Time-and-Energy Consumption Offloading for Mobile Devices in Mobile Cloud Computing

Anh-Tu Bui*, Van-Viet Nguyen, and Ninh-Thuan Truong

***Abstract*—Concurrent processing of sophisticated tasks on mobile devices could consume a lot of energy and processing time because of their limited resources. In order to offload mobile devices, some tasks are uploaded to the cloud server for execution. However, it is very difficult to choose which tasks to upload to the cloud because it needs to ensure two requirements: Optimizing energy costs and optimizing execution time costs. In this paper, we introduce a method to offload mobile devices when it processes multiple tasks concurrently. By applying the proposed energy automata from previous studies, our method allows for the identification of factors influencing the energy consumption and execution time of tasks, while also proposing an objective function and algorithms to make the offloading decision. When we applied the proposed method to an actual image processing application to process 1000 photos on a mobile device, it could save a maximum of 24.1% of energy, 37.6% of processing time, and an average of 18% of energy, 21% of processing time compared to non-offloading.**

***Index Terms*—Android, computation offloading, mobile cloud computing, power model**

I. Introduction

In recent years, mobile devices have developed briskly. Nowadays, mobile devices are used for many different purposes instead of making calls as before. The computing needs of mobile devices increase rapidly due to processing a lot of complex tasks. However, all mobile devices have certain limitations in terms of processing ability and energy. In order to offload on mobile devices, researchers have applied computational models such as Cloud Computing (CC) or Mobile Cloud Computing (MCC). In MCC, tasks on the mobile device can be uploaded to the cloud server for processing to optimize performance for the mobile device [1–4]. Recent research has proven that offloading to the cloud has enhanced processing ability and energy savings for mobile devices [5–7].

When the mobile device processes a task, the mobile device consumes an amount of energy and time to process. On the contrary, if the task is uploaded to the cloud server to process, the mobile device will consume time and energy to send and receive data to/from the cloud server. In the condition of a good network, high data transfer rate, uploading the task to the cloud server will be convenient and consume less time. However, in the condition of a slow network, mobile devices will consume a lot of time and energy. Therefore, making offloading decisions (uploading a task to a cloud server for processing) is the most important problem of offloading methods for mobile devices.

Two problems that researchers need to solve when offloading to optimize energy consumption and execution time are:

Firstly, making an offloading decision depends on the results of the calculation of the mobile device's energy consumption. In order to calculate the energy consumption of mobile devices while executing tasks, researchers use two main methods: estimation and measurement. In the problem of making the offloading decision, the energy consumption must be calculated before making the decision. Therefore, measurement methods using specialized measuring devices cannot be used in this case. The main method used by researchers is the method of estimation based on mathematical models. Most of the published research does not describe clearly the method of estimation, but the estimation error can affect the offloading decision.

Secondly, making an offloading decision must simultaneously ensure two main issues: minimizing energy cost and execution time. Some studies focus on optimizing energy costs when making offloading decisions [8–12]. However, it is a fact that process execution time may take longer. Some other studies focus on minimizing the execution time of tasks [13–16], that do not analyze the energy costs when offloading to shorten the execution time. Hence, the result is that mobile devices experience dropping energy rapidly. Some studies such as [17–20] have provided solutions to optimize both time and energy costs, but the possibility of doing this is very little.

In order to solve those two problems, in this paper, our approach propose an approach to make the offloading decision for the mobile device when processing some independent tasks. To accurately estimate the energy consumption, the study uses the power automata, which were presented in the previous study [21]. This power automata allows estimating the energy consumption in different use cases based on the analysis of program source code. Next, we calculate the factors that affect the energy cost and execution time in two cases: Execution on the mobile device and execution on the Cloud Server Execution to build a multiplicative function, which aims to determine which is the most optimal execution case.

The layout of this paper is organized as follows: Section II introduces the related works to offloading calculation and our previous research on the method of estimating energy consumption. Section III presents our approach to offloading for mobile devices, which aims to optimize simultaneously both energy consumption and task execution time. Section IV analyzes the factors that affect energy consumption and

Manuscript January 15, 2023; revised April 13, 2023; accepted April 10, 2023.

Anh-Tu Bui and Van-Viet Nguyen are with University of Engineering and Technology, Hanoi, Vietnam and Thai Nguyen University of Information and Communication Technology, Thai Nguyen, Vietnam.

Ninh-Thuan Truong is with University of Engineering and Technology, Hanoi, Vietnam.

*Correspondence: batu@ictu.edu.vn (A.T.B.)

execution time. Section V describes the formula for calculating energy and time costs based on the factors, which was identified in Section IV. In Section VI, we build the objective function and introduce the algorithm to make the offloading decision, which aims to optimize multi-objectives. Subsequently, the study applies the proposed method to specific cases, and the most optimal result is presented in Section VII. Section VIII concludes the work done and gives oriented development.

II. Related Work

In this section, the paper presents our previous research on the method for estimating energy consumption for mobile devices based on analyzing program source code. Subsequently, the paper presents some related works to offloading calculations and the problems to be solved.

A. Estimating Energy Consumption for the Mobile Device

As shown in research [21], the first thing we have to do to estimate the energy consumption of the mobile device is to analyze the operating states of each hardware. For example, the operating state of the audio transmitter under the control of the software source code is presented in Fig. 1.

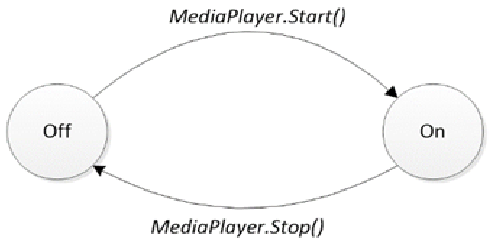

Fig. 1. Audio automata.

Therefore, we use Energy Automata to model the energy consumption across the hardware. Eg:

$$A_{Audio} = (Q_{Audio}, \in_{Audio}, \delta_{Audio}, q_{0Audio}, F_{Audio})$$

Subsequently, we merge and optimize the power automata of the hardware to become the power automata for mobile devices:

$$A_{Audio} = (Q_{Audio}, \in_{Audio}, \delta_{Audio}, q_{0Audio}, F_{Audio})$$

After determining the energy states of the device, we determine the energy consumption coefficient and the lifetime in each state, thereby estimating the total energy consumption for each specific use-case.

In summary, our previous research helps us estimate energy consumption for use cases by analyzing the program's source code.

B. Offloading for the Mobile Device

There are many studies on offloading in mobile cloud computing, such as offloading data, offloading energy consumption, or offloading to optimize performance. Some studies focus on saving energy [8, 22, 23], while others focus on reducing the task execution time of the mobile device. There are very few current studies that can simultaneously optimize both energy consumption and execution time. Most of them only focus on an optimization goal.

1) *Saving Energy:* Some studies such as [10–12] provide the approach by determining the parts that will be reduced in the program. They calculate the energy consumption for each function in the program that aims to construct a graph that includes energy consumption paths and then use algorithms to choose the least energy consumption path. In addition, the studies [24–29] use energy models to calculate energy consumption, thereby making offloading decisions for energy-intensive tasks. However, those studies overlooked some important factors that affect the offloading decision such as Network speed, the processing speed of mobile devices, or data traffic to send and receive to/from the cloud server.

2) *Saving execution time:* In order to reduce the execution time, some studies [14–16] upload data or a part of a program to the cloud server for execution. Researchers identify parts of the program that are likely to take a large amount of processing time and then upload them to the cloud server to save time. These studies test for small software, and may not work for more complex software.

3) *Simultaneous saving of energy and time*: Both energy and execution time are important requirements in the offloading process. However, there are few studies that can simultaneously optimize both of these objectives [25, 26]. The studies [17, 18] provide the partial offloading solution, and researchers also consider simultaneously both of two factors: time and energy. However, both of these studies suggest that energy consumption is proportional to execution time, which is not exactly in some cases. Moreover, these studies have not provided a method to estimate energy consumption before making the offloading decision.

In this paper, we use the results of previous research as a premise for accurately estimating the energy consumption of the mobile device when executing tasks. Based on the analysis of influencing factors, we construct an objective function that ensures to optimize both energy and time. Our proposed method also solves the two problems mentioned in Section I.

III. Methodology

In this section, we describe the overview of the offloading model for mobile devices when processing n independent tasks. These tasks do not depend on each other and can be executed independently. In order to make it easier to distinguish which tasks are executed on the mobile device and which tasks are executed on the cloud server, we assign each task a label with a value of 0 or 1. In which, 0 represents the execution on the mobile device, and 1 represents the execution on the cloud server.

Hence, there are many cases that can be performed to execute the tasks. First of all, we need to define an execution case.

Definition 1: An execution case of tasks (referred to as an execution case) is the process of executing all tasks. In which, each task is executed on the mobile device (labeled as 0) or is offloaded by executing on a cloud server (labeled as 1). An execution case of tasks is represented by a sequence of "0, and 1".

Consequently, the problem of offloading (also known as finding the optimal execution case) will become the problem of finding a sequence of n numbers "0, and 1" such that the total energy and time cost when executing according to that sequence is the most optimal.

In order to determine the total energy cost and execution time for all tasks, we need to determine the energy and time costs for each task. The energy and time cost of a task depends on many factors such as: CPU speed, sending/receiving data speed, data size, and processing complexity of each task. After determining the total energy cost and execution time for all tasks in different cases, we choose the most optimal execution case.

Fig. 2 depicts our approach to offloading on mobile devices:

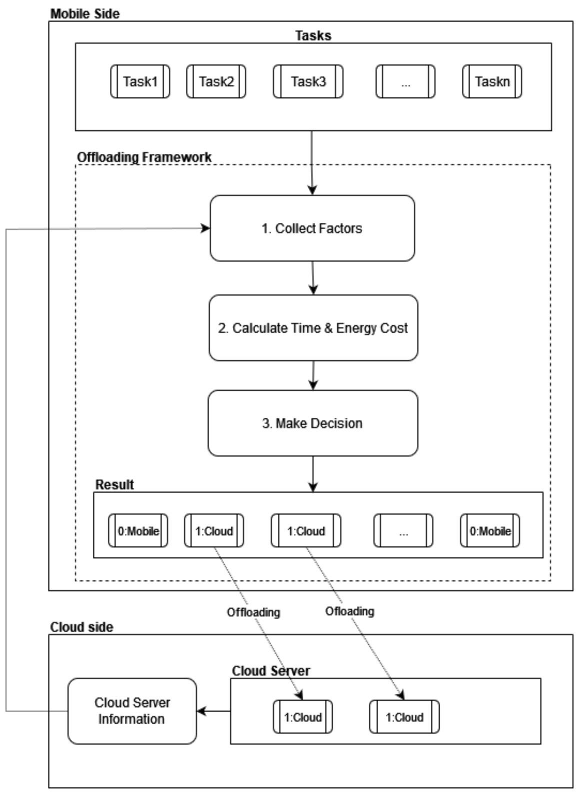

Fig. 2. Offloading model for mobile devices.

Tasks: Tasks are a set of *n* (n is a positive integer) independent tasks that need to be processed on a mobile device. The source code of the tasks is used to analyze the information, which aims to build the energy-state model, estimate the energy consumption in the cases and calculate the time to send and receive data, and execution time.

Collect Factors: The process of determining the values of the factors that affect energy consumption and execution time will be done at this step. The analyzer system will aggregate information from Cloud Server Information, Mobile Devices, and Tasks.

Calculate Time & Energy Cost: At this step, the calculator system will base on the values of the factors determined in the previous step to calculate the execution time and energy consumption for each task in both cases: Execution on the mobile device and execution on the cloud server.

Make Decision: Based on the parameters of execution time and energy consumption for each task in both cases (execution on the mobile and execution on the cloud server), the offloading algorithms will base on the objective function to decide which task to offload.

Result: The result of offloading for each task will be represented by labeling each task, specifically:

- Tasks that are offloaded will be labeled with the number 1 and then will be uploaded to the cloud for processing.
- Tasks that are executed on the mobile device will be labeled with the number 0.

Offloading: For offloaded tasks, all input data will be uploaded to the cloud server for processing, and the output results will be downloaded to the mobile device after the server completes the processing.

IV. Collect Factors

In order to calculate the energy cost and execution time for all tasks, we determine the factors that affect the energy consumption and execution time of each task. And then, we will calculate the energy cost and execution time for each task. The process of determining factors and calculating their values is executed in two cases: Execution on the mobile device and execution on the cloud server.

A. Execution on the Mobile Device

1) Determining factors affecting energy consumption

When executing tasks on a mobile device, its energy consumption will be calculated according to the proposed model of energy state in the research [21]. The factors that affect energy consumption will be the energy states of the application and the lifetime of these states. In other words, energy consumption will depend on the source code that will execute in specific use-case (UC).

2) Determining factors affecting execution time

When tasks execute on a mobile device, the microprocessor will operate and execute the code in the program. Processing time will depend on the these factors:

- **CPU speed:** The CPU will process the commands in the source code to execute the task. The processing time for the same task will be different depending on the CPU's speed on each device. Normally, in the MCC model, the processing speed of the CPU on the mobile device is much smaller than the processing speed of the cloud, which is an important factor affecting the processing time.
- **The number of instructions to be processed:** In a UC, we can calculate **exactly** the number of instructions that the CPU will have to execute. We can calculate the total task execution times based on the processing speed.
- **The data size to be processed:** The size of the data also affects **the** task processing times. However, if the data increases in size, the number of instructions to be processed is also increased. Hence, this factor can be equivalently converted to the number of instructions to be processed.

B. Execution on the Cloud Server

1) Determining factors affecting energy consumption

For tasks executed in the cloud, we are not considering the energy consumption of the cloud server. It is because this is a fixed system (not a mobile device) with an unlimited energy source. We consider the energy consumption of the mobile device when tasks must be uploaded to the cloud for processing.

If we upload the task to the cloud for processing, the mobile device can save energy to execute the tasks. However, it incurs energy to send the input data of the task and receive output data that was returned from the cloud. At this time, the factors that affect the energy consumption of the mobile device include:

- **The speed of sending/receiving data to the cloud server:** The **speed** of sending/receiving information to the server is one of the factors changing the run-time of the transceiver (3G/4G or Wifi), which causes energy consumption for the device.
- **Data size:** The size of the data affects the time it takes to send/receive information, thus changing energy consumption of the device during sending/receiving information.

2) Determining factors affecting execution time

Task execution time is calculated by the total time to perform 03 stages of the task, including: Time to send data to the server, processing time on the server, and time to receive returned data. Thus, the factors that affect task execution time include: Speed of sending/receiving data, the size of data, processing speed of the server.

Based on the above analysis, we can identify the factors that affect the energy consumption and processing time. Table I describes the list of those factors.

TABLE I: FACTORS AFFECTING ENERGY COSTS AND PROCESSING TIME

Decision Factor	Unit	Definition
S_{send}	Kbps	The speed of sending messages to the cloud
$S_{receive}$	Kbps	The speed of sending messages to the cloud
D^t_{input}	KB	Total input data of task ***t***
D^t_{output}	KB	Total ouput data of task ***t***
I^t_{mobile}	Int	Total number of instructions of task ***t*** when executed on mobile device
I^t_{cloud}	Int	Total number of instructions of task ***t*** when executed on cloud
μ_{mobile}	MHz	Mobile device processing speed
μ_{cloud}	MHz	Cloud server processing speed

V. CALCULATE ENERGY AND TIME COST

In order to calculate the energy costs and execution time of all the tasks, we calculate the time and energy costs for each task and then calculate the total cost for all the tasks. Table II describes the symbols during the calculation.

TABLE II: NOTATION TABLE

Values	Definition
E^t_{mobile}	Energy cost when task *t* is executed on mobile
E^t_{cloud}	Energy cost when task *t* is executed on cloud
E^t_{send}	Energy cost when the data of task *t* is sent to the cloud
$E^t_{receive}$	Energy cost when the data of task *t* is received from the cloud
T^t_{mobile}	Execution time cost of task *t* on mobile
T^t_{cloud}	Execution time cost of task *t* on cloud
T^t_{send}	Time to send data of task *t* to cloud
$T^t_{receive}$	Time to receive data of task *t* from cloud
Tasks	Set of tasks
E_{total}	Total energy to execute all tasks in Tasks
T_{total}	Total time to execute all tasks in Tasks

A. Calculating Time and Energy Costs for each Task

1) Energy-cost for each task

In order to calculate the energy consumption for execution on the mobile device, we use the Energy Automata combined with the method for estimating energy consumption that was proposed in the previous research [21]. By determining the energy states of the application and the lifetime in each of them, we calculate the energy consumption as:

$$E^t_{mobile} = \sum_{i=0}^{k} C_i \times t_i \tag{1}$$

In which, E^t_{mobile} is the execution time of task *t* on the mobile device, *k* is the number of changed states, C_i and t_i are the energy consumption at the *i* state satisfied as:

$$\sum_{i=1}^{k} t_i = T^t_{mobile} \tag{2}$$

In the next step, we calculate the energy cost of the mobile device when executed on the cloud server. In this case, the mobile device will incur energy costs to send and receive data from the server.

We have the formula as:

$$E^t_{cloud} = E^t_{send} + E^t_{receive} \tag{3}$$

In which, E^t_{send} and $E^t_{receive}$ are calculated by using the energy state model [21] with the total respective execution time of T^t_{send} and $T^t_{receive}$.

2) Time-cost for each task

The execution time of task *t* on the mobile device is calculated as the total number of commands that need to be processed multiplied by the processing speed of the mobile device as:

$$T^t_{mobile} = \frac{I^t_{mobile}}{\mu_{mobile}} \tag{4}$$

The task execution time on the cloud server is calculated as the total of the time to transfer data from the mobile device to the server, the task processing time on the server, and the time to transfer output result from the server to the mobile device as:

$$T^t_{cloud} = D^t_{input} \times S_{send} + I^t_{cloud} \times \mu_{cloud} + D^t_{ouput}\, S_{receive} \tag{5}$$

B. Calculating Time and Energy Costs for all Tasks

Assuming that each task is assigned a label (**0** or **1**) to represent either execution on the mobile device or execution on the cloud server, the execution of ***n*** tasks in ***Tasks*** will be represented by a sequence of n numbers of 0,1. Each different sequence will represent a case of executing all n tasks in ***Tasks***. For each execution case, we calculate the total energy cost and execution time in order to choose the most optimal execution case.

1) Total energy-cost

The total energy cost of $\boldsymbol{E_{total}}$ to execute all tasks in ***Tasks*** will be calculated as the total of the energy cost of executing all tasks on the mobile device and the cloud service as:

$$E_{total} = \sum_{t=1}^{n} \{E_{mobile}^{t}(label = 0) | E_{cloud}^{t}(label = 1)\} \quad (6)$$

In which, the tasks labeled as ***0*** will be calculated by the formula ***E^t_{mobile}***, and the tasks labeled as ***1*** will be calculated by the formula E^t_{cloud}.

2) Total time-cost

Similarly, the formula for the total execution time of ***n*** tasks is calculated as:

$$T_{total} = \sum_{t=1}^{n} \{T_{mobile}^{t}(label = 0) | T_{cloud}^{t}(label = 1)\} \quad (7)$$

VI. Offloading Algorithm

If each task in *Tasks* is assigned a label (0 or 1) to represent the process of executing on the mobile device or the cloud, we can determine the total number of possible executions. Specifically, with ***n*** tasks in *Tasks*, we will have 2^n execution cases. We need to find the case with the most optimal time and energy cost.

Hence, we need to solve two problems:

- *First:* What is the most optimal cost? In fact, if it benefits in time, it will lose in energy and vice versa. Therefore, we need to construct an objective function to determine the optimal cost.
- *Second:* How to calculate the cost for all cases? With *n* tasks in **Tasks**, we will have 2^n execution cases. It is not possible to calculate the cost for all cases so we choose the evolutionary algorithm to find the optimal case in the allowed time.

A. Designing Objective Function

In order to be able to find the optimal execution case, we calculate the rate of profit on energy and time cost of each execution compared to executing all tasks on the mobile device (non-offloading).

Specifically, if we consider T^x_{total} is the total execution time in the execution case of ***x*** and T^0_{total} is the total execution time without offloading, the rate of profit on the time of the ***x*** execution case compared to the non-offloading case ***γT*** will be calculated as:

$$\gamma_T = \frac{T_{total}^{x} - T_{total}^{0}}{T_{total}^{0}} \quad (8)$$

Similarly, we calculate the rate of profit on energy ***γE***:

$$\gamma_E = \frac{E_{total}^{x} - E_{total}^{0}}{E_{total}^{0}} \quad (9)$$

As for offloading the processing of ***n*** tasks, we want to offload both time and energy. However, for some practical cases, this factor is contradictory. Therefore, we need to determine which factor to prioritize?

In order to solve this problem, we propose two prioritized coefficients of time and energy ***α*** and ***β*** respectively. These two coefficients represent the priority ratio in time and energy, so we have the objective function that is expressed as a mathematical formula, depending on ***α*** and ***β*** as:

$$f(x) = \alpha \times \gamma_T + \beta \times \gamma_E \rightarrow Min \quad (10)$$

In which, ***f(x)*** is the value of the objective function in the case of executing ***x*** and $\alpha + \beta = 1$.

The priority of time or energy will be designed by the software developer depending on the characteristics of the software. If you want to prioritize energy completely, the software developer can design the coefficient $\alpha = 0$. On the contrary, if you want to prioritize execution time completely, the software developer can design $\beta = 0$. Another suggestion for software developers is that they can depend on the remaining battery energy on the mobile device to design a priority coefficient for ***β*** energy consumption.

1) Offloading algorithm

As we have shown above, offloading ***n*** tasks in ***Tasks*** will be considered as choosing the execution case with minimum objective function value ***f(x)***. Theoretically, there will be a total of 2^n execution cases, so if we can consider all cases we will find the optimal one. However, the reality is that ***n*** is often a lot and the time to make the decision to offload (choose the execution cases) is limited. Hence, the Heuristic algorithms will be normally inefficient.

We propose to use the genetic algorithm to find the best solution in a short time. In which, each chromosome is represented by a set of ***n*** the numbers **0** and **1** that will be crossed and mutated to create new chromosomes. Each generated chromosome will be calculated the value of the objective function and appreciated if the new objective function is more optimal than the previous one.

Algorithm for choosing the most optimal execution case:

Algorithm 1: Offloading Algorithm

```
    Input:
      Source_App Application Source Code
      Infor_C - Cloud Server Information
      Infor_M - Mobile Device Information
      α and β
    Output:
      $x = \{t_1, t_2, t_3, \dots t_n\}$ where $ti = 0|1, i = 1..n$ and $f(x) \rightarrow Min$
1   CF = ColectFactor(Infor_C, Infor_M)
2   $A = \emptyset$
3     $minf = f(a_0)$ where $a_0 = \{t_1, t_2, t_3, \dots t_n\}, t_i = 0, i = 1..n$
4   x = a_0
5   setTimer(timer)
6   while timer>0 do
7       A+=Genetic()
8       foreach a in A do
9           $\gamma_T = CaculateTime(CF)$
10          $\gamma_E = CaculateEnergy(CF)$
11          $f(x) = \alpha * \gamma_T + \beta * \gamma_E$
12          if $f(a) < minf$ then
13              minf = f(a)
14              x = a
15          end
16      end
17      timer--
18  end
19  Execute(x)
```

The algorithm is divided into two main steps:

1) Generating execution cases by the genetic algorithm

Assuming that ***A*** is the set of executions. At first, ***A*** is initialized as an empty set, the timer is set via the function *setTimer(timer)*. Next, the genetic algorithm *Genetic()* is executed to generate new execution cases by hybridization, mutation or selection. The new execution cases will be added to the set *A*. The timer will gradually decrease the execution time, and the genetic algorithm will stop working when the execution time ends.

2) Selecting the most optimal execution case

Firstly, we calculate the value of the objective function in the execution case ***a₀***. In which, ***a₀*** contains all the labels as ***0*** that means executing all tasks on the mobile device *(non-offloading)*. The value of the initial minimum objective function will be *minf* = *f(a₀)*. For each element in the set ***A***, we recompute the objective function as:

$$f(x) = \alpha * \gamma_T + \beta * \gamma_E \rightarrow Min \qquad (10)$$

If the objective function gives a value less than ***minf***, we will update a new optimal execution case and a new ***minf***. Finally, we can choose the optimal execution case ***x*** and make the offloading decision according to the value of ***x***: *Execute(x)*.

VII. Case Study and Result

In order to evaluate the effectiveness of the proposed method, we conducted an experiment to process 1000 photos, which are available on the mobile phone. These photos are inverted at an angle of 90 degrees from the original image. This experiment is be performed in three cases:

- *Case 1 (Non-offloading)*: Executing entirely on the mobile device.
- *Case 2 (Half-offloading)*: *Processing* randomly 500 photos on the mobile device, and uploading the remaining 500 photos to the cloud server to process.
- *Case 3 (Offloading by our method)*: Applying our proposed method to make the offloading decision, the time to make the offloading decision will be *60* seconds.

For each case, we conducted it 10 times on the actual mobile device and measured execution time and energy consumption for each execution. The result is calculated as the average of 10 executions. Subsequently, we compared the execution data of the three cases and considered the efficiency after applying our proposed algorithms.

Software: We have selected an open-source image processing software on the Google Play app store of the Android operating system. It is called Pocket Paint, which is a favorite image-processing application with a rating of 4.6 out of 5 stars. Subsequently, we downloaded the source code of this software from GitHub and edited the source code so that the software could automatically invert available 1000 images at an angle of 90 degrees for in the memory. We implemented processing decision algorithms for the three cases described above and deployed the original software into three different software versions. Finally, we installed them on the mobile device to experiment.

Environment: In order to create a mobile cloud computing environment, we used a mobile phone and a cloud server. For the mobile device, we used the Samsung Galaxy Note 8 *(CPU: Snapdragon 280, GPU: Snapdragon: Adreno 540, Ram: 6 GB, Storage: 64 GB)*. For the cloud server, we used the p2. xlarge of Amazon *(vCPU: 4, GPU: 1, Ram: 61 GiB, GPU Memory: 12GiB, Network Performance: High)*. In which, the cloud server has pre-installed image processing algorithms as on the mobile phone. Hence, it can receive data, process images, and send results to the mobile phone.

The experimental results are shown in Figs. 3–5.

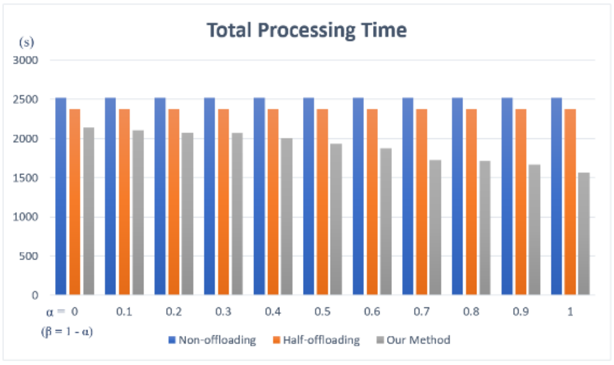

Fig. 3. Execution time comparison

As shown in Fig. 3, we compare the processing time in three cases: In the case of ***α* = *1*** and ***β* = *0***, the time to process all photos is reduced by 37.6% compared to non-offloading while the random half-offloading method is only reduced by approximately 6% of processing time.

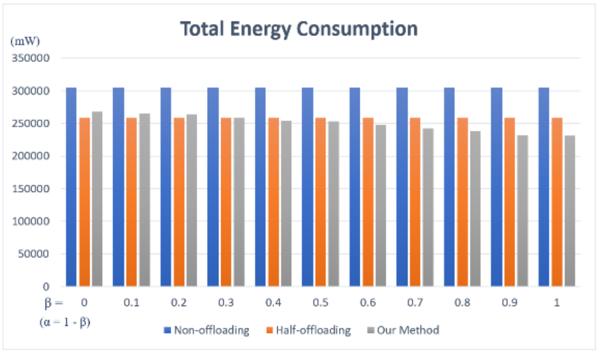

Fig. 4. Energy consumption comparison.

As shown in Fig. 4, we compare the energy consumption when experimenting with the three cases mentioned above.

Beta coefficient is adjusted from 0 to 1: In the case of ***β* = *1*** and ***α*= *0***, the offloading by our method can save 24.1% of energy consumption while the random offloading is only 15.1 %.

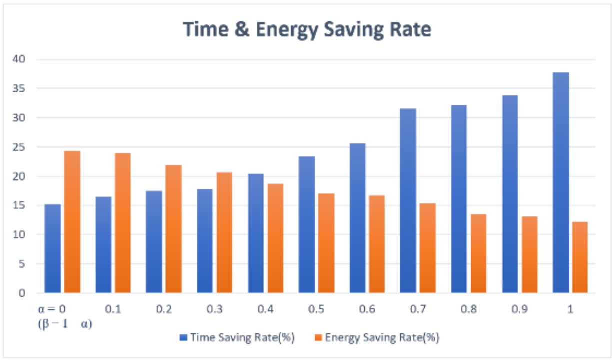

Fig. 5. Time and energy saving rate.

In order to get a better overview of the effectiveness of our method, we compare the time and energy-saving rates when adjusting for alpha and beta coefficients. The most balanced case is alpha = 0.4 and beta = 0.6, which can save 21% of execution time and 18% of energy costs.

VIII. Conclusion

In this paper, we have proposed a method to a method to offload mobile devices when it processes multiple tasks concurrently. We base the process of calculating the energy consumption and processing time of each task to calculate the total energy and time costs for all tasks in different cases to select the most optimal case. In order to determine the best case, we have proposed an objective function with two preference coefficients of time (α) and energy (β) as the basis to calculate the optimal degree of each execution case. Next, the thesis proposes an algorithm, which is based on a genetic algorithm to generate execution cases and find the most optimal execution case. After practical application, the results show that our method saves a maximum of 24.1% of energy, 37.6% of processing time, an average of 18% of energy, and 21% of processing time compared to non-offloading.

Besides the achieved results, our approach promises to open up some research directions in the future. First, this method can be applied to a set of dynamic (time-varying) tasks. Second, our method can be improved to perform on a set of non-independent tasks that do not depend on each other. Finally, we will continue to improve the genetic algorithm in order to have more optimal results.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

Ninh-Thuan Truong initiated the problem, conducted related research, and proposed solutions. Anh-Tu Bui, as an algorithm specialist, devised and implemented algorithms, rigorously testing their correctness. Van-Viet Nguyen executed experiments and evaluated the results.

Together, we established a robust knowledge base and deep understanding of the problem. Our close collaboration, spanning problem formulation, algorithm development, and practical validation, has greatly advanced the field of information technology; all authors had approved the final version.

References

[1] M. Mohammed and J. Shuja, "Computation offloading in mobile cloud computing and mobile edge computing: Survey, taxonomy, and open issues," *Mobile Information Systems*, 2022.

[2] A. Vylala and P. R. Bipin, "Exploitation whale optimization based optimal offloading approach and topology optimization in a mobile ad hoc cloud environment," *Journal of Ambient Intelligence and Humanized Computing*, vol. 13, no. 2, 2022.

[3] S. Ali and M. G. Arani, "A metaheuristic-based computation offloading in edge-cloud environment," *Journal of Ambient Intelligence and Humanized Computing*, vol. 13, no. 5, 2022.

[4] S. Vinu, "Optimal task assignment in mobile cloud computing by queue-based ant-bee algorithm," *Wireless Personal Communications*, vol. 104, no. 1, 2019.

[5] M. Meysam and C. Cavdar, "Device vs edge computing for mobile services: Delay-aware decision making to minimize power consumption," *IEEE Transactions on Mobile Computing*, vol. 20, no. 12, 2020.

[6] S. H. Kim et al., "An optimal pricing scheme for the energy-efficient mobile edge computation offloading with OFDMA," *IEEE Communications Letters*, vol. 22, no. 9, 2018.

[7] F. Lu et al., "Mildip: An energy efficient code offloading framework in mobile cloudlets," *Information Sciences*, vol. 513, 2020.

[8] G. Chen, B. T. Kang, M. Kandemir, N. Vijaykrishnan, M. J. Irwin, and R. Chandramouli, "Studying energy tradeoffs in offloading computation/ compilation in java-enabled mobile devices," *IEEE Trans. Parallel Distrib. Syst.*, vol. 15, no. 9, pp. 795–809, Sep. 2004.

[9] E. Cuervo, A. Balasubramanian, D. Cho, A. Wolman, S. Saroiu, R. Chandra, and P. Bahl, "Maui: Making smartphones last longer with code offload," in *Proc. 8th Int. Conf. MobiSys*, pp. 49–62, 2010.

[10] S. Han, S. Zhang, and Y. Zhang, "Energy saving of mobile devices based on component migration and replication in pervasive computing," in *Proc. Ubiquitous Intell. Comput.*, pp. 637–647, 2006.

[11] B. Seshasayee, R. Nathuji, and K. Schwan, "Energy-aware mobile service overlays: Cooperative dynamic power management in distributed mobile systems," in *Proc. 4th ICAC*, p. 6, 2007.

[12] B. G. Chun, S. Ihm, P. Maniatis, M. Naik, and A. Patti, "Clonecloud: Elastic execution between mobile device and cloud," in *Proc. 6th ACM SIGOPS/EuroSys*, pp. 301–314, 2011.

[13] X. Gu, A. Messer, I. Greenberg, D. Milojicic, and K. Nahrstedt, "Adaptive offloading for pervasive computing," *IEEE Pervasive Comput.*, vol. 3, no. 3, pp. 66–73, Jul./Sep. 2004.

[14] R. Wolski, S. Gurun, C. Krintz, and D. Nurmi, "Using bandwidth data to make computation offloading decisions," in *Proc. IEEE IPDPS*, Apr. pp. 1–8, 2008.

[15] I. Giurgiu, O. Riva, D. Juric, I. Krivulev, and G. Alonso, "Calling the cloud: Enabling mobile phones as interfaces to cloud applications," in *Proc. 10th ACM/IFIP/USENIX Int. Conf. Middleware*, pp. 1–20, 2009.

[16] K. Yang, S. Ou, and H. Chen, "On effective offloading services for resource-constrained mobile devices running heavier mobile Internet applications," *IEEE Commun. Mag.*, vol. 46, no. 1, pp. 56–63, Jan. 2008.

[17] A. Miettinen and J. Nurminen, "Energy efficiency of mobile clients in cloud computing," in *Proc. 2nd USENIX Conf. HotCloud*, p. 4, 2010.

[18] C. Wang and Z. Li, "A computation offloading scheme on handheld devices," *J. Parallel Distrib. Comput.*, vol. 64, no. 6, pp. 740–746, Jun. 2004.

[19] S. Kosta, A. Aucinas, P. Hui, R. Mortier, and X. Zhang, "ThinkAir: Dynamic resource allocation and parallel execution in the cloud for mobile code offloading," in *Proc. 31st Annu. IEEE (INFOCOM) Int. Conf. Comput. Commun.*, p. 806, 2012.

[20] R. Kemp, N. Palmer, and T. Kielmann, "Cuckoo: A computation offloading framework for smartphones," in *Proc. 2nd Int. Conf. Mobile Comput., Appl., Serv.*, Santa Clara, CA, USA, vol. 76, 2010.

[21] A. H. Le, A. T. Bui, and N. T. Truong, "An approach to modeling and estimating power consumption of mobile applications," *Mobile networks and Applications*, vol. 24, no. 1, 2019.

[22] G. Chen, B. T. Kang, M. Kandemir, N. Vijaykrishnan, M. J. Irwin, and R. Chandramouli, "Studying energy tradeoffs in offloading computation/ compilation in java-enabled mobile devices," *IEEE Trans. Parallel Distrib. Syst.*, vol. 15, no. 9, pp. 795–809, Sep. 2004.

[23] K. Kumar and Y. Lu, "Cloud computing for mobile users: Can offloading computation save energy?" *Computer*, vol. 43, no. 4, pp. 51–56, Apr. 2010.

[24] Y. S. Hao et al., "Interval grey number of energy consumption helps task offloading in the mobile environment," *ICT Express*, vol. 9, no. 3, 2022.

[25] S. J. Raj, "Improved response time and energy management for mobile cloud computing using computational offloading," *Journal of ISMAC*, vol. 2, no. 1, 2020.

[26] H. F. Lu et al., "Optimization of lightweight task offloading strategy for mobile edge computing based on deep reinforcement learning," *Future Generation Computer Systems*, vol. 102, 2020.

[27] X. Zhao, P. Tao, S. Yang, and F. Kong, "Computation offloading for H.264 video encoder on mobile devices," in *Proc. IMACS*, pp. 1426–1430, 2006.

[28] C. Cai, L. Wang, S. U. Khan, and J. Tao, "Energy-aware high-performance computing: A taxonomy study," in *Proc. 17th IEEE ICPADS*, pp. 953–958, 2011.

[29] M. V. Barbera, S. Kosta, A. Mei, and J. Stefa, "To offload or not to offload? The bandwidth and energy costs of mobile cloud computing," in *Proc. IEEE INFOCOM*, pp. 1285–1293, 2013.