{i,j} + \gamma_j, \psi_{i,j})$. The maximum and effective minimum energy consumption of "source-relay" is set to $F_{max}$ and $F_{max}$, respectively. Then there are:

$$F_{max} \leq \min(\sigma_{i,j} + \gamma_j, \psi_{i,j}) \cdot \widetilde{\omega}_{max}$$
$$F_{max} \geq \min(\sigma_{i,j} + \gamma_j, \psi_{i,j}) \cdot \widetilde{\omega}_{min}$$

Therefore, the upper bound on the energy consumption of each "device-relay" pair is $F_{max}$. Let $\lambda = \varepsilon F_{max}/\mu$, where λ is the step size of the average adjustment of the maximum energy consumption for each "device-relay", and μ is the maximum number of iterations, then $F_{max} = \lambda\mu/\varepsilon$. Let $F_{op}(D^*)$ be minimized for each "device-relay."

The optimal solution for the maximum energy consumption of $D^*$ corresponds to the optimal policy. Let C′ be the maximum energy consumption of each "device-relay" pair in the greedy algorithm, then $F_{op}(D^*) \leq F(D')$. And $\lambda = \varepsilon F_{max}/\mu$, then there is $\lambda\mu = \varepsilon F_{max}$. Therefore, λμ is one of the upper bounds on the energy consumption adjustment of mobile devices. Accordingly, there are:

$$Fop(D^*) \leq F(D') \leq Fop(D^*) + \lambda\mu \tag{28}$$

According to Eq. (28) and $0 < F_{min} < Fop(D^*)$, we have:

$$F(D') \leq Fop(D^*) + \lambda\mu = Fop(D^*) + \varepsilon Fmax$$
$$= Fop(D^*) + \frac{\varepsilon}{\varepsilon' F_{max}} \leq Fop(D^*) + \frac{\varepsilon}{\varepsilon' Fop(D^*)}$$

Let $\varepsilon'' = \varepsilon/\varepsilon'$, where both $\varepsilon$ and $\varepsilon'$ are positive constants, and $\varepsilon'$ is approximately equal to $\tilde{\omega}_{min}/\tilde{\omega}_{max}$, then $F(D') \leq (1 + \varepsilon'')Fop(D^*)$.

## CONCLUSION

This work investigates the NP-hard issue of energy balancing in multitasking with several mobile devices. The cloud server that consumes more transmission energy is deleted from the original design. The task's relay device node is added as the task's transfer node, resulting in three-layer architecture of smart phones, relay devices, and edge servers. Simultaneously, this study develops the MMG algorithm to address the energy consumption balancing issue and establishes its efficacy *via* a large number of comparison tests.

## ADDITIONAL INFORMATION AND DECLARATIONS


### Funding

The authors received no funding for this work.


### Competing Interests

The authors declare that they have no competing interests.

### Author Contributions

- Jianbin Wu conceived and designed the experiments, prepared figures and/or tables, and approved the final draft.
- Sami Ahmed Haider performed the experiments, analyzed the data, prepared figures and/or tables, authored or reviewed drafts of the article, and approved the final draft.
- Mukesh Soni conceived and designed the experiments, analyzed the data, performed the computation work, prepared figures and/or tables, authored or reviewed drafts of the article, and approved the final draft.
- Ashima Kalra analyzed the data, performed the computation work, authored or reviewed drafts of the article, and approved the final draft.
- Nabamita Deb performed the experiments, performed the computation work, prepared figures and/or tables, authored or reviewed drafts of the article, and approved the final draft.

### Data Availability

The following information was supplied regarding data availability:

The raw data and code are available in the Supplemental Files.

### Supplemental Information

Supplemental information for this article can be found online at http://dx.doi.org/10.7717/peerj-cs.1118#supplemental-information.

## REFERENCES


**Abbas N, Zhang Y, Taherkordi A, Skeie T. 2018.** Mobile edge computing: a survey. *IEEE Internet of Things Journal* **5(1)**:450–465 DOI 10.1109/JIOT.2017.2750180.

**Asheralieva A, Niyato D. 2021.** Learning-based mobile edge computing resource management to support public blockchain networks. *IEEE Transactions on Mobile Computing* **20(3)**:1092–1109 DOI 10.1109/TMC.2019.2959772.

**Badri H, Bahreini T, Grosu D, Yang K. 2018.** Risk-based optimization of resource provisioning in mobile edge computing. In: *2018 IEEE/ACM Symposium on Edge Computing (SEC)*. Piscataway: IEEE, 328–330.

**De Nitto Personè V, Grassi V. 2019.** Architectural issues for self-adaptive service migration management in mobile edge computing scenarios. In: *2019 IEEE International Conference on Edge Computing (EDGE)*. Piscataway: IEEE, 27–29.

**Dolui K, Datta SK. 2017.** Comparison of edge computing implementations: fog computing, cloudlet and mobile edge computing. In: *2017 Global Internet of Things Summit (GIoTS)*. 1–6.

**Guo S, Dai Y, Guo S, Qiu X, Qi F. 2020.** Blockchain meets edge computing: stackelberg game and double auction based task offloading for mobile blockchain. *IEEE Transactions on Vehicular Technology* **69(5)**:5549–5561 DOI 10.1109/TVT.2020.2982000.

**Hou Y, Liu W, Lin H, Wang X. 2020.** Multi-layer access control mechanism based on blockchain for mobile edge computing. In: *2020 IEEE International Conference on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom)*. Piscataway: IEEE, 285–291.

**Huang Y, Zhang J, Duan J, Xiao B, Ye F, Yang Y. 2022.** Resource allocation and consensus of blockchains in pervasive edge computing environments. *IEEE Transactions on Mobile Computing* **21(9)**:3298–3311 DOI 10.1109/TMC.2021.3053230.

**Hung C-H, Hsieh Y-C, Wang L-C. 2017.** Control plane latency reduction for service chaining in mobile edge computing system. In: *2017 13th International Conference on Network and Service Management (CNSM)*. 1–5.

**Jiang S, Li X, Wu J. 2019.** Hierarchical edge-cloud computing for mobile blockchain mining game. In: *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*. Piscataway: IEEE, 1327–1336.

**Kim K, Hong CS. 2019.** Optimal task-UAV-edge matching for computation offloading in UAV assisted mobile edge computing. In: *2019 20th Asia-Pacific Network Operations and Management Symposium (APNOMS)*. 1–4.

**Kim N, Lee Y, Lee C, Nguyen TV, DatTuong V, Cho S. 2020.** GPU-specific task offloading in the mobile edge computing network. In: *2020 International Conference on Information and Communication Technology Convergence (ICTC)*. 1874–1876.

**Li X, Ding R, Liu X, Yan W, Xu J, Gao H, Zheng X. 2018.** COMEC: computation offloading for video-based heart rate detection APP in mobile edge computing. In: *2018 IEEE International Conference on Parallel & Distributed Processing with Applications, Ubiquitous Computing & Communications, Big Data & Cloud Computing, Social Computing & Networking, Sustainable Computing & Communications (ISPA/IUCC/BDCloud/SocialCom/SustainCom)*. Piscataway: IEEE, 1038–1039.

**Li Q, Ma X, Zhou A, Luo X, Yang F, Wang S. 2021.** User-oriented edge node grouping in mobile edge computing. In: *IEEE Transactions on Mobile Computing*. Piscataway: IEEE.

**Li H, Shou G, Hu Y, Guo Z. 2016.** Mobile edge computing: progress and challenges. In: *2016 4th IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud)*. Piscataway: IEEE, 83–84.

**Li Y, Wang S. 2018.** An energy-aware edge server placement algorithm in mobile edge computing. In: *2018 IEEE International Conference on Edge Computing (EDGE)*. 66–73.

**Li J, Wu J, Chen L, Li J, Lam SK. 2021.** Blockchain-based secure key management for mobile edge computing. In: *IEEE Transactions on Mobile Computing*.

**Marjanović M, Antonić A, Žarko IP. 2018.** Edge computing architecture for mobile crowdsensing. *IEEE Access* **6**:10662–10674 DOI 10.1109/ACCESS.2018.2799707.

**Muniswamaiah M, Agerwala T, Tappert CC. 2021.** A survey on cloudlets, mobile edge, and fog computing. In: *2021 8th IEEE International Conference on Cyber Security and Cloud Computing (CSCloud)/2021 7th IEEE International Conference on Edge Computing and Scalable Cloud (EdgeCom)*. Piscataway: IEEE, 139–142.

**Queralta JP, Qingqing L, Zou Z, Westerlund T. 2020.** Enhancing autonomy with blockchain and multi-access edge computing in distributed robotic systems. In: *2020 Fifth International Conference on Fog and Mobile Edge Computing (FMEC)*. 180–187.

**Reiter A, Prünster B, Zefferer T. 2017.** Hybrid mobile edge computing: unleashing the full potential of edge computing in mobile device use cases. In: *2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*. 935–944.

**Ren P, Qiao X, Chen J, Dustdar S. 2018.** Mobile edge computing – a booster for the practical provisioning approach of web-based augmented reality. In: *2018 IEEE/ACM Symposium on Edge Computing (SEC)*. Piscataway: IEEE, 349–350.

**Samanta A, Li Y. 2018.** Latency-oblivious incentive service offloading in mobile edge computing. In: *2018 IEEE/ACM Symposium on Edge Computing (SEC)*. Piscataway: IEEE, 351–353.

**Subramanya T, Goratti L, Khan SN, Kafetzakis E, Giannoulakis I, Riggio R. 2017.** SDEC: a platform for software defined mobile edge computing research and experimentation. In: *2017 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*. Piscataway: IEEE, 1–2.

**Takahashi N, Tanaka H, Kawamura R. 2015.** Analysis of process assignment in multi-tier mobile cloud computing and application to edge accelerated web browsing. In: *2015 3rd IEEE International Conference on Mobile Cloud Computing, Services, and Engineering*. Piscataway: IEEE, 233–234.

**Wei X, Wang S, Zhou A, Xu J, Su S, Kumar S, Yang F. 2017.** MVR: an architecture for computation offloading in mobile edge computing. In: *2017 IEEE International Conference on Edge Computing (EDGE)*. Piscataway: IEEE, 232–235.

**Wu H, Wolter K, Jiao P, Deng Y, Zhao Y, Xu M. 2021.** EEDTO: an energy-efficient dynamic task offloading algorithm for blockchain-enabled IoT-edge-cloud orchestrated computing. *IEEE Internet of Things Journal* **8(4)**:2163–2176 DOI 10.1109/JIOT.2020.3033521.

**Xu J, Wang S, Zhou A, Yang F. 2020.** Edgence: a blockchain-enabled edge-computing platform for intelligent IoT-based dApps. *China Communications* **17(4)**:78–87 DOI 10.23919/JCC.2020.04.008.

**Zeng J, Sun J, Wu B, Su X. 2020.** Mobile edge communications, computing, and caching (MEC3) technology in the maritime communication network. *China Communications* **17(5)**:223–234 DOI 10.23919/JCC.2020.05.017.

**Zhao J, Yang H. 2021.** Social-aware cross-chain authentication strategy in mobile edge computing. In: *2021 IEEE 9th International Conference on Information, Communication and Networks (ICICN)*. Piscataway: IEEE, 170–174.