ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)

Volume 10 Issue 12 December 2023

Study on Parallel Offloading of Subtasks with Time Constraint and Cost Considerations

[1] Yen-Wen Chen, [2] Tsung-Te Hu

[1] National Central University, Taiwan
[2] National Central University, Taiwan
Corresponding Author Email: [1] ywchen@ce.ncu.edu.tw, [2] porterhu849@gmail.com

Abstract*— In the applications of Internet of Things (IoT), the computing and power resources of the end devices limit the deployment in real life especially for the real time services. Although the cloud server provides huge computing and storage resources, the transmission latency and its uncertainty due to the network congestion cause the difficulty in the provision of services. The edge server is then a possible way toward this problem. However, the time delay tolerance if some computing bound tasks may still not be satisfied due to the computing power of the offloaded server. Then it needs more than one server to execute its subtasks in parallel if the tasks can be partitioned into several independent subtasks. This paper studies the subtasks parallel offloading for local user equipment (UE), neighbor device to device (D2D) UE, and mobile edge computing (MEC) server by using the concept of 0/1 knapsack model. The main objective of the proposed subtask offloading for parallel processing (SuOPP) scheme is proposed to increase the acceptance ratio of task offload requests under the sustained delay. As the costs of different processing units may be different according to their processing speed, the proposed scheme also takes the cost into consideration. The simulation results illustrate that the proposed SuOPP scheme achieves higher acceptance ratio and lower cost when comparing to the other scheme. Although the average waiting time of the proposed scheme is a little higher than the compared scheme, it still satisfies the desired delay constraint.

Keywords*— Internet of Things, Mobile Edge Computing, Task Offloading, Knapsack Problem, Computing Resource.

I. INTRODUCTION

The applications of internet of things (IoT) increase dramatically, recently. These applications provide more convenient and safer environment for human life. Several new IoT services, such as real time navigation, driving safety, entertainment streaming, object matching and finding, industry automation, etc., need time-constraint processing and response. However, the IoT device has the limitation of computing power and energy resource and, therefore, affects the service deployment especially for the need of real time processing and response. Then to offload the task to other processing unit is one of the appropriate approaches to deal with this problem.

The cloud computing concept provides huge computing power and storage for the needs from the user equipment (UE). And the straightforward idea of the task offloading is to transfer the task to be executed by the cloud server and deliver the processing result to the IoT UE [1, 2]. Although the cloud server can provide larger, flexible and reliable computing and storage resources, there are two issues need to be concerned. One is the resource allocation of the server as many end UEs may request for offloading; and the other is the transmission delay as the cloud server may be far from the UE. Therefore, to offload task to the edge server, e.g. beside the base station, is one of the feasible approaches. In wireless networks, such as local WiFi, public fourth generation (4G) and fifth generation (5G) communication system, the mobile edge computing (MEC) servers are suitable to be located beside the base stations. And the UE can offload its task to the edge server beside the base station that the UE is connected to.

The edge server provides the computing resource for the required tasks with shorter and more predicable transmission overhead, however, some computing bound tasks may not be satisfied due to the long processing time. Then one of the applicable approaches is to utilize the concept of parallel processing. Thus, if the task can be divided into several independent subtasks, we can offload multiple subtasks to more than one edge server for processing simultaneously. The independent subtasks mean that those subtasks have no logical correlation and need not to be executed in sequence. For example, the matchings or recognitions of multiple objects within an image. It is noted that, as the number of edge servers may not be the same as the number of subtasks and the processing powers of edge servers may be different, there is no restriction for the assignment between subtasks and edge servers. That is the edge server may be allocated for zero, one or more than one subtask. The definite constraint is that the completion time of the whole task, i.e. the time of finishing all subtasks, shall be under the desired delay time requirement. Additionally, the costs (or the payment) of using the servers may be different by referring to their processing power. The offload shall tradeoff between the processing time and the required cost. And the main objective of this paper is to design the subtask offloading for parallel processing (SuOPP) algorithm to meet the desired delay constraint of the task while minimize the processing cost.

The remainder of the paper is organized as follows. Section 2 presents the background and related works of this

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)

Volume 10 Issue 12 December 2023

study. Section 3 describes the proposed parallel subtasks offloading scheme in detail. In Section 4, we evaluate the performance of the proposed scheme through exhaustive simulations. Finally, Section 5 concludes our works.

II. BACKGROUND AND RELATED WORKS

As mentioned in previous section, the cloud server provides huge computing and storage being shared by users and resource on demand capability to achieve flexible resource allocation services. Comparing to the cloud server, the edge server has much less computing power, however, it provides less transmission overhead and tends to be more suitable to be the target of the offloading for real time tasks. It is more practical to adaptively utilize the resource of cloud server and edge server in real environment. The hybrid cloud computing and edge computing architecture, as shown in Fig. 1, is more suitable for practical applications [2].

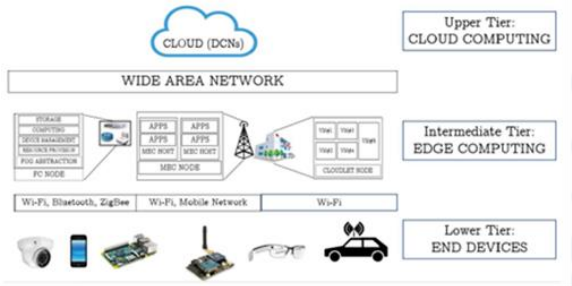


Fig. 1 The hybrid cloud and edge servers service architecture [2]

Recently, several studies investigated the issue of task offloading for better resource utilization and higher acceptance ratio of the offload requests. In [3], an overview of mobile edge computing was provided. The optimization of the computation offloading for an edge computing system was formulated and reviewed. And the collaborative scheduling scheme was illustrated for better resource allocation [4]. In [5], the authors deal with the offloading requests from multiple end devices that are located within the overlapping area of more than one base station. The time and energy costs are correlated considered for the arrangement of those offloading requests. In [6], the Load-Adaptive and Joint Resource Allocation (LAJRA) offload decision algorithm by considering the integrated arrangement of computing and communication resource for the offloading requests from IoT devices. And strategy of service migration for the offloaded tasks in vehicle moving environment was studied and analyzed in [7]. The decision of offloading to either the cloud server or edge server was studied in [8]. The authors proposed the layered orchestrated scheme to deal with this issue.

The knapsack algorithm is very suitable to deal with the optimization of multidimensional objects allocation problem [9]. And it has been applied to several applications [10, 11]. It is noted that the parallel processing approach was adopted for the to the dynamic programming algorithm of Knapsack problem [11]. In this paper, we assume that the edge servers, and neighbor end devices can be candidates for offloading with different processing powers and costs. And the subtask allocation problem is modeled to a multidimensional knapsack problem. Based on the basic concept of knapsack algorithm, the proposed algorithm allocates the subtasks to suitable server/device for parallel processing to meet the delay requirement of the task.

III. THE PROPOSED SUOPP SCHEME

The main concept of parallel processing of task offloading is that the task can be subdivided into subtasks and the subtasks can be allocated to more than one processing unit to be concurrently executed. The system model of the proposed scheme is shown in Fig. 2. The local user equipment (UE) is the device that needs to offload its task and the device-to-device (D2D) UEs are the accessible neighbors. The MEC server, D2D UE, and local UE are the possible allocated computing resource of the offloaded task.

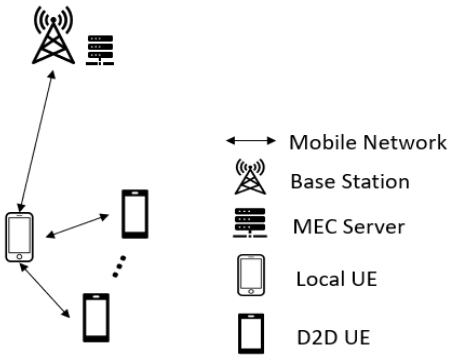


Fig. 2 The system model of the subtask offloading

In the system model, once the task is generated, the local UE determines whether to perform the offloading or not according to the required delay tolerance. If the task is not possible to be executed solely, it will issue the subtask offloading to the MEC server, D2D UEs. Fig. 3 illustrates the example comparison of the required computing times with/without parallel processing. Generally, the computing power of the MEC server is higher than the UE. The example shows that this task needs 85ms computing time when processed by the MEC server, and the total time is 88ms including 3ms for uplink and downlink transmission delay. And it requires 95ms processing time if processed by either local or D2D UE. However, it needs extra 1ms transmission time when offloading to the D2D UE.

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)

Volume 10 Issue 12 December 2023

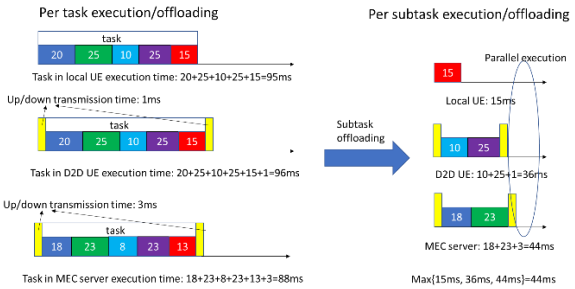


Fig.3 Execution time comparison

It is noted that the local UE can keep some subtasks to be executed by itself. On the left side of the figure, five subtasks are offloaded to the D2D UE, and MEC server, and one subtask is executed by itself for parallel processing. The required total time shall be determined by the longest time among them. In this example, the MEC server needs 44ms, which is the longest one, to complete the subtasks assigned to it, and, therefore, the required total time to finish the whole task is 44ms.

The purpose of the proposed scheme is to allocate subtasks, with different execution times and processing cost, to the MEC server, D2D UE, and local UE for parallel processing under the delay constraint of the whole task. As mentioned above that the all subtasks shall be completely before the required delay constraint. We first assume that the MEC server, D2D UE, and local UE be the knapsacks with different capacities according to their processing power. Then the subtasks allocation problem can be modeled as the knapsack problem as shown in Fig. 4.

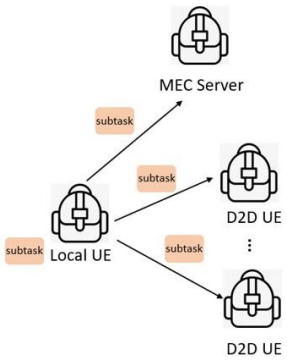


Fig. 4 The knapsack model of the subtask offloading

We assume that the task can be subdivided into k independent subtasks. The processing powers of local UE, D2D UE, and MEC server are denoted as f_{Local}、f_{D2D}、f_{MEC}, respectively. Then we can obtain the processing times required for the subtask i being processed by local UE, D2D UE, and MEC server as

$$T_{Local_i} = \frac{\varphi_{L_i}}{f_{Local}}$$

$$T_{D2D_i} = \frac{\varphi_{D_i}}{f_{D2D}}$$

$$T_{MEC_i} = \frac{\varphi_{M_i}}{f_{MEC}}$$

where φ_{L_i}, φ_{D_i}, and φ_{M_i} denote the required computing loads of subtask i being processed by local UE, D2D UE, and MEC server, respectively. And there may be more than one subtask to be offloaded one processing unit, therefore, the total processing time shall be the summation of the execution time of the subtasks in it as

$$T_{Local} = \sum_{i=1}^{N_{Local}} T_{Local_i}$$

$$T_{D2D} = \sum_{i=1}^{N_{D2D}} T_{D2D_i}$$

$$T_{MEC} = \sum_{i=1}^{N_{MEC}} T_{MEC_i}$$

In addition to the processing time, there will be transmission delay if the subtask is offloaded to the D2D UE or MEC server. The transmission delay can be derived as follows.

$$T_{D2D}^{transmission} = \frac{\sum_{i=1}^{N_{D2D}} d_i}{R_{Local \leftrightarrow D2D}}$$

$$T_{MEC}^{transmission} = \frac{\sum_{i=1}^{N_{MEC}} d_i}{R_{Local \leftrightarrow MEC}}$$

The delay times, including the processing time and transmission time, of the subtask to be offloaded to either the local UE, D2D UE, or MEC server are treated as the weights to be put into the three kinds of knapsacks, respectively. And as mentioned, the required cost to be executed by different processing units are different. Then the cost to be processed by the local UE, D2D UE, and MEC server, according to different processing time, are given as follows

$$P_{Local} = T_{Local} * \alpha$$
$$P_{D2D} = T_{D2D} * \beta$$
$$P_{MEC} = T_{MEC} * \gamma$$

The inverse of the above costs can be denoted as the values of the subtask to be allocated for the associated knapsack. Now the subtask (i.e. object) offloading problem is modeled to be the knapsack problem, however, the weights and values of the objects are not always the same, their weight and value depends on which knapsack (processing unit) to be offloaded. Then, according to above modeling, the procedure of the proposed SuOPP scheme is provided in Fig. 5.

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)

Volume 10 Issue 12 December 2023

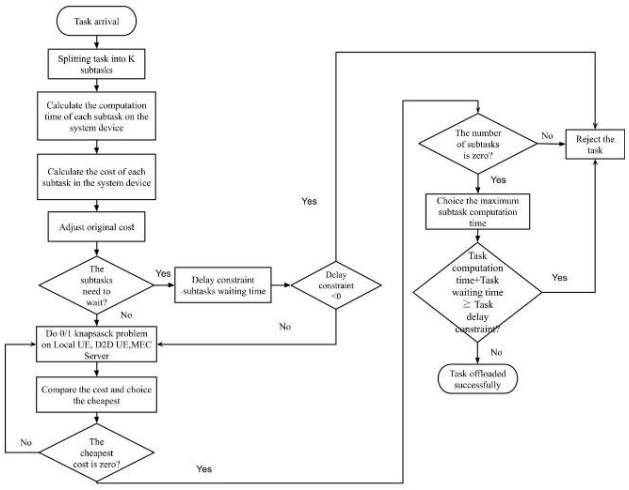


Fig. 5 The procedure of the proposed SuOPP scheme

IV. EXPERIMENTAL SIMULATIONS AND DISCUSSIONS

The performance of the proposed SuOPP scheme was conducted through the exhaustive simulations. The simulation environment consists of one MEC sever, which is located beside the base station, one local UE, and several D2D UEs. The parameters during the simulation is provided in the following Table 1. It is noted that the "U" listed in the table indicates the uniform distribution.

Table 1. Simulation parameters

Parameters	Values
Task Arrival Rate (Lambda)	Poisson Distribution 15,20,25 Task/Sec
Task Delay Constraint	U(100ms,150ms)、U(150ms,200ms) U(200ms,250ms)、U(250ms,300ms)
Number Of Subdivided Subtasks (K)	U(15,20)
Each Subtask Workload	U(100,200)M CPU Cycles
D2D UE Transmission Delay	1ms
MEC server Transmission Delay	3ms
Number Of Local UE	1
Number Of D2D UE	2、4、6
Number Of MEC Server	1
Local UE CPU	2.65GHz
D2D UE CPU	2.65GHz or 3.23GHz
MEC Server CPU	10GHz
Local UE Unit Price (α)	1
D2D UE Unit Price (β)	1.75
MEC Server Unit Price (γ)	1.25

The proposed SuOPP scheme considers the integrated delay time and cost for subtask allocation, and we compare its performance to the allocation that considers the delay time only scheme. It is noted that the task is accepted only all of its subdivided subtasks are successfully offloaded. Table 2 provided the results of the acceptance ratios for proposed scheme and compared scheme with lambda being 15 but different delay constraints and numbers of D2D UE.

Table 2. The acceptance ratio comparisons

Lambda = 15 and Delay Constraint (100ms-150ms)			
D2D UE no.	2	4	6
Integrated delay time and cost consideration (SuOPP)	0.073	0.16	0.233
Delay time consideration only	0.02	0.027	0.047
Lambda = 15 and Delay Constraint (150ms-200ms)			
D2D UE no.	2	4	6
Integrated delay time and cost consideration (SuOPP)	0.187	0.26	0.307
Delay time consideration only	0.127	0.147	0.16
Lambda = 15 and Delay Constraint (200ms-250ms)			
D2D UE no.	2	4	6
Integrated delay time and cost consideration (SuOPP)	0.26	0.3	0.333
Delay time consideration only	0.2	0.22	0.247
Lambda = 15 and Delay Constraint (250ms-300ms)			
D2D UE no.	2	4	6
Integrated delay time and cost consideration (SuOPP)	0.287	0.36	0.373
Delay time consideration only	0.233	0.253	0.273

The results illustrate that the acceptance ratio increases when the increase of either the number of D2D UE or the delay constraint as expected. The proposed SuOPP scheme demonstrates higher acceptance ratio than the compared delay time only scheme. The main reason is that the delay time consideration only scheme tends to firstly chose the MEC server for offloading because it has higher computing power and can complete the offloaded subtask in shorter execution time. For example, in a subtask needs 150M cycles to complete it, the execution time is only 18ms if processed by the MEC server, however, it requires 47~56ms for the D2D UE and local UE. Then the issued subtask will only seek for the offload to D2D and local UE when it can not be completed by the MEC server in time. However, most of them are also possible to be offloaded to the UE because of the lower computing power in UE. And the proposed SuOPP scheme takes the cost (price) into consideration. The MEC server has much higher than the D2D and local UE. The proposed scheme can effectively tradeoff and balance the traffic load between MEC server and UEs and, therefore, can achieve higher acceptance ratio.

As the cost is taken into consideration, the proposed SuOPP scheme can not only achieve the higher acceptance ratio but also less cost. Fig. 6 shows the cost of the proposed scheme for different numbers of D2D UE and the desired delay constraints. It is noted that the costs are shown in normalized representation. Thus, the cost is normalized as the ratio when comparing to the that of the delay cost consideration only scheme.

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)

Volume 10 Issue 12 December 2023

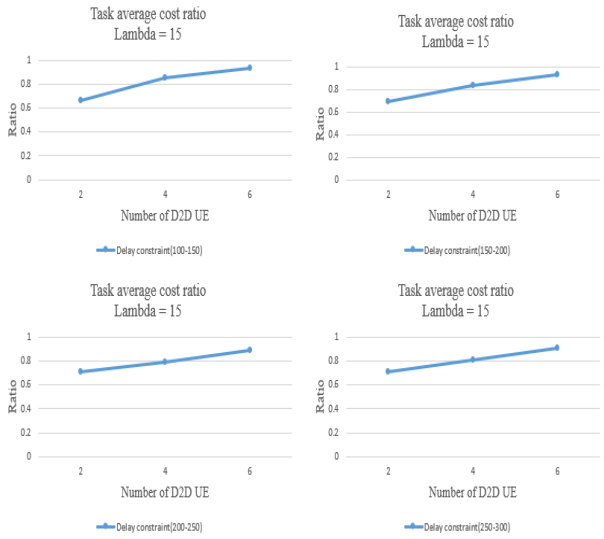


Fig. 6 The normalized costs of the proposed SuOPP scheme

The percentages of the accepted subtasks by MEC server, D2D UE, and local UE for different values of lambda are given in Table 3(a), (b), and (c), respectively.

Table 3. The percentages of the accepted subtasks

(a) Lambda=15

Lambda =15 delay constraint(100-150)	D2D UE no.	2	4	6
Integrated delay time and cost consideration	Local UE acceptance ratio	0.195	0.166	0.147
	MEC server acceptance ratio	0.621	0.526	0.476
	D2D UE acceptance ratio	0.183	0.307	0.376
Delay time consideration only	Local UE acceptance ratio	0.196	0.180	0.194
	MEC server acceptance ratio	0.565	0.607	0.593
	D2D UE acceptance ratio	0.239	0.213	0.213
Lambda =15 delay constraint(150-200)	D2D UE no.	2	4	6
Integrated delay time and cost consideration	Local UE acceptance ratio	0.208	0.181	0.149
	MEC server acceptance ratio	0.632	0.535	0.471
	D2D UE acceptance ratio	0.160	0.284	0.380
Delay time consideration only	Local UE acceptance ratio	0.206	0.170	0.182
	MEC server acceptance ratio	0.621	0.605	0.623
	D2D UE acceptance ratio	0.174	0.225	0.195
Lambda =15 delay constraint(200-250)	D2D UE no.	2	4	6
Integrated delay time and cost consideration	Local UE acceptance ratio	0.227	0.176	0.159
	MEC server acceptance ratio	0.633	0.531	0.476
	D2D UE acceptance ratio	0.140	0.293	0.365
Delay time consideration only	Local UE acceptance ratio	0.151	0.175	0.181
	MEC server acceptance ratio	0.699	0.664	0.680
	D2D UE acceptance ratio	0.149	0.161	0.138
Lambda =15 delay constraint(250-300)	D2D UE no.	2	4	6
Integrated delay time and cost consideration	Local UE acceptance ratio	0.235	0.202	0.184
	MEC server acceptance ratio	0.592	0.548	0.520
	D2D UE acceptance ratio	0.173	0.251	0.296
Delay time consideration only	Local UE acceptance ratio	0.066	0.087	0.080
	MEC server acceptance ratio	0.789	0.777	0.777
	D2D UE acceptance ratio	0.145	0.136	0.142

(b) Lambda=20

Lambda =20 delay constraint(100-150)	D2D UE no.	2	4	6
Integrated delay time and cost consideration	Local UE acceptance ratio	0.187	0.156	0.173
	MEC server acceptance ratio	0.608	0.503	0.457
	D2D UE acceptance ratio	0.205	0.341	0.370
Delay time consideration only	Local UE acceptance ratio	0.200	0.194	0.197
	MEC server acceptance ratio	0.600	0.581	0.590
	D2D UE acceptance ratio	0.200	0.226	0.213
Lambda =20 delay constraint(150-200)	D2D UE no.	2	4	6
Integrated delay time and cost consideration	Local UE acceptance ratio	0.207	0.170	0.156
	MEC server acceptance ratio	0.587	0.520	0.465
	D2D UE acceptance ratio	0.207	0.309	0.379
Delay time consideration only	Local UE acceptance ratio	0.219	0.197	0.205
	MEC server acceptance ratio	0.584	0.589	0.609
	D2D UE acceptance ratio	0.197	0.214	0.187
Lambda =20 delay constraint(200-250)	D2D UE no.	2	4	6
Integrated delay time and cost consideration	Local UE acceptance ratio	0.215	0.180	0.158
	MEC server acceptance ratio	0.605	0.525	0.470
	D2D UE acceptance ratio	0.181	0.294	0.373
Delay time consideration only	Local UE acceptance ratio	0.115	0.118	0.149
	MEC server acceptance ratio	0.706	0.722	0.720
	D2D UE acceptance ratio	0.179	0.161	0.131
Lambda =20 delay constraint(250-300)	D2D UE no.	2	4	6
Integrated delay time and cost consideration	Local UE acceptance ratio	0.214	0.180	0.169
	MEC server acceptance ratio	0.602	0.521	0.484
	D2D UE acceptance ratio	0.184	0.299	0.348
Delay time consideration only	Local UE acceptance ratio	0.117	0.098	0.093
	MEC server acceptance ratio	0.734	0.764	0.759
	D2D UE acceptance ratio	0.149	0.139	0.149

(c) Lambda=25

Lambda =25 delay constraint(100-150)	D2D UE no.	2	4	6
Integrated delay time and cost consideration	Local UE acceptance ratio	0.188	0.161	0.152
	MEC server acceptance ratio	0.600	0.513	0.469
	D2D UE acceptance ratio	0.213	0.326	0.385
Delay time consideration only	Local UE acceptance ratio	0.167	0.200	0.194
	MEC server acceptance ratio	0.633	0.600	0.597
	D2D UE acceptance ratio	0.200	0.200	0.210
Lambda =25 delay constraint(150-200)	D2D UE no.	2	4	6
Integrated delay time and cost consideration	Local UE acceptance ratio	0.212	0.175	0.150
	MEC server acceptance ratio	0.609	0.525	0.486
	D2D UE acceptance ratio	0.178	0.300	0.364
Delay time consideration only	Local UE acceptance ratio	0.201	0.159	0.187
	MEC server acceptance ratio	0.608	0.665	0.634
	D2D UE acceptance ratio	0.191	0.175	0.179
Lambda =25 delay constraint(200-250)	D2D UE no.	2	4	6
Integrated delay time and cost consideration	Local UE acceptance ratio	0.210	0.173	0.167
	MEC server acceptance ratio	0.603	0.526	0.470
	D2D UE acceptance ratio	0.187	0.302	0.363
Delay time consideration only	Local UE acceptance ratio	0.149	0.118	0.139
	MEC server acceptance ratio	0.688	0.727	0.696
	D2D UE acceptance ratio	0.163	0.155	0.165
Lambda =25 delay constraint(250-300)	D2D UE no.	2	4	6
Integrated delay time and cost consideration	Local UE acceptance ratio	0.212	0.176	0.164
	MEC server acceptance ratio	0.601	0.526	0.471
	D2D UE acceptance ratio	0.186	0.298	0.365
Delay time consideration only	Local UE acceptance ratio	0.092	0.101	0.105
	MEC server acceptance ratio	0.768	0.746	0.710
	D2D UE acceptance ratio	0.140	0.153	0.185

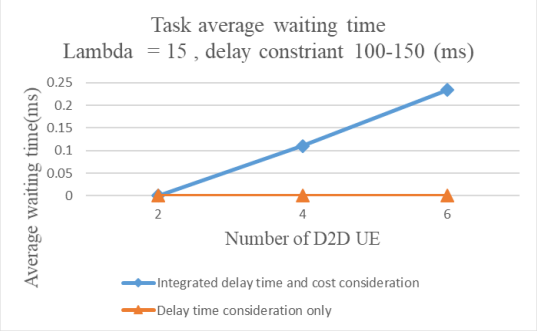
ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)

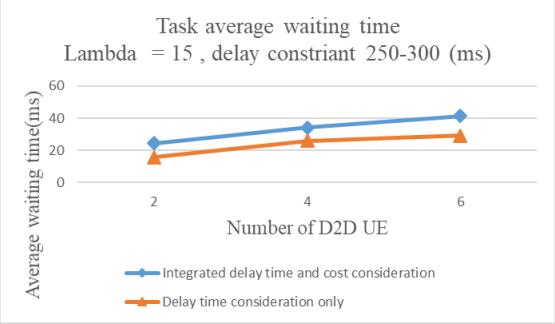
Volume 10 Issue 12 December 2023

The results show that the accepted percentage of the MEC server decrease as the number of D2D UE increase. And it also clearly shows that the delay time consideration only scheme has higher percentage to accept the subtasks for the MEC server and lower percentage for UE when comparing to the proposed integrated delay time and cost consideration scheme especially when the value of lambda is 25 as discussed in previous results.

Fig.7 (a) and (b) compare the average waiting times of short delay tolerance (100~150ms) and long delay tolerance (250~300ms), respectively, by given the lambda being 15. The results show that proposed scheme has higher average waiting time. The main reason is that the proposed scheme tends to balance the server load to achieve higher acceptance ratio, however, the waiting time still satisfies the desired delay constraint. The result also indicates the difference between these two schemes is very close when the required delay constraint is long. The reason is that the delay time consideration only scheme has higher possibility to accept the subtask especially for the D2D and local UE, due to the longer constraint as illustrated in Table2.



(a) The average waiting time comparison for short delay constraint



(b) The average waiting time comparison for long delay constraint

Fig. 7 The comparison of average waiting time for lambda=15

V. CONCLUSIONS

In this paper, we propose the SuOPP scheme for the offloading of computing bound tasks. The task is subdivided into multiple independent subtasks for offloading and parallel processing by. The proposed scheme properly models the multiple assignment issue into the knapsack problem. Then the proposed scheme extends the knapsack algorithm to determine the targets of the offloading subtasks. In practical environment, the processing cost of the above three processing units are generally different. So, both the delay constraint and the cost are considered in the proposed scheme. The performance of the proposed scheme is compared to the other scheme, which does not consider the cost issue, through extensive simulations. The results show that the proposed scheme achieves higher acceptance ratio under the desired task delay constraint. It is noted that the transmission delay of wireless communication is not deterministic and predicable. In this paper, this issue is not well considered and this is one of our research directions toward the real applications.

ACKNOWLEDGMENTS

This research was supported in part by the National Science and Technology Council (NSTC), Taiwan, under the grant of 112-2221-E-008-060.

REFERENCES

[1] M. De Donno, K. Tange and N. Dragoni, "Foundations and Evolution of Modern Computing Paradigms: Cloud, IoT, Edge, and Fog," in IEEE Access, 2019.

[2] K. Dolui and S. K. Datta, "Comparison of edge computing implementations: Fog computing, cloudlet and mobile edge computing," in Global Internet of Things Summit (GIoTS), 2017.

[3] K. Cao, Y. Liu, G. Meng and Q. Sun, "An Overview on Edge Computing Research," in IEEE Access, vol. 8, pp. 85714-85728, 2020.

[4] Chen, Shichao, Qijie Li, Mengchu Zhou, and Abdullah Abusorrah. "Recent Advances in Collaborative Scheduling of Computing Tasks in an Edge Computing Paradigm" Sensors 21, no. 3, 2021.

[5] Fanfan Wu; Xiuhua Li; Hui Li; Qilin Fan; Linquan Zhu," Energy-Time Efficient Task Offloading for Mobile Edge Computing in Hot-Spot Scenarios", in IEEE International Conference on Communications (ICC), 2021.

[6] Yen-Wen Chen, and BoHan Huang, “Design of Adaptive Offload Decision and Resource Allocation Algorithm for Critical and Non-critical Tasks,” Current Trends in Computer Sciences & Applications, vol. 2, issue 3, pp. 197-204, April, 2022.

[7] Yen-Wen Chen, XinYi Li, “Study of Time Constrained Task Offloads for Traveling Vehicles in Multi-Edge Computing Environment,” ISERD - 1454th International Conference on Vehicular, Mobile and Wearable Technology (ICVMWT), May, 2023.

[8] Chuan Sun; L. Hui; Xiuhua Li; Junhao We; Qingyu Xiongl; Xiaofei Wang; Victor C.M. Leun,"Task Offloading for End-Edge-Cloud Orchestrated Computing in Mobile Networks",in 2020 IEEE Wireless Communications and Networking Conference (WCNC), 2020.

[9] Valentina Cacchiani, Manuel Iori, Alberto Locatelli, Silvano Martello, “Knapsack problems — An overview of recent advances. Part II: Multiple, multidimensional, and quadratic knapsack problems,” Computers & Operations Research, Volume 143, July 2022.

[10] Simona Mancini, Michele Ciavotta, Carlo Meloni, “The Multiple Multidimensional Knapsack with Family-Split Penalties,” European Journal of Operational Research, Volume 289, Issue 3, 2021.

[11] Si Thu Thant Sin,"The Parallel Processing Approach to the Dynamic Programming Algorithm of Knapsack Problem" in 2021 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering ,2021