Optimum Selection of Mobile Edge Computing Hosts Based on Extended Balas-Geoffrion Additive Algorithm

Shanmuganathan Thananjeyan, Chien Aun Chan, Elaine Wong, and Ampalavanapillai Nirmalathas
Department of Electrical and Electronic Engineering, University of Melbourne, Australia.
(t.shanmuganathan@student.unimelb.edu.au)

Abstract—Multi-access edge computing (MEC) is emerging as a solution to serve offloaded tasks from mobile devices that are computing intensive and have very low latency and high bandwidth requirements. Since compute resources are limited at the MEC hosts, collaboration among hosts could enhance the capabilities of sharing limited resources while minimizing the cost of such hosts. However, the selection of optimal hosts to instantiate the user applications is a major challenge when considering the total service provisioning cost. In this paper, we formulate the MEC hosts selection problem as a binary integer problem with the objective to minimize the total cost of providing the offloading services. We extend the Balas-Geoffrion algorithm to solve the special case of binary programming problems similar to MEC host selection problem. The time complexity of the MEC host selection problem is therefore minimized. We show that our modified algorithm outperforms Balas-Geoffrion algorithm in the number of iterations required to reach the optimal solution. Then we conduct an extensive simulation to show that the overall quality-of-service of the MEC system is improved by the MEC hosts collaborations in a limited bandwidth scenario by up to 13%. However, the tradeoff is an increase in the cost of provisioning the services.

Keywords- multi-access edge computing; multi-access edge hosts collaborations; multi-access host selection; Balas additive algorithm

I. INTRODUCTION

Multi-access edge computing (MEC) is the next paradigm of mobile edge architecture in the fifth generation (5G) mobile communications to support future mobile applications. New and emerging applications such as virtual reality, mixed reality applications require high computing resources and very low latency. As smartphone devices have constraint in local computing resources and battery capacity, offloading computing intensive tasks to MEC system is considered as a promising solution. MEC was originally proposed as a three-tier architecture in which MEC hosts that are located between the cloud datacenters and the end users could provide edge computing services to the end users [1]. Depending on the application requirements, offloaded tasks could be either executed at the MEC host or the cloud data centers. Offloaded tasks from mobile users with stringent requirements could be executed in the MEC host and offloading tasks with flexible requirements could be executed at the cloud datacenters. In this paper, we focus on applications with stringent requirements in terms of computation, latency and network bandwidth.

MEC hosts can be deployed at a radio node, or at the edge of the core network, or at an aggregation point that is between the radio node and the edge of core network [1]. In order to maximize the utility of the MEC hosts with limited resources, collaboration among MEC hosts is proposed in [2]. Consequently, resource allocations within the MEC infrastructure, which comprises of multiple MEC hosts in multiple network locations, becomes a major challenge.

When an application in a mobile device sends an offloading request, this request will be processed by a MEC orchestrator to decide an appropriate MEC host to accept the request. Then a "*user application*" will be instantiated at the selected MEC host in response to the offloading request of the device application. However, each user application could have different requirements. Thus, selecting the ideal MEC hosts to instantiate the user application by abiding user application's rules and requirements such as the required resources, latency, connectivity and mobility, is a critical challenge for the MEC service providers. Solving this challenge is known as the *MEC hosts selection problem*. Limitations of resources (bandwidth and computing) and the associated usage costs should be jointly considered in solving the MEC hosts selection problem in order to minimize the overall network costs. However, to the best of our knowledge, *consideration of the cost of provisioning the MEC offloading services in the MEC hosts selection problem has yet to be fully investigated*. Here, we address the key question of how to minimize the total cost incurred by the MEC service providers in selecting the ideal MEC hosts to instantiate the user application in a collaborative environment without compromising latency requirements. Therefore, the contributions of this paper are as follows.

- We first formulate the MEC hosts selection problem as a 1-0 integer program (binary programming) problem to minimize the total cost of provisioning the offloading services at the MEC host;
- We then extend the use of Balas-Geoffrion additive algorithm to find the optimal solution for the special case of binary programming problems similar to the MEC hosts selection problem. Our modifications are as follows:
    - We modified the strategy to select the free variable, which has the minimum coefficient in the objective function.

  - We simplify the algorithm described in [3] to omit the tests that are not relevant to the context of this special case problems;

- Our proposed algorithm minimizes the computation time complexity since only the addition and subtraction operations are employed to find the optimal solution;

- We show that the number of iterations to find the optimal solution in our extedned algorithm outperforms the original Balas-Geoffrion algorithm using simulations;

- We then show the tradeoff between the servicing costs and the number of tasks rejected in the collaborative and independent methods respectively.

The remaining sections are organized as follows. Section II discusses the related work in MEC host selection. Section III discusses the methodology and our modifications on the Balas-Geoffrion additive algorithm. Then, Section IV discusses the simulation setup and results that verify the effectiveness of the optimization and the MEC hosts collaborations. Finally, Section V concludes the paper with a summary of the insights gained from our work.

II. RELATED WORK

Application placement algorithms related works can be found in various domains such as content distribution or cloud server management. In [4], the authors developed algorithms for resource allocations in a geographically distributed cloud environment. The problem is formulated as finding the nodes with available resources to minimize the latency between the selected nodes. They proposed an algorithm based on subgraph selection of the hierarchical network topology inside the datacenters.

In [5], the placement problem is studied in the context of content delivery networks. They design heuristics that replicate and deploy objects on selected nodes so that the cost is minimized. The cost is calculated based on the hop count. They concluded that the greedy approach can result in a performance that is close to the optimal. A placement method is applied to solve the problem of distributing workloads to multiple computing nodes in [6].

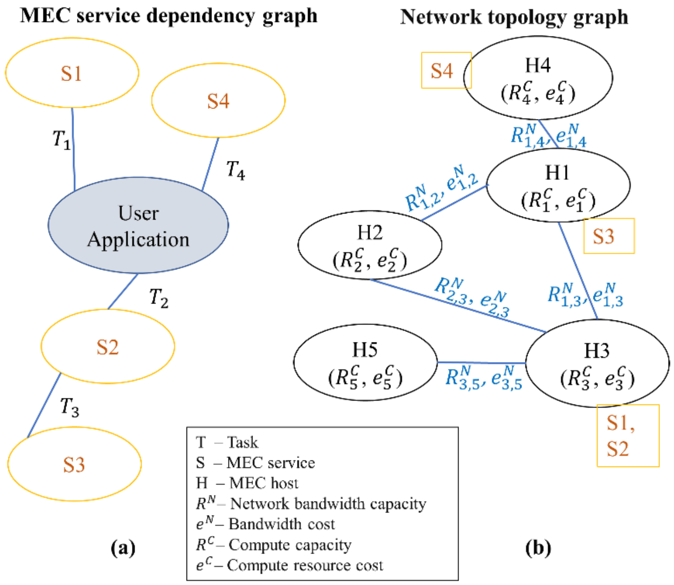


Figure 1(a) MEC service dependency graph of a user application (b) network topology graph of the MEC system

Some existing works on application placement and scheduling in MECs have considered applications with two components, one running on the cloud (which can either be by the MEC or core cloud) and the other running on the mobile device [7] [8]. Another body of existing work usually involves only two physical computing entities (i.e., the mobile device and the cloud) [9] [10]. Multi-component applications that can be deployed across one or multiple levels of MECs and core cloud(s) have not been considered [11], whereas such applications widely exist in practice because MEC hosts can be located at different hierarchical levels of the network [12]. A task placement algorithm of an edge computing orchestrator is also proposed in [13] that performs replication and placement of application components in a telecom-driven application-hosting infrastructure. Pre-computed shortest paths heuristics algorithm, which runs in polynomial time, is highlighted as the only solution that comes very close to optimality in polynomial time.

Collaborative service placement for mobile edge computing application is considered in [14] with the aim of minimizing the traffic load caused by service request forwarding. Optimal placement of virtual machine replica copies to minimize the average response time is studied in [15] and a hysteric placement algorithm is proposed. The Nova scheduler of OpenStack [16] is an existing application placement architecture that selects suitable computing nodes to initiate the virtual machines. However, it is explicitly disconnected from the networking component and does not consider requirements such as latency, connectivity and mobility that originated from the application providers [13].

III. METHODOLOGY

A. Computation task offloading

Computation offloading requirements: The MEC orchestrator is responsible to instantiate the user application in the most suitable MEC host(s) in the MEC system in response to the request from the device. The MEC hosts selections should satisfy the rules and requirements of the request such as deployment mode, specific hardware requirement, required resources, latency, connectivity and mobility requirements. Once the user application is instantiated in the MEC hosts, device application collaborates with user application to offload the whole or part of the computation tasks.

Costs: On the other hand, hosting offloading services to the customers incur costs to the MEC service providers. Computing resources and network bandwidth resource usages may have separate cost structures. Thus, from the MEC service provider's point of view, the objective is to minimize the total cost incurred by the MEC system when provisioning the offloading services while satisfying the service requirements of the MEC application.

These MEC service dependencies of a user application can be represented in a graph as shown in the Fig. 1 (a). Vertices of the graph represent the MEC services and the edges represent the task offloading requests within the services. Each MEC service request from the user application can be considered as a computational task offloading request from the user application. Computation intensive tasks (T) can be represented by the size of the offloading data (q^{N}), task deadline (τ^{MAX}) and the number of CPU cycles required to complete the task (q^C), i.e., $T \triangleq (q^N, \tau^{MAX}, q^C)$.

On the other hand, MEC hosts deployment can be represented in MEC network topology graph as shown in Fig. 1(b). Vertices of the graph represent the MEC hosts and the MEC host capacity in terms of compute resources. Edge of the graph indicates the network capacity in terms of bandwidth resources. A complete topology graph of the MEC system can be generated in which there is a direct communication link between all pairs of MEC hosts. It should be noted that both compute and bandwidth resource usages are associated with different costs in the complete graph.

B. Minimizing the total cost in MEC host selection

Let MEC hosts in the MEC system be represented in a set $H = \{1,2, \ldots h, \ldots\}$. Let I ($I \subseteq H$) be the filtered subset of the MEC hosts that could support the rules and requirements of the MEC applications such as special hardware requirement. Different MEC services might be required by the user application. Let $J (J \subseteq H)$ be the filtered subset of MEC hosts in which the required services are hosted. Let $x_{i,j}^N$ be the maximum amount of network bandwidth that can be allocated to the user application during the data transmission to be used between the MEC hosts i and j. This network bandwidth allocation is based on the network resource allocation policy of the MEC service providers and the available bandwidth. For instance, a maximum of 50% of the remaining bandwidth can be allocated to the incoming resource request. Thus, the minimum network latency is given by $\tau_{i,j}^N = \frac{q_{i,j}^N}{x_{i,j}^N}$, where $q_{i,j}^N$ is the data that needs to be transmitted between the MEC hosts. Further, the data transmission cost is $e_{i,j}^N * q_{i,j}^N$, where $e_{i,j}^N$ is the cost of transferring a unit of data between MEC hosts.

Let x_i^C be the maximum computing resource that can be allocated at the MEC host during the task execution. Noted that the computing resource allocation also depends on the MEC service provider's resource allocation policy and the resource availability at the MEC host with limited resources. Thus, the minimum computing latency in the servicing MEC host is $\tau_i^C = \frac{q_i^C}{x_i^C}$, where q_i^C is the required computing resources. The computing cost is thus $e_i^C * x_i^C$, where e_i^C is the unit allocation cost of the computing resources at MEC host i. The total service cost is the sum of the data transmission costs and computing costs, i.e., $e_i^C * x_i^C + \sum_{j\in J}(e_{i,j}^N * q_{i,j}^N + e_j^C * x_j^C)$. We assume each service are independent and can be consumed in parallel.

The objective of the MEC service providers is to minimize the total cost incurred by provisioning the offloading services while maintaining the quality-of-service (QoS):

$$\min \{ \sum_{i\in I} \alpha_i \, (e_i^C * x_i^C + \sum_{j\in J}(e_{i,j}^N * q_{i,j}^N + e_j^C * x_j^C)) \} \quad (1)$$

Subject to;

$$\alpha_i(\tau_i^C + \max\{\tau_j^C + \tau_{i,j}^N\}) \le \tau^{MAX}; \forall\, i \in I,\ \forall\, \mathrm{j} \in \mathrm{J} \quad (2)$$

$$\sum_{i\in I} \alpha_i \ge m \text{ and } \alpha_i \in \{0,1\} \quad (3)$$

where $e_i^C \ge 0$, $e_{i,j}^N \ge 0$ $e_j^C \ge 0$, $x_i^C \ge 0$, $\mathrm{q}_{\mathrm{i,j}}^{\mathrm{N}} \ge 0$, $x_j^C \ge 0$ and α_i is a binary variable. If $\alpha_i = 1$, the MEC host is selected for instantiating the user application, otherwise it is not selected. In addition to the latency constraints in Eq.(2), we have another constraint in Eq. (3): at least m MEC hosts shall be selected to instantiate the user application as per the deployment mode requirement of the application. Eq. (3) can be written as

$$-\sum_{i\in I} \alpha_i \le -m \text{ and } \alpha_i \in \{0,1\} \quad (4)$$

The optimal (feasible) solution needs to satisfy Eqs. (1) (2) and (4). We can represent the above minimization problem in the standard form as follows:

$$\min\{ \sum_{i\in I} \alpha_i * z_i \} \quad (5)$$

Subject to;

$$\sum \alpha_i * y_{i,k} + \beta_k = \gamma_k \,,\ k = 1,2 \quad (6)$$

where $z_i = e_i^C * x_i^C + \sum_{j\in J}(e_{i,j}^N * q_{i,j}^N + e_j^C * x_j^C)$ and slack variable of the constraint k is $\beta_k (\ge 0)$. Objective cost function is derived using cost functions of network and computing resources. Similarly, latencies are derived from the resource limitations in the network and computing resources. As all the cost components are non-negative, the total cost is also non-negative ($z_i \ge 0$).

We consider Eqs. (1), (2) and (4) as MEC hosts selection problem. The problem is dual feasible as $z_i \ge 0$. Further, by analyzing the problem, we have only one constraint with all nonpositive constant (-1) coefficients as in Eq. (4) and other with all nonnegative coefficient as in Eq. (2). In other words, it is a minimization problem with an upper bound constraints with all positive coefficients and all negative constant coefficients.

C. Balas-Geoffrion additive algorithm

It is important to note that the solution of MEC hosts selection problem will be a 0-1 integer (binary) programming. Binary programming is NP-complete and is one of Karp's 21 NP-complete problems [17]. Techniques available for solving the 0-1 integer programming problem include algorithms of Glass, Balas, Glover, Lawler and Bell, Geoffrion, Lemke and Spielberg etc. as summarized in [3]. These additive algorithms are enumerative and developed for solving 0-1 binary programming problems. The general idea of additive algorithm is to enumerate though some of all 2n possible solutions of a problem explicitly to find the best solution.

Bala's additive algorithm [18] with some modifications can be applied to solve the MEC host selection problem as it is dual feasible. The approach of Bala's algorithm that makes it efficient is that only some solutions are selected for explicit enumeration. Geoffrion reformulated the additive algorithm by reducing the spatial complexity (storage) to improve the efficiency of the Balas algorithm [19]. The only operations required under the algorithm are additions and subtractions.

The computation complexity of addition and subtractions is $\Theta(n)$ whereas the computational complexity of multiplication and division is $\Theta(n^2)$. This shows the advantage of applying additive algorithm in terms of computational complexity in solving the MEC selection problem. Another advantage of the algorithm is that it provides near optimal solution, even if the calculations stop before all the possible solutions are enumerated. [20].

D. *Extended Balas-Geoffrion additive algorithm*

Even though, Balas-Geoffrion algorithm can be applied to a general binary programming problem, it is not efficient in solving binary problems with upper bound constraints with all positive coefficients and all negative constant coefficients. **We modified the strategy of selecting free variables and fathoming tests of Balas-Geoffrion algorithm to improve the efficiency of the algorithm for the above-mentioned special case of binary programming problems**. Bala's strategy was to choose the free variables, which would then result in the least infeasibility. As we have an upper bound constraint with all negative constant coefficients, least infeasibility calculations will end up listing all the free variables and have to select one free variable randomly. Thus, we modified the strategy to select the free variables, which have the minimum coefficient in the objective function in order to guarantee the optimality of the problem.

Balas developed four tests as described in [3] to validate whether the given partial solution is fathomed or not. Based on the context of our problem, we omit Tests 1 and 3 in [3]. Because all the coefficients in each constraint are either positive or negative, there is no nonnegative coefficient for a free variable in all the constraints in Test 1 and hence should be omitted. Similarly, in Test 3, if the slack variable of a partial solution is negative, it should be from one of the upper bound constraints with all positive coefficients. Thus, there is no way to improve it by converting it to be positive by assigning 1 to any free variables in that constraint.

If the algorithm terminates with the feasible solutions, then the computation offloading request will be accepted, otherwise, the offloading request will be rejected. We define the total requests accepted in the MEC system as a metric of performance measurement to compare the collaborative MEC hosts method and non-collaborative (independent) MEC hosts method.

IV. SIMULATION AND RESULTS

A. *Simulation set up*

In order to evaluate the performance of MEC hosts collaborations, we simulated the MEC hosts selection problem involving an urban area of 2 km × 2 km served by a mobile wireless network in which radio nodes are spaced 200 m apart. We assumed a total of 1,000 users, each with a mobile device moving in vehicles for an hour, i.e., 3,600 seconds during the morning peak hour rush. We used the correlated mobility model in [21] to produce the users' trajectories of morning peak hour in which users start from the outer suburbs of the city and then move towards the central business district. Different types of servers are deployed in the MEC hosts considering energy efficiency as in our previous work [22]. Electricity cost is a major operational cost for MEC host servers. The mean electricity cost is USD 78 per MWh as in [23]. We considered the power consumption of servers to vary with server utilization.

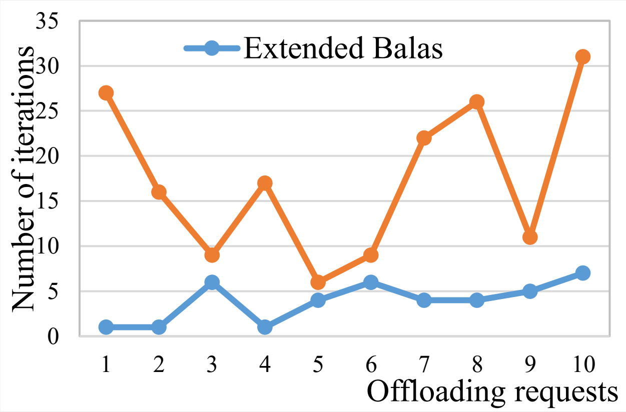


Figure 2 Number of iterations required to find the optimal solution for Balas and modified Balas algorithms

Further, we assumed different link capacities with different costs (1 Gbps, USD 0.05/ GB), 400 Mbps, USD 7/GB), (100 Mbps USD 0.09/GB) as in [23]. Link capacities are allocated according to the geographical distances between the MEC hosts. The amount of CPU resources distributed over the MEC hosts are based on utilitarian resource distribution algorithm as in [21].

In our previous work, we assumed that the MEC hosts are independent [21]. Thus, when a mobile device sends an offloading request, if the serving MEC host does not satisfy the requirements, the offloading request is rejected, otherwise the request is accepted. In this paper, we introduce MEC hosts collaboration in which user application can be instantiated in any suitable MEC hosts, not necessarily the serving MEC host.

As mentioned in Section III, Balas-Geoffrion algorithm is not efficient in solving MEC host selection problem as shown in Fig. 2. Because of the constant negative coefficients, Balas-Geoffrion selects a random value as the next free variable to iterate. Our modified algorithm outperforms Balas-Geoffrion algorithm in number of iterations required to select the optimal solution in MEC host selection problem.

B. *Collaboration of MEC hosts*

Figure 3 shows the benefit of MEC hosts collaboration method. Fig. 3(a) shows the improvement in the total number of accepted requests compared to the independent MEC hosts method. All the offloading requests during the middle of the morning peak hours are accepted. During the initial stages and the later stages of morning peak hours, maximum of 15% and 2% of the total requests in collaborative method are rejected, respectively. The total number of accepted requests is improved from 87% in the independent method to 95% in the collaborative method. According to our analysis, the link capacity is the limiting factor in causing the number of requests rejected in the MEC hosts collaboration method, i.e., if we have unlimited link capacity, all the requests will be accepted in the MEC hosts

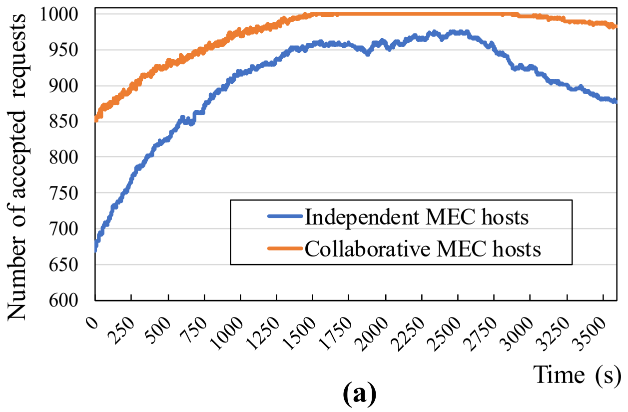
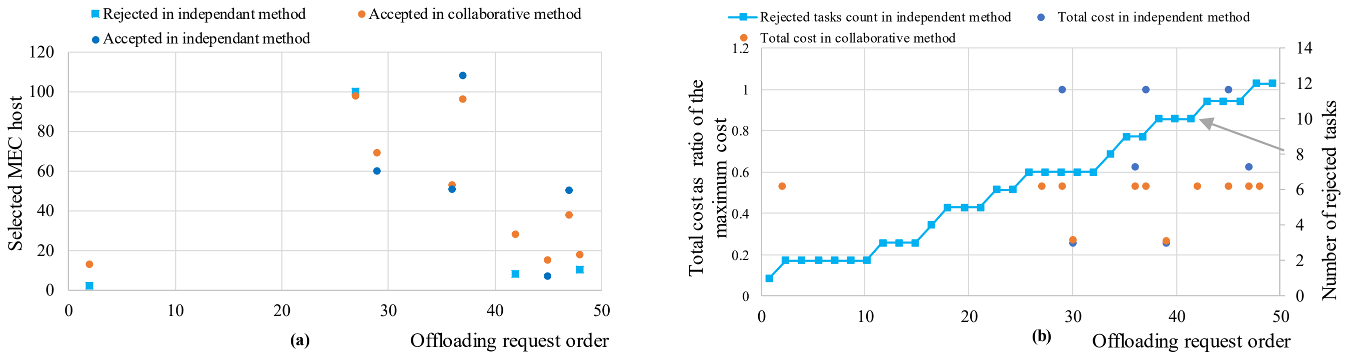


Figure 3 (a) Comparison of accepted requests in independent MEC hosts and collaborative MEC hosts methods. (b) Details of accepted requests in collaborative MEC hosts method.

collaborative method. Fig. 3(b) shows the details of accepted requests in the MEC host collaboration method. Self-accepted requests of the MEC host are the requests accepted by the MEC host that are requested by the devices in the serving coverage area of the MEC host. Delegated requests of the MEC host are the requests transferred to delegated MEC host by the serving MEC host. Delegation may occur due to the resource limitations in the serving MEC host or it is cheaper to hosting the user application in the delegated MEC host than the serving MEC host.

As we utilized first come first server scheduling at the MEC orchestrator, offloading requests are processed in the sequential order they are requested. Fig 4 (a) shows the deviations in MEC host selections of the first 50 offloading requests at time 100 s, i.e. only the variations in both methods in MEC host selections are displayed in the figure. Fig 4 (b) shows total cost of providing the offloading services for the requests. The cost is generally lower in collaboration method than the independent method.

MEC hosts selections for first fifty offloading requests at time 100s



MEC hosts selections for first seven hundred offloading requests at time 2400s

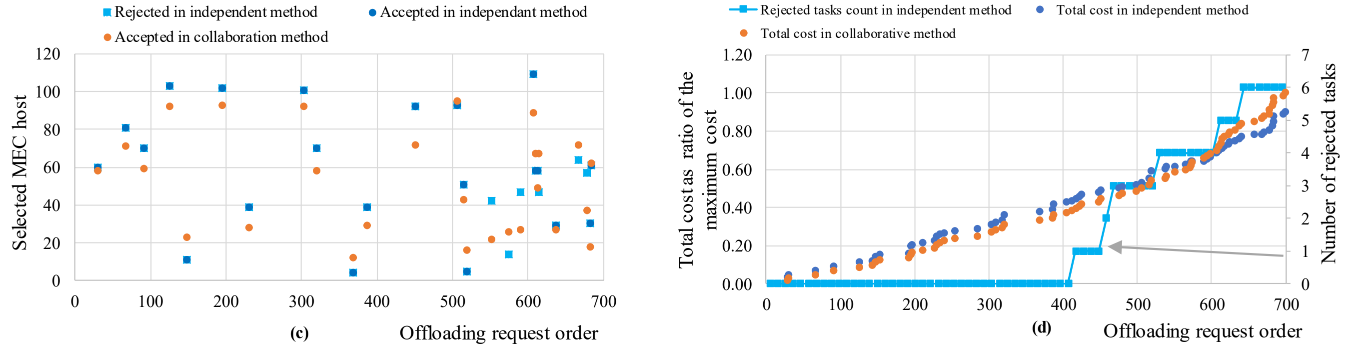


Figure 4(a) selected MEC hosts (b) total costs; for first fifty offloading requests at time 100 s. (c) selected MEC hosts (d) total costs; for first seven hundred offloading requests at time 2400 s.

Fig. 4 (c) shows the deviations in MEC host selections of the first 700 offloading requests at time 2,400 s. While all the first 700 tasks requests in collaborative method are accepted, however, 6 offloading requests are rejected in independent method as shown in blue line in the figure. Thus, the total number of offloading requests are higher in collaborative method. Further, number of MEC hosts collaborations increases with the number of offloading requests increases.

Fig. 4(d) shows the total cost of providing the offloading services, as the ratio of maximum cost, with the request arrival order in both methods. It also shows the number of rejected offloading requests in independent method. After the first three tasks are being rejected by the independent method, the cost of instantiating the application in collaborative method increases compared to independent method from the 593th offloading request onwards. The cost gap further increases as the number of tasks rejected at the independent method increases. This implies that when the number of requests accepted in the collaborative method increases beyond the margin, the cost of providing the offloading services also increases compared to independent method. This is the tradeoff between the cost of providing the offloading services to the quality of service in terms of the number of requests accepted. If the service providers want to increase the quality of service beyond the margin, then the cost increases.

V. SUMMARY

The MEC hosts selection problem in a collaborative MEC system is a challenging task for the MEC service providers in order to satisfy users' QoS requirements while minimizing service costs. In this work, we formulated the MEC hosts selection problem as a special case of binary programming problems, which minimizes the total cost incurred by the MEC service providers without compromising on QoS requirements. We modified Balas-Geoffrion algorithm to solve this problem as the additive algorithm minimizes the time complexity. Further, it provides near optimal solution, even if the calculations stop before all the possible solutions are enumerated. Our findings show that our extended algorithm outperforms the original Balas-Geoffrion algorithm in the number of iterations required to reach the optimal solution in the context of special case of binary programming similar to MEC host selection problem. Furthermore, we analyzed the tradeoff between the increased costs of the collaboration methods to the number of tasks rejected in the independent method.

REFERENCES

[1] ETSI, "Mobile edge computing (MEC): Technical requirements." ETSI Industry Specification Group (ISG), Mar-2016.

[2] T. X. Tran, A. Hajisami, P. Pandey, and D. Pompili, "Collaborative Mobile Edge Computing in 5G Networks: New Paradigms, Scenarios, and Challenges," *IEEE Communications Magazine*, vol. 55, no. 4, pp. 54–61, Apr. 2017.

[3] M. Rajib Arefin, T. Hossain, and M. Ainul Islam, "Additive Algorithm for Solving 0-1 Integer Linear Fractional Programming Problem," *Dhaka University Journal of Science*, vol. 61, Nov. 2013.

[4] M. Alicherry and T. V. Lakshman, "Network aware resource allocation in distributed clouds," in *2012 Proceedings IEEE INFOCOM*, 2012, pp. 963–971.

[5] J. Kangasharju, J. Roberts, and K. W. Ross, "Object replication strategies in content distribution networks," *Computer Communications*, vol. 25, no. 4, pp. 376–383, Mar. 2002.

[6] L. Tong, Y. Li, and W. Gao, "A hierarchical edge cloud architecture for mobile computing," in *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*, 2016, pp. 1–9.

[7] T. Taleb and A. Ksentini, "An analytical model for Follow Me Cloud," in *2013 IEEE Global Communications Conference (GLOBECOM)*, 2013, pp. 1291–1296.

[8] R. Urgaonkar, S. Wang, T. He, M. Zafer, K. Chan, and K. K. Leung, "Dynamic service migration and workload scheduling in edge-clouds," *Performance Evaluation*, vol. 91, pp. 205–228, Sep. 2015.

[9] S. E. Mahmoodi, R. N. Uma, and K. P. Subbalakshmi, "Optimal Joint Scheduling and Cloud Offloading for Mobile Applications," *IEEE Transactions on Cloud Computing*, pp. 1–1, 2018.

[10] J. Liu, Y. Mao, J. Zhang, and K. B. Letaief, "Delay-optimal computation task scheduling for mobile-edge computing systems," in *2016 IEEE International Symposium on Information Theory (ISIT)*, 2016, pp. 1451–1455.

[11] S. Wang, M. Zafer, and K. K. Leung, "Online Placement of Multi-Component Applications in Edge Computing Environments," *IEEE Access*, vol. 5, pp. 2514–2533, 2017.

[12] "Mobile-edge Computing - Introductory Technical White Paper V1 18-09-14 | Cloud Computing | Computer Network," *Scribd*. [Online]. Available: https://www.scribd.com/document/281496744/Mobile-edge-Computing-Introductory-Technical-White-Paper-V1-18-09-14. [Accessed: 05-Oct-2017].

[13] V. Karagiannis and A. Papageorgiou, "Network-integrated edge computing orchestrator for application placement," in *2017 13th International Conference on Network and Service Management (CNSM)*, 2017, pp. 1–5.

[14] N. Yu, Q. Xie, Q. Wang, H. Du, H. Huang, and X. Jia, "Collaborative Service Placement for Mobile Edge Computing Applications," in *2018 IEEE Global Communications Conference (GLOBECOM)*, 2018, pp. 1–6.

[15] L. Zhao and J. Liu, "Optimal Placement of Virtual Machines for Supporting Multiple Applications in Mobile Edge Networks," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 7, pp. 6533–6545, Jul. 2018.

[16] "OpenStack Docs: Filter Scheduler." [Online]. Available: https://docs.openstack.org/nova/latest/user/filter-scheduler.html. [Accessed: 15-Jan-2019].

[17] R. M. Karp, "Reducibility among Combinatorial Problems," MA: Springer US, 1972, pp. 85–103.

[18] E. Balas, F. Glover, and S. Zionts, "An Additive Algorithm for Solving Linear Programs with Zero-One Variables," *Operations Research*, vol. 13, no. 4, pp. 517–549, 1965.

[19] A. Geoffrion, "Integer Programming by Implicit Enumeration and Balas' Method," *SIAM Rev.*, vol. 9, no. 2, pp. 178–190, Apr. 1967.

[20] J. ter Wengel, "Allocations of Industries Given Generalized Distributional Constraints," in *Allocation of Industry in the Andean Common Market*, J. ter Wengel, Ed. Dordrecht: Springer Netherlands, 1980, pp. 109–131.

[21] S. Thananjeyan, C. A. Chan, E. Wong, and A. Nirmalathas, "Deployment and Resource Distribution of Mobile Edge Hosts Based on Correlated User Mobility," *IEEE Access*, vol. 7, pp. 148–159, 2019.

[22] Thananjeyan, S., Chan, C.A., Wong, E., and Nirmalathas, A., "Energy-Efficient Mobile Edge Hosts for Mobile Edge Computing System," presented at the 2018 International Conference on Information and Automation for Sustainability (ICIAfS), Colombo, 2018.

[23] L. Jiao, A. M. Tulino, J. Llorca, Y. Jin, and A. Sala, "Smoothed Online Resource Allocation in Multi-Tier Distributed Cloud Networks," *IEEE/ACM Transactions on Networking*, vol. 25, no. 4, pp. 2556–2570, Aug. 2017.