AN OFFLOAD SCHEME FOR ENERGY OPTIMIZATION IN MOBILE EDGE COMPUTING SYSTEMS

Hoàng Trọng Minh[*], Nguyễn Thanh Trà[+], Hoàng Thị Thu[+]
[*]Khoa Viễn thông, Học viện Công nghệ Bưu chính Viễn thông
[+] Viện công nghệ Thông tin và Truyền thông, Học viện Công nghệ Bưu chính Viễn thông

Abstract: Currently, edge computing technology has attracted a lot of research due to its ability to provide distributed computing, optimize energy, and improve processing speeds for users. The advantages of approaching edge computing are the sharing of computational tasks between devices and access devices at the network edge to reduce backbone traffic and delay. An offloading solution for supported devices to compute part of a task locally instead of moving the whole calculations to the Mobile Edge Computing (MEC) device, which is the core of the approach to reduce latency and accelerate processing. However, an optimal solution for multiple constrain problems belongs to the NP-Hard class problem. Therefore, enhancing the network performance of edge computing through an offload solution is still opening issues. In this paper, an offloading mechanism is carried out alternately for the proposed support device to optimize the overall energy of the equipment while still satisfying the conditions of latency constrain and computational requirements. The proposed algorithm is validated by the numerical results that show certain advantages of this optimized solution.

Keywords: Mobile Edge Computing, optimization, linear programming, D2D communication, network performance.

I. INTRODUCTION

The explosive development of mobile devices and services in recent times brings a lot of utility to users and has also created a series of challenges for the communications network infrastructure. The fast, efficient computing requirements of the terminals demand new networking solutions. The cloud computing system, Fog Network, and Edge computing are ones of the recent approaches to addressing computing and connectivity needs for IoT (Internet of Things) [1]. IoT is now infiltrating our daily lives, providing tools to measure and gather important information to support decisions. Sensors and terminals are continuously generating data and exchanging information over the wireless communications infrastructure including Machine-to-Machine communications and Intelligent Computing.

As a strategy to lessen the escalation of resource congestion, edge computing has become a new paradigm to address the needs of IoT and compute localization. Besides the ability to connect large numbers of terminals, reducing transmission latency time and energy efficiency has been a subject of many researchers and deployed interested in the current edge computing model [2, 3].

MEC is a distributed computing solution at the network edge for mobile devices connected via wireless media. MEC reduces centralized computing pressure for cloud computing and reduces information processing latency for computing requests from terminals. This distributed, traffic-balanced architecture is deployed in a wide range of practical applications [4, 5]. Field research reduces a load of computing to address the sending of tasks to devices that play the support role (Helper) and to the MEC server. The servers are capable of delivering a lot more computing resources than mobile devices (MD) but the communication latency is very large compared to direct connections between the MD. With the mission requirements from different MD, the load reduction strategies are launched to simultaneously satisfy the constraints to enhance network performance using MEC. Therefore, the load reduction targets often include reduced energy consumption and execution time by spending on-demand Tasks [6, 7].

To implement load reduction strategies, centralized and distributed computing models at the edge of the network are conducted with small or non-cloud architectures [8, 9]. The optimum solutions based on heuristic or mathematical analysis are proposed to search for optimal target functions [10]. However, according to the best understanding of the authors' group, the approach using rotating helpers for load reduction requirements has not been mentioned in previous studies. Therefore, this paper will present an optimal solution for the edge computing system to optimize the energy of mobile devices while adapting to the input task requirements along with the latency as required. The layout of the paper is as follows: The next section will state the research of previous authors related to the content of the study, part III will present the proposed model, the hypothesis, the simulation switches, and the final part will present the resulting conclusions as well as the direction of subsequent development.

Tác giả liên hệ: Hoàng Trọng Minh,
Email: hoangtrongminh@ptit.edu.vn
Đến tòa soạn: 8/2020, chỉnh sửa: 9/2020, chấp nhận đăng: 10/2020.

II. RELATED WORK

Edge computing's trend is to process data near the source with support from the terminal mobile devices themselves. The growing number of intelligence apps has set new challenges in real-time data processing as well as resource optimization. To carry out the reduction of the load for local computing at MEC devices and servers, the load reduction model in [11] has been proposed in binary style and for each component of the required tasks. Based on timeframe T, computing tasks at the mobile device, assistive devices, and at the AP are allocated and optimized using a linear programming method. This solution allows optimum energy consumption of the process to perform the calculation of all required tasks with strict latency conditions. However, the study did not mention the processing for the consecutive timeframes and only used 01 devices that support the load reduction. As a scheduler, the author group in [12] has proposed an automatic load reduction in the order of prioritization of tasks. With services that require strict latency limits, computing resources are allocated high priority and minimize computational time as well as affective computing performance. Despite this, the preprocessing steps at the same time in the same timeframe are a major obstacle to the progress requirements.

In search of the optimal load reduction strategy, a series of proposals based on game theory has been introduced. The multi-purpose optimization problems of latency and application requirements are exploited through the balanced characteristics of game theory [13]. Combining a load of tasks with power control, the authors in [14] have used reinforcement learning to approximate the optimum problem of resource optimization for mobile devices to avoid the issue of NP-Hard problems. The above proposal suggests that balancing the energy of terminals in load-reduction processes is a key issue in the goal of improving the system performance MEC. Therefore, this paper will approach load balancing problems through the choice of useful support devices by each round of access to optimize the overall energy of the equipment in the process of operation. Conditions that meet the mission input and delay limits requirements will be ensured with the balanced energy balance that is approached by the integer linear programming method. The system model, input conditions, and proof of results of the study will be presented below.

III. THE PROPOSED MODEL

This section describes the proposal from a typical model of edge computing with the symbols used in the study. Assuming a MEC system consisting of three basic components including user devices, support elements, and the AP access point that are integrated with MEC servers as in Figure 1. In a simple form, MEC servers are attached by AP to process local computations. The user, with connections to helpers, can transfers data and requests support to process data; both user and helpers are connected to the MEC through the AP.

With the assumption that user device moves with a certain probability between the cells served by the support element 1 and 2. To resolve the issue, two support elements reduce the computational load for the user device to optimize computing power, limit latency, and ensure user mobility. The symbols described in the paper are denoted in table I.

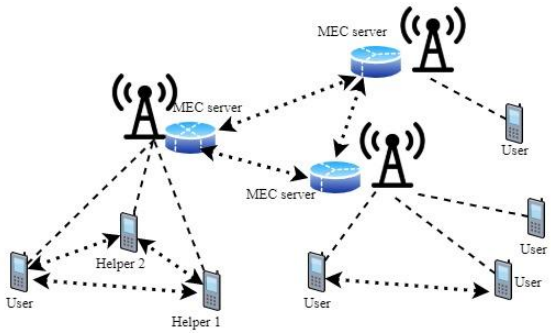

Figure 1. MEC system's configuration

Table I. Related Parameters

$l_{i,u}$	Number of task bits processed at the user device in *i* round
$l_{i,h}$	Number of task bits processed at the support element in *i* round
$l_{i,a}$	The number of tasks bits processed at the AP in *i* round
C	CPU Cycles require to process 1 bit
f_u	Ability to process at user devices
f_h	Processing capability at the support element
f_a	Processing capability in *AP*
k	Capacitance factor
r_a	The transfer rate from a user device to *AP*
r_h	The transfer rate from user devices to Helper support elements
P_{tx}	Transport capacity

The algorithm focuses on time slots that have a duration $T * > 0$, in which the user needs to handle all the bits of the input task $L > 0$. We have $L = \{l_1, l_2, l_3,...,l_n\}$ which is the set of bits to process from the user device. With the input bits, l_i can be divided into 3 parts $l_{i,u}$, $l_{i.h}$, $l_{i,a}$ for computing at the user device, supported elements, and at the respective AP (access point). Hence we have the formula:

$$l_{i,u} + l_{i,h} + l_{i,a} = l_i. \quad (1)$$

A. Computing and communication models at user devices

(i) Computing model at user devices

The number of bits needed to be processed at the user device is $l_{i,u}$ and so it needs $l_{i,u}.C$ period. The latency of the computing at the user device is denoted by T and is computed as the following formula:

$$\tau_{i,u}^{comp} = \frac{l_{i,u}.C}{f_u} \quad (2)$$

We consider a model of low-voltage task execution and energy consumed by a CPU cycle [15] computed by formula $k.f_u^2$ where k is the capacitive constant. Computing power consumption at user devices is performed as formula (3) below:

$$E_{i,u}^{comp} = l_{i,u}.C.k.f_u^2 \quad (3)$$

In which, $E_{i,u}^{comp}$ is the computing power consumed by user device in i^{th} round.

(ii) Transmission model at the user device.

The load is reduced for the user device that needs to be transferred to the support element and the previous *AP*. Therefore, the estimated transmission time is computed as follows:

$$\tau_{i,u}^{trans} = \frac{l_{i,h}}{r_h} + \frac{l_{i,a}}{r_a} \quad (4)$$

Power consumption of the transmission at the respective user device is calculated as:

$$E_{i,u}^{trans} = \tau_{i,u}^{trans}.P_{tx} = (\frac{l_{i,h}}{r_h} + \frac{l_{i,a}}{r_a})P_{tx} \quad (5)$$

B. Computing and transmission models at the support element and access point

(i) The computing model at the support element.

The support element has limited computational power because of the limited energy compared to the access point. The ability to compute load element support is signed as f_h. The workload of the support element from the i user device is $l_{i,h}$, and its computational period number $l_{i,h}.C$. The computing time at the support element is computed as follows:

$$\tau_{i,h}^{comp} = \frac{l_{i,h}.C}{f_h}. \quad (6)$$

Energy consumed for computing at the performance support element as:

$$E_{i,h}^{comp} = l_{i,h}.C.k.f_h^2. \quad (7)$$

After calculating a part of the task, the number of bits is transmitted from the support element to the user device. Therefore, the transmission time corresponds to the following:

$$\tau_{i,h}^{trans} = \frac{l_{i,h}}{r_h}. \quad (8)$$

The energy consumed for transmission at the support element is as follows:

$$E_{i,h}^{trans} = \tau_{i,h}^{trans}.P_{tx}. \quad (9)$$

Total latency at the support element is made up of transmission latency and computing delay in the form of:

$$\tau_h = \tau_{i,h}^{comp} + 2.\tau_{i,h}^{trans}. \quad (10)$$

(ii) Computing model at the access point.

Ignoring the computing power and transmission power at the access point, we only consider computing latency and transmission delays from the access point to the user's device. The workload of the access point transmitted from the i user device is and the number of its computational cycles is $l_{i,a}.l_{i,a}.C$. Computing time at the respective access point:

$$\tau_{i,a}^{comp} = \frac{l_{i,a}.C}{f_a}. \quad (11)$$

After calculating a part of the task in the access point, the number of cut down bits is transmitted to the access point. Therefore, the actual transmission time is as follows:

$$\tau_{i,a}^{comp} = \frac{l_{i,a}}{r_a}. \quad (12)$$

Total latency at the access point includes the computing delay and transmission delay respectively:

$$\tau_a = \tau_{i,a}^{comp} + 2.\tau_{i,a}^{trans}. \quad (13)$$

C. Constructing problem

Based on the equation (3) and equation (5), the energy consumption of the user equipment including computational energy and transmission energy is performed in the form of:

$$E_u = E_{i,u}^{comp} + E_{i,u}^{trans}. \quad (14)$$

The task of the user's device is executed in parallel in three components (user equipment, supporting element, and access point), and the following is the execution latency of τ_i:

$$\tau_i = \max\{\tau_{i,u}^{comp}, \tau_h, \tau_a\}. \quad (15)$$

Energy-efficiency issues in the processing of task bits based on delay limits are considered to meet practice requirements. We need to find out a solution reached the minimum energy of all user devices as the target function as below.

$$\boldsymbol{P:}\ Min\sum_{i=1}^{n} E = E_{i,u}^{comp} + E_{i,u}^{trans} + E_{i,h}^{comp} + E_{i,h}^{trans} \quad (16)$$

s.t:

$$l_{i,u} + l_{i,h} + l_{i,a} = l_i \quad (16a)$$

$$E_{i,u}^{comp} + E_{i,u}^{trans} \le \gamma E_u \quad (16b)$$

$$E_{i,h}^{comp} + E_{i,h}^{trans} \le \gamma E_h \quad (16c)$$

$$\tau_{i,u}^{comp} \le T* \quad (16d)$$

$$\tau_h \le T* \quad (16e)$$

$$\tau_a \le T* \quad (16f)$$

Where, $T*$ is the maximum time limit for processing every task. (16a) represents the task partition constraint; (16b) and (16c) as the power constraints available at the user equipment and support elements. In which, γ ratio factor presents the maximum emitted energy of a user (16d) (16e) and (16f) that show time constraints. Note

that the problem (16) applies the integer linear programming (*ILP*) method so that we can effectively resolve it through standard convex optimization techniques such as the interior point method.

IV. SIMULATION RESULTS AND DISCUSSIONS

The above proposal for integer linear programming (ILP) aims to optimize energy consumption in the MEC system with multiple access rounds. Therefore, energy consumption constraints are computed locally, transmitted on each component, and latency limits are intended to provide the most optimal approach from the multitude of decision-making schemes. To verify the model, CPLEX software is used to calculate the optimization of total energy consumed. CPLEX Optimizer provides flexible, high-performance mathematical programming solvers for linear programming, mixed-integer programming, quadratic programming, and quadratically constrained programming problems.

The characteristic of the energy depends on the number of bits the input task is performed as in Figure 2. The computing bit count at the support element cut from the user device decreases after each round, against the number of bits computed at the point of access cuts from the increased user device. Thus, the number result is given to evaluate the implementation of allocation of bits of input computing in the following three scenarios:

Scenario 1: Scheduling computing: The system consists of three basic buttons consisting of a user device, support element, and access point.

Scenario 2: Scheduling computing and changes to support elements: The system includes user devices, support elements, access points, and backup support elements.

Scenario 3: Scheduling computing and Support element selection: The system includes the user device, the first support element, the second support element, and the access point.

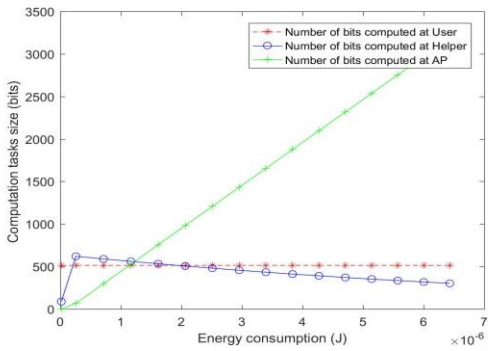

Figure 2. Distribution of task bits when there is a support element.

Assuming the set of input parameters three simulation scenarios are fixed. The task bits at each round of user devices change incrementally in the range of $600 < L_i < 4000$ *(bits)* in that CPU cycle = *250 (cycle/bit)*, Latency $T^* = 0.45$ *s*. The ability to compute locally at user devices $f_u = 2.85.10^5$ *(cycle/s)* and capacitive coefficient $k = 10^{-28}$ [16]. In addition, the computational capability at the support element and the access point is respectively $f_h = 15.10^5$ (*cycle/s*), $f_a = 20.10^5$ *(cycle/s)*. Maximum transmission capacity $P_{tx} = 0.0002$ *Watts*. The transfer speed from the user device to the support element is $r = 10^5$ *(bit/s)*. With the initial energy initializing $E_u = 3.10^{-3}$ *(j)*, $= 2.5.10\, E_h{}^6$ *(j)* The energy will vary depending on the computing task rounds.

After each computational task, the computational power of the support element decreases, depending on the remaining energy after each cut-off task. Mission bits offload at the support element (the Blue line) represents the ability to compute the linear descending task based on the remaining energy levels. Figure 2 shows energy consumptions depending on required tasks, the computational bits at the user and helpers are equivalent to keep load balancing.

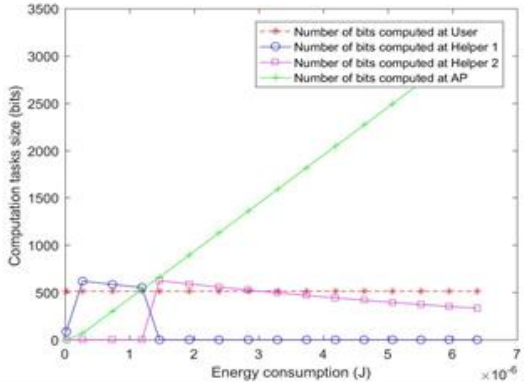

Figure 3. Distribution of task bits combining support element conversion

Assume at a time when any user device is out of the overlay of the *1*support element. The transformation of the task bits from the *1* and *2* support elements is shown in Figure 3. As a result, the interaction between the two support elements indicates the flexibility and mobility of the user device are still guaranteed. Load processing is interchangeable between the user and helper to adapt the required delay constraint. Besides, to choose the energy-based support element, we describe the simulation results as Figure 4 below.

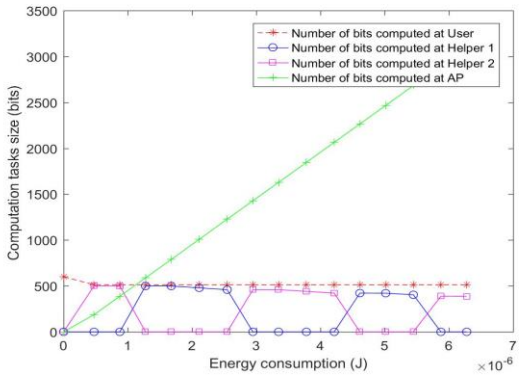

Figure 4. Combined task distributions

In Figure 4, at each round of computing tasks instead of selecting a random support element at any time, we have a solution based on energy optimization. The support element with a higher level of power will be preferred to cut down on the computation load. Therefore, the total energy consumed by the entire network element will be minimal and maintain the lifetime of mobile devices in the network.

V. CONCLUSION

By using the integer linear programming method, the overall energy optimization issue of multi-round task distributions has been solved for the *MEC* system. Input task variable and delay constraints are computed and reasonably allocated to support elements as a helper. Simulation results show the ability to meet the most non-native mobile carriers in terms of local processing capability and plan for reduced load efficiency. Based on the background knowledge of this study, it is possible to scale up with many user devices or build a smart computing strategy to select helpers in intelligent computing algorithms, and that is also the next research direction of the research.

REFERENCES

[1] N. Abbas, Y. Zhang, A. Taherkordi and T. Skeie, "Mobile Edge Computing: A Survey," in IEEE Internet of Things Journal, vol. 5, no. 1, pp. 450-465, Feb 2018.

[2] X. Xu et al., "A computation offloading method over big data for IoT-enabled cloud-edge computing," Future. Generation. Computer. Systems, vol. 95, pp. 522–533, 2019.

[3] L. Huang, S. Bi and Y. J. Zhang, "Deep Reinforcement Learning for Online Computation Offloading in Wireless Powered Mobile-Edge Computing Networks," in IEEE Transactions on Mobile Computing, doi: 10.1109/TMC.2019.2928811, 2019.

[4] Z. Ning, J. Huang, X. Wang, J. J. P. C. Rodrigues and L. Guo, "Mobile Edge Computing-Enabled Internet of Vehicles: Toward Energy-Efficient Scheduling," in IEEE Network, vol. 33, no. 5, pp. 198-205, Sept.-Oct. 2019.

[5] A. H. Sodhro, Z. Luo, A. K. Sangaiah, and S. W. Baik, "Mobile edge computing based QoS optimization in medical healthcare applications," Int. J. Inf. Manage., vol. 45, pp. 308–318, 2019.

[6] P. Zhao, H. Tian, C. Qin, G. Nie, "Energy-Saving Offloading by Jointly Allocating Radio and Computational Resources for Mobile Edge Computing,", IEEE Access Vol.5, 2017.

[7] J. Zhang, X. Hu, Z. Ning, E. C. Ngai, L. Zhou, J. Wei, J. Cheng,B. Hu, "Energy-latency Trade-off for Energy-aware Offloading in Mobile Edge Computing Networks," IEEE Internet Things Journal, Vol.4662, 2017.

[8] J. Ren, G. Yu, Y. Cai, Y. He, "Latency optimization for resource allocation in mobile-edge computation offloading, " IEEE Trans. Wirel. Commun. Vol.17, 2018.

[9] L. Yang, H. Zhang, X. Li, H. Ji, V. C. Leung, "A Distributed Computation Offloading Strategy in Small-Cell Networks Integrated With Mobile Edge Computing," IEEE/ACM Trans. Netw., 2018.

[10] Q.-V. V. Pham, T. Leanh, N. H. Tran, B. J. Park, C. S. Hong, "Decentralized Computation Offloading and Resource Allocation for Mobile-EdgeComputing: A Matching Game Approach," IEEE Access 6, 2018.

[11] X. Cao, F. Wang, J. Xu, R. Zhang, and S. Cui, "Joint computation and communication cooperation for energy-efficient mobile edge computing," IEEE Internet Things J., vol. 6, no. 3, pp. 4188–4200, 2019.

[12] H. A. Alameddine, S. Sharafeddine, S. Sebbah, S. Ayoubi, and C. Assi, "Dynamic Task Offloading and Scheduling for Low-Latency IoT Services in Multi-Access Edge Computing," IEEE J. Sel. Areas Commun., vol. 37, no. 3, pp. 668–682, 2019.

[13] Shakarami, A., Shahidinejad, A., &Ghobaei-Arani, M. "A review on the computation offloading approaches in mobile edge computing: A game-theoretic perspective." Software: Practice and Experience, vol.50, pp. 1719–1759, 2020.

[14] Zhang, Bingxin & Zhang, Guopeng & Sun, Weice & Yang, Kuanli. "Task Offloading with Power Control for Mobile Edge Computing Using Reinforcement Learning-Based Markov Decision Process." Mobile Information Systems, vol. 2020, Article ID 7630275, 6 pages, 2020.

[15] Y. Pei, Z. Peng, Z. Wang, and H. Wang "Energy-Efficient Mobile Edge Computing: Three-Tier Computing under Heterogeneous Networks," Wireless Communications and Mobile Computing journal, vol. 2020, Article ID 6098786, 17 pages, 2020.

[16] F. Wang, J. Xu, X. Wang and S. Cui, "Joint Offloading and Computing Optimization in Wireless Powered Mobile-Edge Computing Systems," in IEEE Transactions on Wireless Communications, vol. 17, no. 3, pp. 1784-1797, March 2018.

MỘT LƯỢC ĐỒ GIẢM TẢI ĐỂ TỐI ƯU NĂNG LƯỢNG TRONG CÁC HỆ THỐNG TÍNH TOÁN BIÊN DI ĐỘNG

Tóm tắt: Hiện nay, công nghệ điện toán biên đã thu hút rất nhiều nghiên cứu do khả năng cung cấp tính toán phân tán, tối ưu hóa năng lượng và cải thiện tốc độ xử lý cho người dùng. Ưu điểm của tiếp cận điện toán biên là sự chia sẻ các tác vụ tính toán giữa các thiết bị và biên mạng để giảm lượng tính toán tại trung tâm và đáp ứng thời gian trễ nhỏ. Giải pháp giảm tải được sử dụng cho các thiết bị để hỗ trợ để tính toán một phần nhiệm vụ tại chỗ thay vì chuyển toàn bộ tính toán sang thiết bị điện toán biên di động (MEC: Mobile Edge Computing), đây là cốt lõi của phương pháp tiếp cận để nhằm giảm độ trễ và tăng tốc xử lý. Tuy nhiên, một giải pháp tối ưu cho các bài toán nhiều ràng buộc thường thuộc về lớp bài toán NP-Hard. Do đó, việc nâng cao hiệu suất mạng của tính toán biên thông qua giải pháp giảm tải vẫn còn đang là vấn đề mở. Trong bài báo này, cơ chế giảm tải được thực hiện luân phiên cho thiết bị hỗ trợ được đề xuất để tối ưu hóa năng lượng tổng thể của thiết bị, trong khi vẫn đáp ứng các điều kiện về giới hạn độ trễ và yêu cầu tính toán. Thuật toán đề xuất được chứng minh bởi các kết quả mô phỏng số cho thấy những ưu điểm nhất định của giải pháp tối ưu hóa này.

***Từ khóa*:** Điện toán biên, quy hoạch tuyến tính, truyền thông D2D, tối ưu hóa, hiệu năng mạng.

Hoàng Trọng Minh tốt nghiệp đại học Bách khoa Hà Nội (1994), tiến sỹ chuyên ngành Kỹ thuật viễn thông tại Học viện Công nghệ Bưu chính Viễn thông (2014). Hiện đang là giảng viên tại Khoa Viễn thông 1, Học Viện CNBCVT. Các lĩnh vực nghiên cứu liên quan bao gồm: tối ưu, điều khiển và bảo mật mạng truyền thông.