Wang et al. EURASIP Journal on Wireless Communications and Networking (2019) 2019:133
https://doi.org/10.1186/s13638-019-1469-2

EURASIP Journal on Wireless Communications and Networking

RESEARCH Open Access

An energy saving based on task migration for mobile edge computing

Yichuan Wang*, He Zhu, Xinhong Hei, Yue Kong, Wenjiang Ji and Lei Zhu

Abstract

Mobile edge computing (MEC), as the key technology to improve user experience in a 5G network, can effectively reduce network transmission delay. Task migration can migrate complex tasks to remote edge servers through wireless networks, solving the problems of insufficient computing capacity and limited battery capacity of mobile terminals. Therefore, in order to solve the problem of "how to realize low-energy migration of complex dependent applications," a subtask partitioning model with minimum energy consumption is constructed based on the relationship between the subtasks. Aiming at the problem of execution time constraints, the genetic algorithm is used to find the optimal solution, and the migration decision results of each subtask are obtained. In addition, an improved algorithm based on a genetic algorithm is proposed to dynamically adjust the optimal solution obtained by genetic algorithm by determining the proportion of task energy consumption and mobile phone residual power. According to the experimental results, it can be concluded that the fine-grained task migration strategy combines the advantages of mobile edge computing, not only satisfies the smooth execution of tasks, but also reduces the energy consumption of terminal mobile devices. In addition, experiments show that the improved algorithm is more in line with users' expectations. When the residual power of mobile devices is reduced to a certain value, tasks are migrated to MEC server to prolong standby time.

Keywords: 5G, Computation offloading, Mobile edge computing, Energy saving, Internet of Things

1 Introduction

With the Internet of Things and the mobile Internet are booming, people have entered the era of the Internet of Everything. With the rapid increase in the number of network edge devices, mass data is generated by the perception layer of the Internet of Things, which leads to a sharp increase in the load of the cloud computing network, resulting in a long network delay [1]. With the rise of mobile edge computing, the storage services and computing services can be provided on the edge of the network [2]. At the same time, the task of intelligent terminal devices can be migrated to the mobile edge computing server to solve the problem of insufficient mobile terminal resources, and effectively reduce the delay and energy consumption. These features make it a key technology to improve the 5G network [3] user experience in the future.

Task migration technology can transfer complex tasks on mobile devices to remote edge server through wireless networks, rely on the rich computing resources of remote servers and complex computing tasks, and return the results to mobile devices, so as to solve the problems of inadequate computing power and limited battery capacity of mobile terminals.

At present, the existing task migration strategy is to make the migration decision under the premise that the migration service node has been established [4]. It does not take into account the scenarios when the multi-service nodes are available, and cannot give full play to the characteristics and advantages of mobile edge computing. The earlier task migration algorithms are proposed for cloud computing platforms. Generally, the task is completely migrated as a migration object, which needs a great migration power [5, 6]. In some optimized strategies, tasks are divided into fine-grained subtasks with chain relationship [7], and a minimize task completion time model is constructed to solve the problem. Although the task partition is refined, it is only applicable to the case where the subtasks are chain relationship, and it is not suitable for the application where the dependency relationship between subtasks is

* Correspondence: chuan@xaut.edu.cn
School of Computer Science and Engineering, Xi'an University of Technology, Xi'an, China

complex. Therefore, it is an urgent problem to study the migration strategy based on edge computing platform.

The rest of the paper is organized as follows. Section 1 introduces the overview of mobile edge computing. Some correlation background is introduced in Section 2. Including the software architecture of task migration system, the research status of edge computing and task migration at home and abroad as well as related migration strategies. In Section 3, we propose two algorithms: a task migration strategy based on genetic algorithm and a task migration energy-saving strategy algorithm considering battery residual power. In Section 4, experimental simulation and results are presented, and the performance of the algorithm is analyzed. Finally, the work is summarized in Section 5.

2 Background

2.1 Mobile edge computing

Edge computing refers to the ability of cloud computing "sink" to the edge of the network. It is a new computing mode to execute tasks on network edge devices. Edge computing refers to any resource between the data source generated by network edge terminal devices and the data path of cloud computing center. Its basic idea is to transport tasks to a resource center which close to the data source [8, 9]. The relationship between edge computing and cloud computing is not one or the other, but complementary. They make up for each other's shortcomings and jointly promote the development of the Internet of Things.

The concept of mobile edge computing (MEC) was first proposed by Carnegie Mellon University in 2009. The European Telecommunication Standards Association (ETSI) formally defined the concept of MEC in 2014 and set up ETSI Mobile Edge Computing Industry Specification Group. At the same time, the mobile edge computing introductory technical white paper was published. Mobile edge computing is a form of edge computing, which is located between wireless access point (such as WIFI) and wired network. It makes traditional wireless access network have the conditions of service localization and close deployment, reduces the load of the core network, and reduces the bandwidth requirement of data service for network return.

With the development of mobile network to 5G stage, MEC, as its key technology, will bring infinite possibilities for network services and other services. MEC technology transforms the traditional cloud computing centralized storage data mode into mobile edge data storage. This technology can reduce the delay in network operation and service delivery [10]. It is the key technology to provide user experience in the future 5G network.

2.2 Task migration

Task migration is an important way to solve the resource constraints of mobile terminals. It transmits computing-intensive tasks of mobile terminals from local to remote devices for execution, uses remote resources to expand local resources, and finally returns the results to mobile

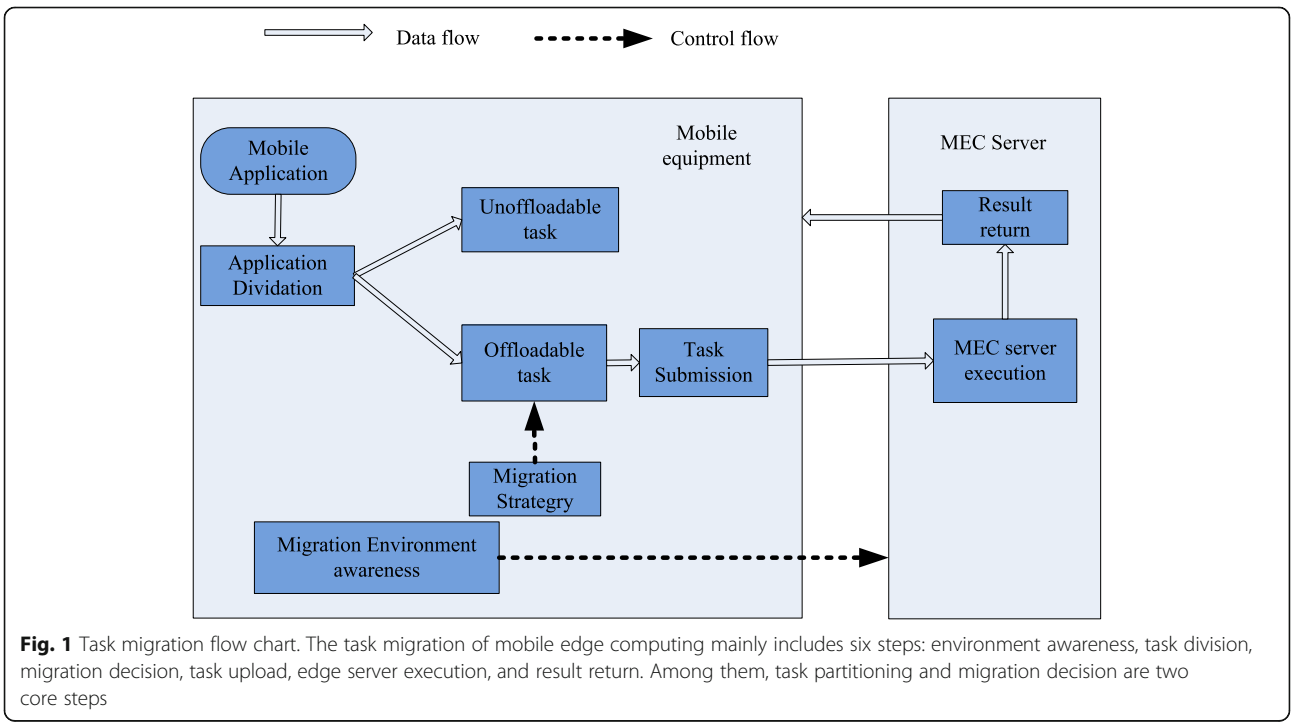

Fig. 1 Task migration flow chart. The task migration of mobile edge computing mainly includes six steps: environment awareness, task division, migration decision, task upload, edge server execution, and result return. Among them, task partitioning and migration decision are two core steps

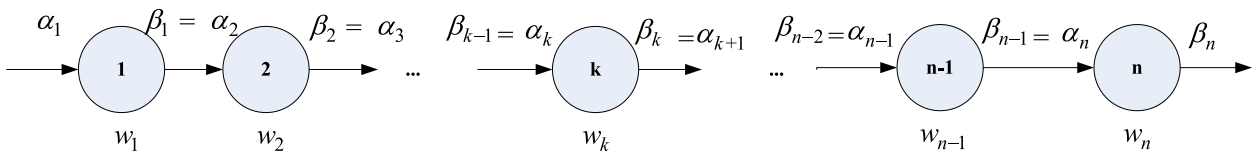

Fig. 2 Fine-grained linear task topology. It is a simple fine-grained linear task topology diagram. Each subtask is executed in sequence. The input data of each subtask is provided by the previous subtask, and its input data is the output data of the last subtask

terminals [11]. The process of task migration is as follows: first, the smart mobile terminal sends task requests to the mobile edge computing platform, then the mobile edge computing platform executes corresponding tasks according to the task requests, and finally the mobile edge computing platform returns the task brother-in-law to the user. The specific flow chart for task migration is shown in Fig. 1.

In the implementation of the migration strategy, the amount of electricity and time consumed to complete tasks are two decisive factors affecting the quality of service, especially in wireless networks. Therefore, under acceptable time constraints, minimizing the energy consumption of mobile terminals is a challenging issue.

2.3 Related works

Migrating data flow of network applications and tasks of computing-intensive applications to cloud or small servers has become a recognized solution to the problem of insufficient resources of mobile devices. The MAUI task migration platform proposed by Cuervo E and Cho D K et al. supports task migration by identifying application code; Chae D and Kim J et al. put forward a cost-focused approach. The overhead task migration platform CMcloud migrates as many mobile applications as possible to a single server to minimize server costs. Lagersoetz E [12] puts forward a criterion for migration evaluation: first, the energy consumption needed for task migration is judged; if the energy consumption for migration is greater than that for local execution, the mobile terminal is calculated; conversely, the migration calculation is carried out. It is concluded that task migration has the best energy-saving effect when the amount of data is smaller, the amount of computation is larger and the bandwidth is higher.

Generally speaking, the existing task migration strategies are divided into three categories: the whole application is executed in the cloud [13]; the whole application is migrated to the cloud to execute or to the mobile to execute according to the power saving situation; part of the migrable tasks are migrated to the cloud to execute, and the rest are completed in the mobile end [14]. However, the existing migration strategies do not take into account the multi-dependence between subtasks, and the mobile terminal battery development speed cannot meet the growing demand for energy consumption of mobile terminals [15].

Therefore, this paper will focus on energy consumption, propose a task migration strategy for complex dependency mobile terminal applications, and solve the problem of "how to achieve low-energy migration for complex dependency mobile applications."

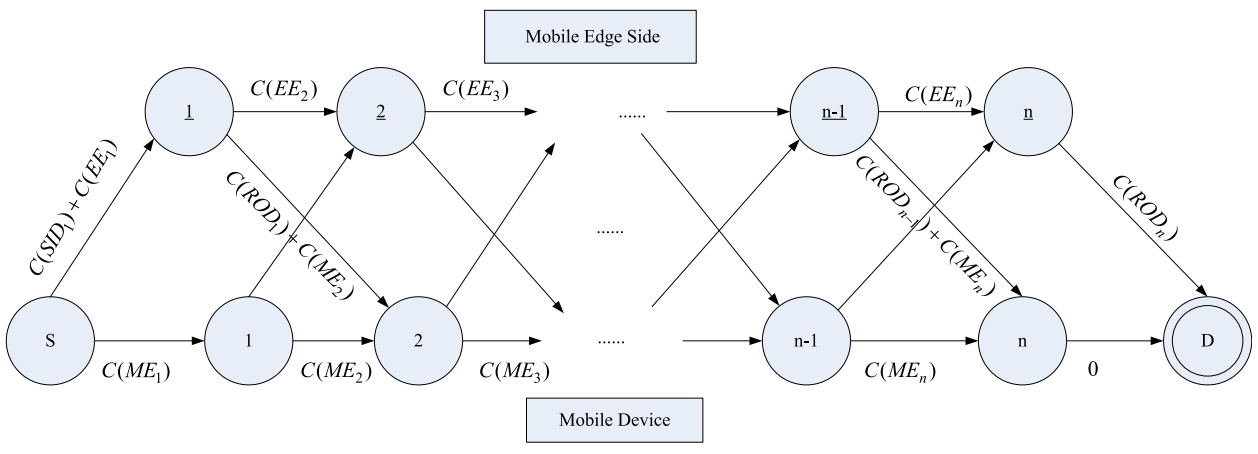

Fig. 3 Fine-grained linear task execution flow. It is a simple fine-grained linear task topology diagram. It considers that each subtask is either executed on the mobile terminal side or executed on the mobile edge side, and the calculated energy consumption generated when the task is performed on the mobile terminal side is denoted as *C*(*ME*), and the energy consumption generated when the task is performed on the mobile edge side is denoted as *C*(*EE*). Send energy consumption is *C*(*SID*), and receive energy consumption is *C*(*ROD*)

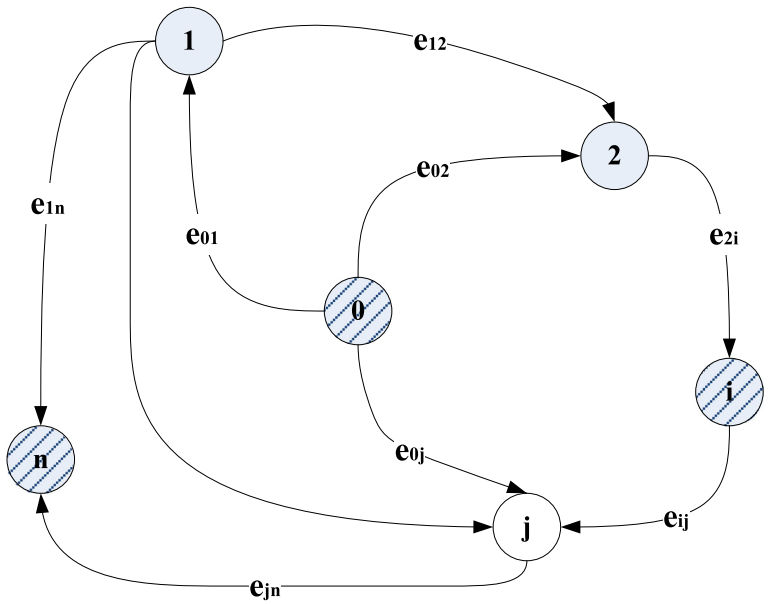

Fig. 4 Fine-grained directed acyclic graph task topology model. It is a topological model of a fine-grained directed acyclic graph task topology. A directed acyclic graph (DAG) is used to represent the complex dependencies among subtasks. For the graph G, $G = (V, E)$, the node set V represents the task set to be processed, $v \in V$ represents the divided subtask, E represents the directed edge set, represents the dependencies between the tasks, and $e_{uv} \in E$ represents the amount of transfer data between the task u and the task v, which represents when the task u is finished, e_{uv} data is transmitted to task v, task v can start execution only after task u is executed

Table 1 Parameter table for energy consumption and time model

Variable	Meaning	Unit
w_v	Workload of task v	CPU instructions
f_l	Calculation rate of mobile terminal	MIPS
f_e	Calculation rate of mobile edge server	MIPS
p_l	Power of the mobile terminal when performing tasks	W
p_i	Power of the mobile terminal when idle	W
p_s	Sending power of the mobile terminal	W
p_r	Receiving power of the mobile terminal	W
e_{uv}	Data transfer amount between task u and task v	Bits
B_s	Data sending rate	Bits/s
B_r	Data receiving rate	Bits/s
T_v^l	Time spent on the mobile terminal of task v	s
T_v^e	Time spent on the mobile edge server of task v	s
T_{uv}	Time spent on transferring data from task u to v	s
E_v^l	Energy consumed on the mobile terminal of task v	J
E_v^e	Energy consumed on the edge server of task v	J
E_{uv}	Energy consumed by transferring data from task u to v	J

3 Energy-saving strategy

3.1 Task decomposition

3.1.1 Coarse-grained task decomposition

A coarse-grained migration strategy typically uses the entire mobile terminal application as a one-migration task without decomposing and merging the terminal application [16]. The advantage of this strategy is that it is simple to implement and fast to schedule. For an application task with certain data transmission and computational requirements, the shortcoming of this scheduling strategy is that it often faces a dilemma: (1) if the task is scheduled to the edge, it faces a large amount of data

Table 2 Parameter table for minimization of energy consumption model

Variable	Meaning	Unit
E_{total}	Total energy consumption	W
T_{total}	Time consuming	s
$T_{\max}$	Task completion time threshold	s
T_v^b	Start moment of task v	s
T_v^{ex}	Executing time of task v	W
T_v^f	Finish moment of task v	W
E_v^{ex}	Executing energy consumption of task v	W
R_{uv}	Dependency between task u and task v	None
D_v	Execution location of task v	None

Table 3 Simulation parameters

Variable	Meaning	Value
f_l	Calculate speed of UE	300 MIPS
f_e	Calculate speed of MEC server	5000 MIPS
B_s	UE sending rate	2 Mbps
B_r	UE receiving rate	2 Mbps
p_l	UE working power	0.50 W
p_i	UE idle power	0.04 W
p_s	UE sending power	0.03 W
p_r	UE receiving power	0.01 W

transmission delay; (2) if putting the task on the local processing, the local computing power will cause a lot of computational overhead.

3.1.2 Fine-grained linear chain task decomposition

To improve the coarse-grained migration strategies, fine-grained linear chain task decomposition divides the mobile application into a set of subtasks [17]. These subtasks are executed in a linear topological order, and their input data are provided by their previous subtasks. The time limit of the entire mobile application is T.

Figure 2 presents a simple fine-grained linear task decomposition. In this model, the number of liner sequences of subtasks consisted by its mobile application is n. w_k denotes the load of computation, where $k = 1, 2, \ldots, n$. So, we can get the total load of the computational subtasks which are $\sum_{k=1}^{n} w_k$. In Fig. 2, α_k and β_k denote the number of input and output data of kth subtask respectively.

Figure 3 shows a simple fine-grained task decomposition topology [18]. In this topology, each task can be executed on local or edge side. $C(ME)$ denotes the energy consumption in the local, and $C(EE)$ denotes the energy consumption in the edge side. In the communication between the mobile terminal and edge equipment, the energy consumption for sending data is denoted as $C(SID)$; relatively, energy consumption for receiving data is $C(ROD)$.

3.1.3 Fine-grained directed acyclic graph task decomposition

We describe the tasks and their dependencies in a directed acyclic graph (DAG), as shown in Fig. 4 [19]. We name this task topology graph as task DAG $G=_nV, E_n$. In the task DAG, vertices $v \in V$ represent tasks, and the edges $e_{uv} \in E$ are used to represent dependencies between tasks. Thus, the e_{uv} represent the task v start execution must be after task u, where $u, v \in V$.

R (Rely), $R \in \{0, 1\}$ denotes the dependency relationship between each subtask:

$$R_{\mathrm{uv}} = \begin{cases} 1, e_{euv}\text{exist} \\ 0, \text{others} \end{cases} \quad (1)$$

3.2 Energy consumption and delay limit

In order to achieve tasks with minimum computational and communication energy consumption before the task delay threshold arrives, it is necessary to design a more optimized task scheduling strategy. Similar to the method of [20], this paper also adopts a method of weighing the time and energy consumption between the mobile terminal side and the MEC side and generates a scheduling decision.

For a brief description, we first list the symbol names used in this section and their meanings, as shown in Table 1.

Equation (2) gives the execution location of each subtask:

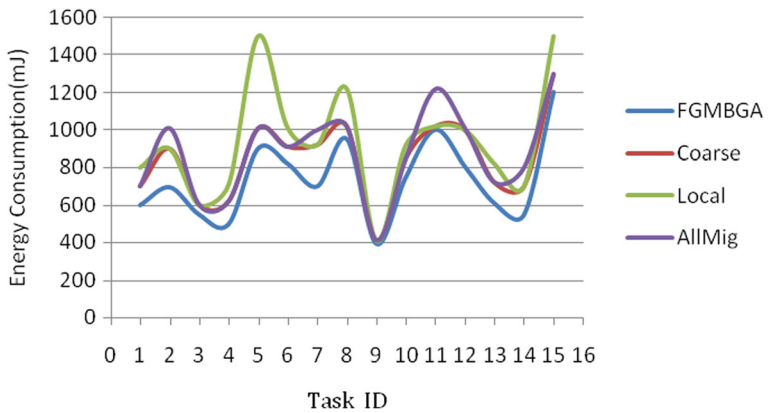

Fig. 5 Comparison of task energy consumption under four strategies. It can be observed that the results of the coarse-grained task migration strategy are always the same as the results of the local execution strategy or the total uninstallation to the MEC server strategy, because it is based on the entire mobile terminal application as an object for index measurement analysis

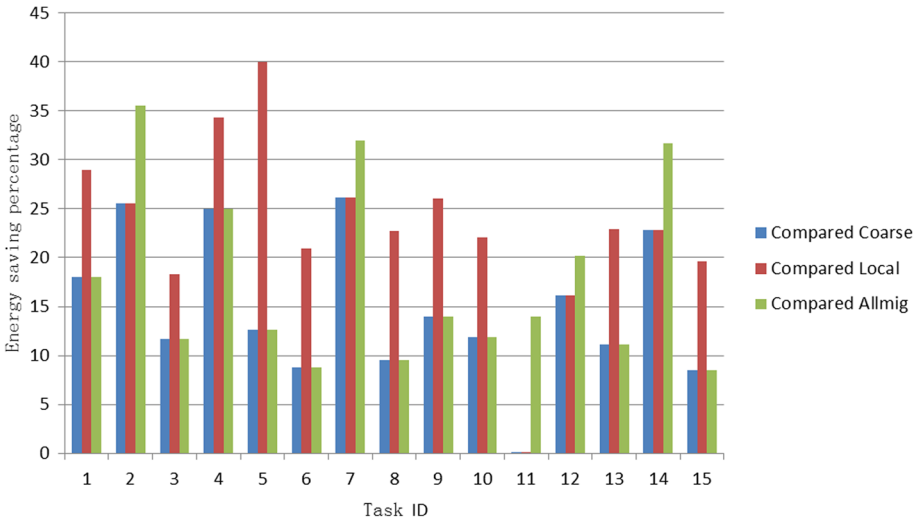

Fig. 6 The comparison of energy conservation between FGMBGA and the other three strategies. It shows the percentage of energy-conserving energy consumption compared with the other three strategies for the fine-grained task migration strategy based on genetic algorithms

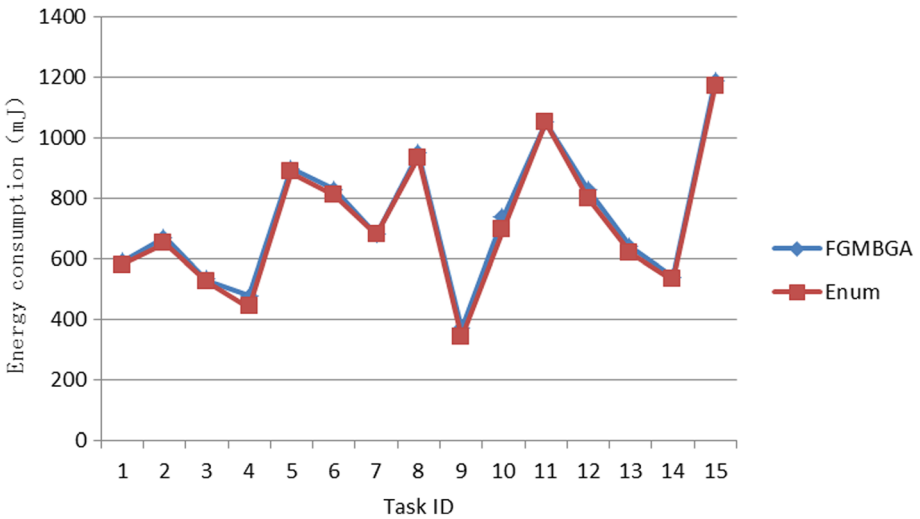

Fig. 7 Comparison of task energy conservation between FGMBGA and Enum. The figure compares the energy consumption of the fine-grained task migration strategy (FGMBGA) and the exhaustive method (Enum) based on the genetic algorithm. It can be observed that the line graphs of the Enum and FGMBGA algorithms basically overlap. Although the FGMBGA strategy does not traverse the entire solution space, the results obtained by the optimal solution and the exhaustive method are almost the same, which proves that the algorithm can reduce the time complexity and also ensure the accuracy

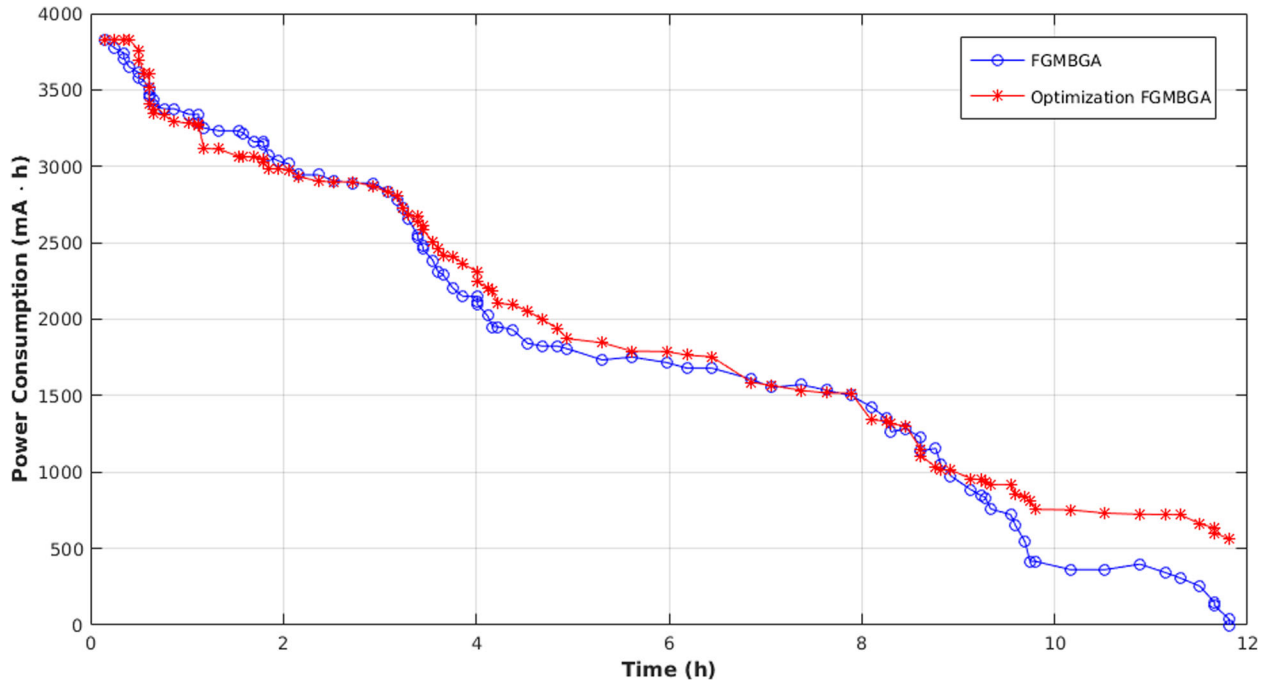

Fig. 8 Comparison of residual electricity trend between FGMBGA and optimization FGMBGA. As can be seen from the figure, when the residual power of mobile devices is reduced to a certain value, the optimization algorithm will no longer consume the battery energy of mobile devices, and the residual power tends to be flat, while the unmodified algorithm will continue to execute tasks on mobile devices according to the optimal solution obtained by genetic algorithm until the power of mobile devices is exhausted

$$D_v = \begin{cases} 1 , & u \text{ is excuted in MEC server} \\ 0 , & u \text{ is excuted in mobile equipment} \end{cases} \tag{2}$$

Execution time for task v in a mobile equipment side, denoted as T_v^l, is shown in Eq. (3):

$$T_v^l = \frac{w_v}{f_l} \tag{3}$$

Correspondently, the time of task v execution in edge equipment:

$$T_v^e = \frac{w_v}{f_e} \tag{4}$$

Thus, we get the energy consumption in local of the task v executing:

$$E_v^l = T_v^l p_l \tag{5}$$

Certainly, we must consider the energy consumption for other ideal processes in local:

$$E_v^e = T_v^e p_i \tag{6}$$

We give the delay for task u and v communications, T_{uv}:

$$T_{uv} = \frac{e_{uv}}{B_r} \tag{7}$$

And

$$C(SID)_{uv} = T_{uv} P_r \tag{8}$$

$$C(ROD)_{uv} = T_{uv} P_s \tag{9}$$

3.3 Energy consumption minimization

An easy-to-conceived idea is that the entire application is divided into smaller subtasks, appropriate migration strategies are invoked, migration destinations are determined, and task assignments are completed [21]. The so-called good migration strategy means that the terminal energy consumption can be effectively reduced, and the task completion time also needs to meet the user delay expectation constraint. Therefore, when building a model for minimizing energy consumption problems, you need to consider time constraints and thresholds for task completion time. To ensure that energy consumption is minimized and time-delayed, these two important factors in ensuring the quality of the user experience.

In this section, we establish a model of energy consumption minimization to characterize the above problems. For the sake of description, we first list the symbol

names used in the model and their meanings, as shown in Table 2.

The task execution time T_v^{ex} is a function about D_v, T_v^e, and T_v^l. It is shown in Eq. (10).

$$T_v^{ex} = (1-D_v)T_v^e + D_v T_v^l \tag{10}$$

Actually, T_v^b is the maximum of the dependency between u and v. We have

$$T_v^b = \max_{u \in V} R_{uv}\left(T_u^f + |D_u - D_v| T_{uv}\right) \tag{11}$$

Thus, we get:

$$T_v^f = T_v^b + T_v^{ex} \tag{12}$$

The execution energy of task v can be calculated as:

$$E_v^{\text{ex}} = (1-D_v)E_v^e + D_v E_v^l \tag{13}$$

Therefore, the energy consumption of the entire mobile application E_{total} is

$$\begin{aligned} E(\text{total}) = \underset{D(n)}{E} &= \sum_{v \in V}\left[(1-D_v)E_v^e + D_v E_v^l\right] \\ &+ \sum_{u,v \in V}\left[|D_u - D_v| E_{uv}\right] \quad \text{subject to} \\ &: \quad D(n) \\ &= [D_1, D_2, ..., D_n] \quad D_v \in \{0,1\}, \ \forall v \in V_{\text{offload}} D_v \\ &= 0, \ \forall v \in V_{\text{local}} \end{aligned} \tag{14}$$

Finally, we get the energy consumption minimization model:

$$\begin{aligned} \min E(\text{total}) = &\min_{D(n)} E \\ \text{subject to}: \quad & T_{\text{total}} \le T_{\max} \\ & D(n) = [D_1, D_2, ..., D_n] \\ & D_v \in \{0,1\}, \forall v \in V_{\text{offload}} \\ & D_v = 0, \forall v \in V_{\text{local}} \end{aligned} \tag{15}$$

where the $D(n)$ is a solution of our model. $D(n)$ is a vector, which contains n binary variables. "0" denotes process the task in the local, and "1" denotes in the cloud. Genetic algorithms can avoid the disadvantages of the time complexity of enumeration algorithms being too high.

3.4 Task migration strategy

Minimizing the energy consumption problem can be seen as a 0–1 programming problem with nonlinear constraints, which is an NP problem. In this paper, the genetic algorithm is used to calculate the approximate solution of the problem.

3.4.1 Initial

In order to minimize the energy consumption problem, the task migration variable in this paper adopts a binary coding scheme that takes into account the minimum coding length, such as the migration decomposition $D(n) = [D_1, D_2, ..., D_n]$ and each of its sub-elements.

3.4.2 Fitness function

In order to evaluate the corresponding chromosomes, the reciprocal of the total energy consumption is used as a fitness function to characterize the energy consumption and the necessity of migration for each application in the local device.

$$\begin{aligned} \text{fitness} &= \frac{1}{E(\text{total})} \\ &= \frac{1}{\sum_{v \in V}\left[(1-D_v)E_v^e + D_v E_v^l\right] + \sum_{u,v \in V}\left[|D_u - D_v| E_{uv}\right]} \end{aligned} \tag{16}$$

3.4.3 Design of crossover method

Crossover methods are often used for new chromosome construction, which has the disadvantage of easily causing the population to become "premature." In order to avoid this situation, in this paper, we adopt a random crossover method, which can ensure the diversity of the population and avoid the occurrence of "premature" conditions.

3.4.4 Design of variation method

In order to improve the diversity of the population and avoid "premature," we randomly modify the chromosome coding, so that the individual has the possibility of mutation. Commonly used mutation methods include uniform variation, Gaussian approximation, and the like. In this paper, we use a simple binary code variation method for basic position changes.

3.4.5 Genetic termination conditions

After the previous step, we obtained a new generation of chromosome populations. Whether the current solution is a suboptimal solution is evaluated, that is, whether the target of minimum energy consumption and delay requirement has been achieved. If a satisfactory solution is found, then the algorithm terminates; otherwise, the iterative search continues.

3.5 Optimization strategy

Although genetic algorithm can get the most energy-saving strategy of task migration, in real life, the calculation of migration also needs to consider the problem of mobile equipment residual power. If the power consumption of mobile phone is much larger than the energy consumed by task execution, the energy-saving strategy of fine-grained task

migration is followed; otherwise, the task will be executed at the MEC server to reduce the energy consumption of mobile phone CPU and cause downtime. The core algorithm of the strategy is:

$$D_v(\lambda) = (1-x)\lambda + x \tag{17}$$

Where E is task energy consumption and P is the residual power of mobile equipment, the relationship between task energy consumption E and residual power of mobile equipment P is as follows:

$$\lambda(E,P) = \begin{cases} 1, E > P \\ \dfrac{E}{P}, E \le P \end{cases} \tag{18}$$

where $\lambda(E,P) \in (0,1)$ and the energy consumption ratio has a linear relationship with the execution location. This algorithm is applied to the energy-saving strategy of fine-grained task migration based on genetic algorithm and judges the energy consumption of tasks and the residual power of mobile equipments. If the energy consumption and residual power of mobile equipments are relatively small, it follows the results of the energy-saving strategy of fine-grained task migration based on the genetic algorithm; if the energy consumption and residual power of mobile phones are too large, the optimal energy-saving strategy of migration is obtained through the above algorithm. When $\lim\lambda(E,P) = 1$, then $D_v(\lambda) = 1$. It means that the task should be handled at the MEC server.

4 Simulation and results

The simulation configuration of this algorithm is as follows: one mobile terminal device and one mobile edge computing (EMC) server for mobile terminal task migration service. Other experimental parameters are shown in Table 3.

Tasks generated in mobile terminals are composed of 20 related subtasks, and the computation and data transmission of each task are evenly distributed.

The initial settings of the genetic algorithm are as follows: the population size is 50, the mutation rate is 0.15, the crossover probability is 0.6, and the maximum number of iterations is 200. The termination condition of the algorithm is that the evolutionary speed of the fitness function of the individual with the best adaptability is less than 0.2% in five successive generations.

In the analysis of experimental results, we select several common migration algorithms compare with fine-grained migration based on generation algorithm (FGMBGA) which is the algorithm this paper proposed on their performance. By calculating the energy consumption under different strategies, we reflect the superiority and adaptability of the migration strategy. These migration strategies are coarse-grained migration strategy (Coarse), local execution strategy (Local), and EMC server execution strategy (AllMig). Figure 5 shows the energy consumption of mobile terminals under four strategies.

From the energy consumption comparison of the four migration strategies in Fig. 5, we can see that the Coarse strategy is consistent with the trend of FGMBGA strategy. Although the coarse strategy is the coarse-grained execution of tasks, the coarse execution strategy also measures and compares the task indicators and, after analysis, chooses the local server or EMC server as the task execution site. In addition, Local strategy and AllMig strategy only show excellent performance for the one subtask, but the overall performance is not good. Among the 15 randomly generated subtasks, it is worth noting that FGMBGA can save about 40% energy than Local strategy on the fifth subtask energy consumption.

Figure 6 shows the energy consumption comparison of FGMBGA with other three migration strategies. In this figure, we can see that compared with the Coarse strategy, the seventh subtask has the best effect, saving about 26% of energy; compared with the Local strategy, the best effect is on the fifth subtask, there are about 40% of energy saved; compared with AllMig strategy, the second subtask has the best effect, saving up to 35% of energy. Among the three energy consumption comparisons, the average energy consumption is Local, AllMig, and Coarse. Through the analysis of this result, we can draw the conclusion that migrating task from the mobile terminal to EMC server-side execution is the driving force of the times. The local execution is far slower than the EMC server-side execution in terms of computing power, computing time, and energy consumption.

Figure 7 is a comparison of energy consumption between FGMBGA and Enum. The main purpose of this group of experiments is to verify the accuracy of FGMBGA strategy. The Enum strategy traverses the whole population data, and the energy consumption results are shown in the blue line of Fig. 7; however, FGMBGA does not, but the optimal solution is almost the same as the Enum strategy (the red line in Fig. 7), so it can be concluded that the FGMBGA algorithm proposed in this paper is accurate and feasible.

Figure 8 compares the trend of residual power of mobile equipment between fine-grained task migration strategy (FGMBGA) and optimization algorithm. As can be seen from the figure, when the residual power of mobile devices is reduced to a certain value, the optimization algorithm will no longer consume the battery energy of mobile devices, and the residual power tends to be flat, while the unmodified algorithm will continue to execute tasks on mobile devices according to the optimal solution obtained by genetic algorithm until the power of mobile devices is exhausted. The experimental results show that the optimized genetic algorithm is more in line with the expectation of practical application. Migrate tasks to MEC servers to extend the standby time.

5 Conclusion

The traditional task migration strategy does not consider the multi-service node scenario, the multi-dependence scenario within the mobile terminal application, and the residual battery power of the mobile terminal. It cannot give full play to the advantages of the new characteristics of the mobile edge. This paper focuses on the mobile edge computing, from three aspects: task migration strategy, task execution location, and battery residual power. From the point of view, this paper puts forward the strategy of selecting the destination of task migration, the strategy of optimizing energy saving of task migration, and the strategy of optimizing energy saving of task migration when the battery power is too low. The simulation results fully demonstrate the energy saving and accuracy of FGMBGA strategy. At the same time, the superiority of the improved FGMBGA algorithm is demonstrated by comparing FGMBGA with the improved FGMBGA algorithm from experimental results. In the future, the mobile characteristics of mobile terminals can be considered, and the direction and trajectory of mobile terminals can be predicted. The migration strategy can be dynamically adjusted to adapt to more complex situations.

Abbreviations

5G: Fifth-generation mobile networks; CPU: A central processing unit; DAG: Directed acyclic graph; ETSI: European Telecommunication Standardization Association; FGMGA: Fine-grained migration based on GA; GA: Generation algorithm; IT: Information Technology; MEC: Mobile edge computing

Acknowledgements

The research presented in this paper was supported by the School of Computer Science and Engineering, Xi'an University of Technology, Xi'an.

Funding

This research work is supposed by the National Natural Science Founds of China (61602376, 61773313, 61602374, 61702411, U1334211), National Natural Science Founds of Shaanxi (2017JQ6020, 2016JQ6041), Key Research and Development Program of Shaanxi Province (2015KTZDGY0104, 2017ZDXM-GY-098), Science Technology Project of Shaanxi Education Department (16JK1573, 16JK1552), Ph.D. Research Startup Foundation of Xi'an University of Technology (112-256081504, 112-256081611), and College Research Funds of Xi'an University of Technology (112–451016007).

Authors' contributions

YW is the main author of this paper. He proposed the main idea and deduced the model of FGMGA. HZ is the writer of this paper. She completed the simulation and analyzed the result. XH improved the experiment and analyzed the result of the experiments. YK, WJ, and LZ gave some important suggestions for this paper. All authors read and approved the final manuscript.

Competing interests

The authors declare that they have no competing interests.

Publisher's Note

Received: 20 February 2019 Accepted: 3 May 2019
Published online: 27 May 2019

References

1. J. Gubbi, R. Buyya, S. Marusic, et al., Internet of things (IoT): a vision, architectural elements, and future directions. Futur. Gener. Comput. Syst. **29**(7), 1645–1660 (2013)
2. W. Shi, S. Dustdar, The promise of edge computing. Computer **49**(5), 78–81 (2016)
3. X. Ji, K. Huang, L. Jin, H. Tang, C. Liu, Z. Zhong, W. You, X. Xu, H. Zhao, J. Wu, M. Yi, Overview of 5G security technology. Sci. China (Info. Sci.) **61**(08), 107–131 (2018, v)
4. J. Wang, J. Peng, Y. Wei, et al., Adaptive application offloading decision and transmission scheduling for mobile cloud computing. China Communications(English) **14**(30), 169–181 (2017)
5. M. Whaiduzzaman, A. Naveed, A. Gani, MobiCoRE: mobile device based cloudlet resource enhancement for optimal task response. IEEE Trans. Serv. Comput. **11**(99), 144–154 (2018)
6. Di Pietro N, Merluzzi M, Strinati E C, et al. Resilient Design of 5G Mobile-Edge Computing over Intermittent mmWave Links. 2019
7. S. Li, D. Zhai, P. Du, et al., Energy-efficient task offloading, load balancing, and resource allocation in mobile edge computing enabled IoT networks. Sci. China Inf. Sci. **62**(2), 29307 (2019)
8. C. Yong, S. Jian, M. Cong-Cong, et al., Mobile cloud computing research progress and trends. Chin. J. Comput. **40**(2), 273–295 (2017)
9. Y. Mao, C. You, J. Zhang, et al., A survey on mobile edge computing: the communication perspective. IEEE Commun. Surv. Tutorials **19**(99), 1–1 (2017)
10. W. Shi, J. Cao, Q. Zhang, et al., Edge computing: vision and challenges. IEEE Internet Things J. **3**(5), 637–646 (2016)
11. W. Zhang, S. Tan, X. Chen, et al., A survey on decision making for task migration in mobile cloud environments. Pers. Ubiquit. Comput. **20**(3), 295–309 (2016)
12. W.J. Wang, H.S. He, M.A. Spetich, et al., Cloud computing and grid computing 360-degree compared. Grid Comput. Environ. Workshop Gce **5**, 1–10 (2014)
13. L. Chiar Aviglio, M. Mellia, et al., Minimizing ISP network energy cost: formulation and solutions. IEEE/ACM Trans. Networking **20**(2), 463–476 (2012)
14. K.W. Kwong, L. Gao, Z.L. Zhang, et al., On the feasibility and efficacy of protection routing in IP networks. IEEE /ACM Trans. Networking **19**(5), 1543–1556 (2011)
15. J. Wang, J. Peng, Y. Wei, et al., Adaptive application offloading decision and transmission scheduling for mobile cloud computing. China Commun. **14**(3), 169–181 (2017)
16. S.J. Marrink, A.A.H.D. Vries, A.E. Mark, Coarse grained model for semiquantitative lipid simulations. J. Phys. Chem. B **108**(2), 750–760 (2004)
17. R.M. Pathan, P. Voudouris, P. Stenstrom, Scheduling parallel real-time recurrent tasks on multicore platforms. IEEE Trans. Parallel Distrib. Syst. **29**(99), 1–1 (2018)
18. S. Yu, K. Ren, W. Lou, FDAC: toward fine-grained distributed data access control in wireless sensor networks. IEEE Trans. Parallel Distrib. Syst. **22**(4), 673–686 (2011)
19. S.E. Mahmoodi, R.N. Uma, K.P. Subbalakshmi, Optimal joint scheduling and cloud offloading for mobile applications. IEEE Trans. Cloud Comput., 1–1 (2016)
20. H. Zhang, Q. Zhang, X. Du, Toward vehicle-assisted cloud computing for smartphones. IEEE Trans. Veh. Technol. **64**(12), 5610–5618 (2015)
21. H. Ibrahim, R.O. Aburukba, K. El-Fakih, An integer linear programming model and adaptive genetic algorithm approach to minimize energy consumption of cloud computing data centers. Comput. Electr. Eng. **67**, 551–565 (2018)