A Security-aware Workflow Scheduling Strategy for Edge-cloud Computing used in UVA Delivery System

SHI Wenyu, ZHANG Rui*

School of Information and Engineering, Anhui Xinhua University, Hefei 230087, China
*Corresponding author’s e-mail: zhangrui@axhu.edu.cn

Abstract: With the rapid development of e-commerce, the “last mile” of logistics delivery has the problems of low efficiency and high cost. UVA delivery system can overcome this bottleneck whose system is a business process system which requires massive resources and real-time response. Edge-cloud computing integrates the massive processing power of cloud computing with the advantages of low latency and localization of edge computing. However, energy of UVAs is limited and UVAs are vulnerable to attacks. To address such issues, this paper proposes a secure model based on edge-cloud computing. Second, a new fitness function is proposed to evaluate the energy consumption and the security of task execution. Finally, a security-aware energy-efficient multi-source workflow task scheduling strategy with in response time constraint is proposed. The results show that our strategy can achieve the security requirement and has less energy consumption under the limitation of response time.

1. Introduction

Recently, with the development of e-commerce, there is a large demand for logistics industry. The traditional logistics has problems of traffic, parking, loading and offloading. The “last mile” distribution whose distance is less than 5% of the whole costs more than 50% of the total cost. Many companies attempted to use UAVs to solve these problems such as Amazon, Jingdong, Meituan, Wal-mart. But the use of UAV will cause some security problems, such as the responsibility for the lost of package and imposter of the receiver. Existing delivery systems are mostly based on cloud computing which uploads all the data to the cloud computing center for centralized storage and calculation. The round-trip transmission of data causes long transmission delay and large data movement leads to degradation or insufficiency of network bandwidth. In addition, cloud computing is also facing challenges such as users' unilateral trust to computing center. Edge computing [1] provides computing, storage and network resources to process data locally in edge nodes (such as routers, switches, proxy servers). The lightweight nature of edge servers makes it difficult to undertake computing tasks with large computational load. Edge computing industry alliance formally put forward a new distributed computing paradigm, that is, collaboration with cloud computing and edge computing (edge-cloud computing for short) [2]. Edge-cloud computing utilizes the powerful calculation ability of cloud computing and the advantage of distribution and real-time of edge computing which can satisfy the requirements of various distributed applications in the deployment environment.

Delivery system is a workflow system. The resource types to execute tasks in edge-cloud computing are more complex. UVAs usually have limited power supplied by batteries that will consume energy when executing tasks or uploading and downloading data. Therefore, optimization of energy consumption is one of the important issues. It is very important to coordinate the scheduling management of multi-layer computing resources to meet the QoS of users.
In the present researches, they mainly focus on the performance optimization of time, cost and energy consumption. In cloud computing, [3] explored the use of UAV swarms for delivery and used several warehouses as ground stations to monitor UAVs. The author built a UAV delivery platform and explored a dynamic and hybrid delivery method for UAV swarms. In edge computing, the execution of tasks is mainly determined by offloading method. In the research of edge computing security, there are many surveys, such as the detailed analysis and introduction of security threats and corresponding technologies in [4]. In [5], the idea of combining blockchain and edge computing was proposed, and several important aspects of such integration including motivation, framework, enabling functions and other challenges were determined after investigation and analysis of relevant contents. However, no feasible implementation scheme was proposed. In [6], firstly, relevant technological development of edge computing is summarized, and one of the challenges in edge computing - privacy trust and security guarantee problem is proposed. Then, a comprehensive trust is put forward for multi-constraint offloading scheduling algorithm and optimization scheme of edge computing is studied according to application requirement.

2. Motivating examples

This chapter firstly analyzed three scheduling schemes of task execution from the aspects of terminal energy consumption and task execution time. Then execution time, energy consumption and security required for a task executed on edge node with high security deployment and on a common edge node are compared.

In edge-cloud computing, including terminal devices, edge servers and cloud data centers, there are three corresponding task execution scheduling schemes as shown in Figure 1.

All tasks are performed locally (LE, Local Execution). The Execution capacity of terminals is limited, the energy consumption of terminals is the highest and the execution time is the longest. When the tasks are executed at the edge server(EE, Edge Execution), the terminal devices are always in waiting or transmitting state, so the energy consumption is minimal but the tasks cannot be completed within the response time. In a multi-layer scheduling (ME, Multi-layer Execution) (that is, the edge-cloud computing), the tasks with big workload and small data size will be executed on cloud servers, the tasks with small workload will be executed on terminals, and others executed on edge servers, so the execution time is short and the terminal energy consumption is low.

When the task is executed on terminals, the task is executed locally. When a task is scheduled to be executed on cloud servers, there is centralized management mechanism in the cloud. This paper mainly analyzes the security issue when a task is scheduled to the edge to execute. In Figure 2, in order to quantitative calculation, we give each edge server a security coefficient (the value range between 0 and 1) according to the success rate of task execution. When the terminals placed in different positions on this map, it is within the communication scope of a common edge server (EN) or a main node (MN) with the deployment of blockchain, so tasks can be scheduled to EN or MN. Within time constraint, considering the energy consumption and security, users will be more willing to schedule tasks to MN to execute leading to task execution conflict. Here, we assume that the terminal device is located at A, the workload of task is 1000KB, the amount of data uploaded is 500KB, and the amount of data downloaded is 300KB. The execution speed of EN and MN is 1.0GHz and 1.3GHz respectively, and data transmission bandwidth is 20Mbps. The power of the device in idle state, calculation state, transmission (sending or receiving) state is 0.03W, 0.7W and 0.1W respectively. When this task is executed on EN1 or MN, the terminal energy consumption and security coefficient are shown in Table 1. The results show...
that, for the same task, the terminal energy consumption on MN is lower, the execution time is shorter and the task execution is safer and more reliable.

| Execution device | Terminal energy consumption $E$ | Execution time $T$ | Security coefficient $C$ |
|------------------|---------------------------------|-------------------|------------------------|
| EN1              | 0.034J                          | 1.04s             | 0.6                    |
| MN               | 0.027J                          | 0.81s             | 1                      |

Fig. 2 Task scheduling scheme for security

3. Algorithm model
This section mainly designs the model used in task scheduling strategy.

3.1 A secure network model for edge-cloud computing
Due to the limited distributed resources and capabilities of edge facilities, it is difficult to provide security performance as the cloud server. In traditional solutions, some encryption algorithms are used to ensure the data security which led to massive computation especially for big data. In [7], a blockchain-based network model is proposed to solve such issue. Blockchain is a decentralized distributed electronic accounting system, which is based on a trusted and absolutely secure mode. The combination of blockchain and edge cloud computing can effectively solve the trust problem between nodes, improve the security of task execution and the operation efficiency of multiple devices. However, blockchain requires high performance such as storage, computation capacity and network bandwidth. We cannot deployment blockchain on all the nodes.

In edge-cloud computing, the security of edge severs should be improved. A secure network model is proposed in which several edge nodes with higher performance are choose to be the main node(MN) and others are the normal edge nodes(EN), as shown in Fig. 3.

Fig. 3 A secure network model for edge-cloud computing

Terminals are more willing to request resources from MN nodes within the communication range. When several terminals gathering around one MN node request resources, this MN cannot satisfy the resource requests at the same time under time constraint, and resource requests will conflict. At this
point, scheduling must be made to adjacent nodes or the cloud. Therefore, in the secure network model of edge-cloud computing, we need to select a reasonable algorithm to schedule tasks to solve the problem of task request conflict.

3.2 Energy consumption model

Energy consumption of terminals in the task scheduling process can be divided into two categories: (1) energy consumed by the terminal equipment when the task is executed on the terminals; (2) when the task is scheduled to be executed on the edge server or cloud server, the main energy consumption is generated by sending data, receiving data and waiting for task execution. Based on [8], an optimized energy consumption model is shown in Formula (1).

\[ E = f_1 \times P_{\text{edge}} \times \sum_{i=1}^{n} \frac{l_i}{f_{\text{edge}}} + f_2 \times (P_{\text{tr}} \times \frac{c_i}{R_1} + P_{\text{re}} \times \frac{d_i}{R_2} + P_{\text{idle}} \times \sum_{i=1}^{n} \frac{l_i}{f_c}) \]  

(1)

Where, \( P_{\text{edge}} \) denotes the power of terminals while executing the tasks, \( l_i \) denotes workload of task \( T_i \), \( f_{\text{edge}} \) denotes task execution speed of terminal device; \( P_{\text{tr}}, P_{\text{re}} \) and \( P_{\text{idle}} \) denote power of terminal device sending and receiving data and waiting time respectively. \( c_i \) and \( d_i \) denote data size that need to be uploaded and downloaded. \( R_1 \) and \( R_2 \) respectively denote data upload and download rates in edge-cloud computing. \( f_c \) is task execution speed of edge server or cloud server.

When \( f_1 = 1 \) and \( f_2 = 0 \), it indicates that the task is executed locally; When \( f_1 = 0 \) and \( f_2 = 1 \), terminal schedules the task to the edge or cloud server.

3.3 Security evaluation model for task scheduling

In the open and shared network environment, security risks are everywhere. The conceptual safety evaluation cannot give a reference value among the evaluation factors in a complex system. This paper presents a quantitative safety evaluation model, normalizes the safety of each equipment, and evaluates the advantages and disadvantages of scheduling scheme as a quality evaluation index of the task scheduling scheme. The success rate of task execution of a certain equipment can be got from monitoring for a period. And then the success rate of task execution should be normalized to get the security coefficient of each equipment. Here, the security coefficient of a device is defined as \( s_i \). For a certain task, it can only be scheduled once, so the task completion coefficient is \( s_i \), and the task scheduling security coefficient of the whole workflow is shown in Equation (2).

\[ s_{\text{sum}} = \sum_{i=1}^{n} s_i \]  

(2)

Where, \( s_i \) denotes the normalized value (that is, the safety coefficient) of task \( T_i \), whose value range is \((0,1]\). \( s_i = 1 \) when the task is executed locally or on the cloud server; When the task is executed on the edge server, \( 0 < s_i \leq 1 \) (in which, \( s_i = 1 \) denotes task is executed on MN). Therefore, for a task, the higher \( s_i \) is, the better; for the task scheduling security of a workflow, the larger \( s_{\text{sum}} \) is, the better.

4. A security-aware energy-efficient multi-layer workflow task scheduling strategy in edge-cloud computing(SEMSS)

This chapter presents a new fitness calculation method based on energy consumption model and secure task scheduling model to evaluate the energy consumption, security and time constraint, and combined with resource request conflict problem of task scheduling.

4.1 Fitness value

Fitness value can be used to evaluate the quality of scheduling scheme generated by the algorithm. According to the fitness value designed in [8], combined with the terminal energy consumption model and the security evaluation model in multi-layer environment, a new fitness calculation method is proposed for evaluating scheduling algorithm under time constraint, as shown in Formula (3):

\[ \text{fitness} = \left[ (f_1 \times E_{\text{sum}}) + \left( f_2 \times 10 \times E_{\text{sum}} \times \frac{T_{\text{sum}}}{T_c} \right) \right] \frac{1}{s_{\text{sum}}} \]  

(3)
Where, $E_{\text{sum}}$ denotes total energy consumption of terminals, $T_{\text{sum}}$ denotes total task execution time, and $T_c$ denotes time constraint value. The function consists of two parts: the first part takes terminal energy consumption as the target. When the task execution time can meet time constraint ($f_1 = 1, f_2 = 0$), energy consumption of terminal is the energy consumption of task execution; when task execution time exceeds time constraint ($f_1 = 0, f_2 = 1$), it is the product of the energy consumption of scheduling scheme and the response time constraint beyond the time constraint. The second part takes task safety as the target. The smaller the energy consumption of terminal equipment is, the better, while the larger the safety coefficient is, the better. So energy consumption and safety coefficient are inversely proportional while setting fitness value. It can be drawn that the smaller the fitness value in Formula (3) is, the better the scheduling scheme is.

4.2 The SEMSS strategy

The SEMSS strategy is used in the edge-cloud computing. In this scheduling strategy, edge servers are divided into two types: normal edge nodes ENs and main nodes MNs. PSO algorithm is used to update the scheduling scheme generated through iteration. When resource request conflicts are encountered in scheduling schemes, Min-Min algorithm [15] is used to solve them.

SEMSS mainly includes three parts: initialization, iteration and return the best task scheduling scheme as the final solution. Firstly, randomly generate particle swarm and set relevant parameters (lines 1-4). The iterative process is divided into outer loop and inner loop (lines 5-15), where outer loop is the control iteration (lines 5-15) and inner loop is the number of particles in the control group (lines 6-13). After the inner loop is completed, the scheduling scheme with the minimum fitness value is selected as the global optimal scheme (line 14). When the iteration terminates, the strategy returns the final global optimal scheduling scheme (line 19).

Algorithm: SEMSS

Input: Maximum iteration $Iter$, Task $T$, Time constraint $T_c$, Cloud server virtual machines VMs, Edge servers SNs, Main nodes MNs, Edge devices EDs.
Output: Optimal scheduling solution $g_{best}$

1: for $i = 1$ to $k$ do
2: generate the initial solution $p_i$ and particle velocity $v_i$;
3: set the initial solution $p_i$ as the initial optimal solution $p_{bestk}$;
4: end for
5: for $t = 1$ to $Iter$ do
6: for $i = 1$ to $k$ do
7: update the solution $p_i$ and particle velocity $v_i$;
8: execute the scheduling solution and Min-Min algorithm is used to solve the resource request conflict;
9: compute the task execution time $T_{sum}$ of $p_i$;
10: compute the terminal energy consumption $E_{sum}$ of $p_i$;
11: compute the security coefficient $s_{sum}$ of $p_i$;
12: compute the fitness value of $p_i$;
13: if (fitness($p_k$) < fitness($p_{bestk}$)) then
14: set the $p_k$ as the local optimal scheduling solution $p_{brstk}$;
15: end if
16: end for
17: choose the scheduling solution with the least fitness value as the global optimal scheduling solution $g_{best}$;
18: end for
19: return $g_{best}$
5. Evaluation
In this chapter, the experimental environment and relevant parameters are introduced, and then the simulation results are given.

5.1 Experimental settings
The simulation was run on a desktop computer (Intel core i7 3.6 GHz CPU, 16 GB) with MATLAB R2017b. Each workflow randomly generated using directed acyclic graph which contains 50-200 tasks, the task workload is 3000 - 8000 cycles (Mega cycles) within the scope to meet the random value of normal distribution, the data size of each task to meet the needs of normal distribution between 5 - 300 MB random value, the network bandwidth of LAN and WAN is 100 Mbps and 40 Mbps respectively. The sending, receiving, working and idle power of edge devices are 0.1w, 0.025w, 0.7w and 0.03w respectively, and the CPU's data processing capacity is 0.7GHz. The CPU processing speed of EN is 1.0GHz, the CPU processing speed of MN is 1.3GHz, and the CPU processing speed of the cloud server is 1.6GHz. We randomly deployed 2 cloud servers and 20 edge servers, and randomly generated 4 main nodes from the edge servers (MN). The time constraint given by the user is set from 20% to 100% of the average completion time of the task execution on the 1.4GHz CPU. The security coefficient $s_i$ of the task execution node is set as random value from (0.5, 1] (where the security coefficient of terminal device, cloud server and primary node MN is set as 1) [8].

5.2 Experiment results and analysis
This section analyzed the terminal energy consumption, task execution time, security coefficient and fitness value. The SEMSS strategy is compared with the following algorithms: the local task execution (not offloading scheduling, LE), task execution on the edge (EE), multi-layer scheduling without master node deployment (ME).

![Fig. 4 Terminal energy consumption](image)
![Fig. 5 Task execution time](image)
![Fig. 6 Security coefficient](image)
![Fig. 7 Fitness value](image)

The comparison of terminal energy consumption of the four scheduling strategies is shown in Figure 4. It can be drawn that LE scheduling strategy has the highest energy consumption. It is because that all tasks are executed on the terminals, which have the highest energy consumption. The gap of terminal energy consumption of the other three strategies is not big, but relatively higher in EE scheduling scheme.

The task execution time generated by the four scheduling strategies is shown in Figure 5. LE has the
highest task execution time in all the cases of different number of tasks. The reason is that the execution speed of terminal device is the lowest. SEMSS and ME are lower than EE when the number of tasks is 50, because of the multi-layer, parallel tasks can be scheduled to execute on the edge servers or cloud servers at the same time, but when the number of tasks increases to 200, task execution time of EE is less than ESS and ME, because part of the data in the SEMSS, ME task scheduling strategy was scheduled to the cloud server, so the uploading and downloading time is relatively long.

The comparison of security coefficients of the four scheduling strategies is shown in Figure 6. Among them, LE has the highest security coefficient value, because it is assumed in this paper that the task execution is safe on the terminal device \(s_i = 1\). SEMSS algorithm is obviously superior to the other two. When the number of tasks is 50, the security coefficient of SEMSS is higher than EE and ME by 11.6% and 13.5%.

The fitness values of the algorithms corresponding to the four scheduling strategies are shown in Figure 7. When the number of tasks changes from 50 to 200, the fitness values of scheduling schemes obtained by SEMSS are the lowest. In all cases, the fitness value of LE is the highest, because all the tasks are executed locally and the terminal energy consumption value is the highest.

6. Conclusion
This paper presented a network model in edge-cloud computing in which some nodes with high performance are deployed on the edge to improve the security of task execution. The energy consumption of multi-layer scheduling model is optimized, a new security evaluation model is proposed for the security issue of task scheduling, the fitness value is improved with the terminal energy consumption and security coefficient as evaluation indexes of scheduling algorithms, and a Security-aware Energy-efficient multi-layer workflow task Scheduling Strategy in edge-cloud computing (SEMSS) is proposed. The experiment results show that, under the time constraint, the fitness value, terminal energy consumption value and security coefficient of SEMSS are better than the other three scheduling strategies. In this paper, the encryption algorithms or blockchain technology has not been used in deployment of edge nodes. These problems will be solved in the future research work.

Acknowledgement:
This paper was supported by Quality Engineering Project of Anhui Province 2019zyrc111, 2018jyxm1108, 2018jyxm1107.

References:
[1] SHI W, CAO J, ZHANG Q, et al. Edge Computing: Vision and Challenges[J]. IEEE Internet of Things Journal, 2016, 3(5):637-646.
[2] White paper on edge computing and cloud computing collaboration[OL], http://www.ecconsortium.org/Uploads/file/20181129/1543447244547547.pdf, access on: 2019-02-27.
[3] Kuru K, Ansell D, Khan W, et al. Analysis and optimization of unmanned aerial vehicle swarms in logistics: An intelligent delivery platform[J]. IEEE Access, 2019, 7: 15804-15831.
[4] Vassilakis V, Chochliouras I P, Spiliopoulou A S, et al. Security Analysis of Mobile Edge Computing in Virtualized Small Cell Networks[J]. 2016, 475:653-665.
[5] Yang R, Yu F R, Si P, et al. Integrated Blockchain and Edge Computing Systems: A Survey, Some Research Issues and Challenges[J]. IEEE Communications Surveys & Tutorials, 2019, PP(99):1-25.
[6] Deng Xiaoheng, GUAN Peiyuan, WAN Zhiwen, et al. Intergrated Trust Based Resource Cooperation in Edge Computing[J]. Journal of Computer Research and Development, 2018, 55(3):449-477.
[7] Xu J, Wang S, Bhargava B K, et al. A Blockchain-enabled Trustless Crowd-Intelligence Ecosystem on Mobile Edge Computing[J]. IEEE Transactions on Industrial Informatics, 2019, PP(99):1-9.
[8] XU Jia, LI Xuejun, DING Ruimiao, et al. Energy efficient multi-resource computation offloading
strategy in mobile edge computing[J]. Computer Integrated Manufacturing System, 2019, 25(4): 954-961.