Edge Node Placement with Minimum Costs: When User Tolerance on Service Delay Matters

Xiaoyu Zhang, Shixun Huang, Hai Dong, and Zhifeng Bao

School of Computing Technologies, RMIT University
xiaoyu.zhang5@student.rmit.edu.au, shixun.huang@rmit.edu.au,
hai.dong@rmit.edu.au, zhifeng.bao@rmit.edu.au

Abstract. Edge node placement optimization has been an emerging research area that has drawn extraordinary attention from the disciplines of distributed and services computing. Existing studies, nevertheless, barely focus on overall deployment cost minimization with edge node site selection and server amount optimization, while bearing users' delay tolerance. In this paper, we focus on investigating feasible user delay tolerance-aware edge node site selection and server placement optimization strategies adaptive for real-world large-scale use cases, with the objective of deployment cost minimization. A *Coverage First Search* method is proposed to address this problem in polynomial time. The experiments conducted on a real-world dataset demonstrate the effectiveness of our method.

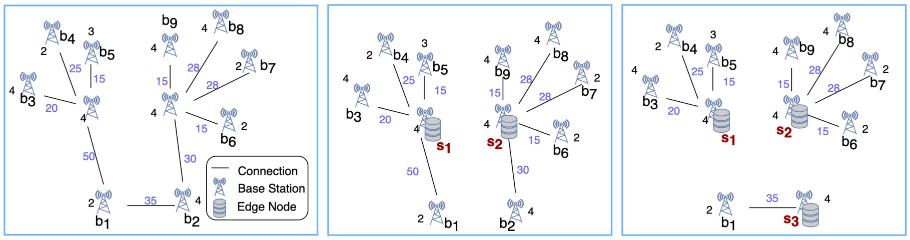
1 Introduction

Mobile Edge Computing (MEC) is a network architecture accompanying 5G. MEC deploys plenty of small-scale servers (known as edge servers or edge nodes) to network edges in a distributed manner. Users stay closer to those edge nodes in a MEC network, which not only can significantly reduce network latency but also can provide substantial computing resources to mobile users [5].

Problem. In this paper, we study the problem of optimal edge node deployment, aiming to provide qualified and low-latency services to massive mobile users city-wide with minimum cost. There are many factors that should be considered during the edge node deployment. First, the network QoS (Quality of Service) guarantee is the baseline of the deployment. For example, delay, as one of the most important QoS factors, should not exceed users' tolerance [4]. Second, minimizing the deployment cost is always welcome and should never be neglected [1, 6]. Third, the resource is finite, but the design of MEC is expected to provide users ample resources, with which goal the MEC should be optimized for higher productivity. [3, 7, 8]. Thus, the selected edge nodes with "just enough" computation resources allocated is always the ideal case.

Motivation. There is always a trade-off between the edge node deployment cost and the delay experienced by mobile users [5]. That trade-off is highly related to edge node site selection and the corresponding resource allocation. Deploying

Zhang, X., Huang, S., Dong, H., Bao, Z. (2021). Edge Node Placement with Minimum Costs: When User Tolerance on Service Delay Matters. In: Hacid, H., Kao, O., Mecella, M., Moha, N., Paik, Hy. (eds) Service-Oriented Computing. ICSOC 2021. Lecture Notes in Computer Science(), vol 13121. Springer, Cham. https://doi.org/10.1007/978-3-030-91431-8_53



(a) Initial network (b) OPT with 3.0s delay (c) OPT with 1.5s delay

Fig. 1: Example of optimal EN deployment under different delay tolerance

more edge nodes can potentially reduce the transmission delay by decreasing the average distance between base stations and edge nodes. Also, adding more servers (i.e. computing resources) into edge nodes can cut down the computation delay as edge nodes would have higher computation capacity. However, both cases will inflate the overall deployment cost. Therefore, with users' delay tolerance, it is necessary to find the most cost-efficient edge node deployment strategy such that the overall deployment cost is minimized, as shown in the following example.

Example 1. Fig. 1 demonstrates how users' delay tolerance affects the optimal edge node deployment when considering cost-efficiency. Fig. 1a shows the initial connections between base stations. Edge nodes will be deployed that co-locate with base stations. Developing an edge node within a base station will introduce a setup cost, while adding servers to an edge node to increase its computing capacity will generate server purchase costs. Given users' delay tolerance threshold and the goal of cost minimization, placing just the right amount of edge nodes accompanying workload-matched server numbers is the ideal case.

With the objective of minimizing the total cost, the optimal edge node deployment strategy will vary under different users' delay tolerance. Fig. 1b illustrates the optimal edge node placement in case the users' delay tolerance threshold is 3.0s, where the most cost-efficient deployment is to develop two edge nodes S_1 and S_2. Adding one more edge node is more expensive than adding more servers to existing nodes. However, when we decrease the delay tolerance threshold to 1.5s, the optimal placement becomes what is shown in Fig. 1c. To satisfy this more rigorous delay tolerance requirement, there are two intuitive options: continuously adding more servers to existing edge nodes to further decline the computation delay, or developing a new edge node to decrease the transmission delay. Fig. 1c shows that the optimal solution is to develop a new edge node instead of adding more servers.

To the best of our knowledge, few researchers attempted to address the trade-off between cost and delay while considering the computation resource allocation [2, 6]. Existing studies are subject to the following major limitations. First, the *scalability* and *practicability* of existing solutions have not been fully explored for large-scale datasets. in reality, the number of deployed base stations is significant and keeps increasing (e.g., Shanghai in China is projected to have 50 5G base stations per square km[1]). Designing a highly scalable and efficient solution is therefore essential. Second, the issue of delay has not been well addressed. The existing studies ignore the fact that the computation delay is supposed to decrease with more servers placed in edge nodes.

Main Contributions. In this paper, we aim to address the trade-off between the cost and delay by formulating our edge node deployment problem with the objective of minimizing the deployment cost while considering users' delay tolerance. Our deployment plan will not only explore optimal edge node sites but also provide the optimal resource allocation according to the real workload in edge nodes. Our major contributions include:

- We formulate a problem to address the trade-off between the deployment cost, and the transmission and computation delay. We propose a peak-based workload measurement for the robustness of our deployment. Moreover, we define a delay measurement to make it fit in real-world cases. (Section 2)
- We propose a Coverage First Search (CFS) algorithm to solve the defined problem in polynomial time. (Section 3)
- We conduct extensive experiments to demonstrate the effectiveness of our method. (Section 4)

2 Problem Formulation

In this section, we firstly define the MEC network and its components. Then, we define the workload and delay measurement. Finally, we formulate our problem with the goal of minimizing the deployment cost with delay tolerance satisfied.

Preliminaries. Here, we introduce some key concepts to facilitate our illustration across the paper.

MEC network. The MEC network consists of a set B of base stations (BSs) and a set S of edge nodes (ENs). Elements in both B and S are denoted by a tuple (id, lat, lng, n) where lat, lng and n represent latitude, longitude and number of servers added respectively. Following a widely adopted setting [3, 7, 8]: ENs are co-located with BSs, we upgrade a BS to an EN by adding servers to it. Multiple servers are allowed to an EN to provide enough computation capacity. Then, we have: (1) $\forall\ b \in B,\ b.n = 0$; (2) $\forall\ s \in S,\ s.n \geq 1$;

EN setup cost and server cost. We define two kinds of costs: *EN setup cost* and *server cost.* Let p_r denote the setup cost, which is the cost of upgrading a BS to an EN, such as infrastructure renting fee and construction fee. Let p_s denote the server cost, which is the cost for purchasing new servers to ENs. To be more specific, installing a server to a base station will cost $p_r + p_s$, while adding a server to edge node will simply cost p_s.

[1] https://techblog.comsoc.org/2020/08/07/5g-base-station-deployments-open-ran-competition-huge-5g-bs-power-problem/

Connectivity and EN service range. We define two BSs b_1 and b_2 are connected if they meet a certain delay threshold which is constrained by transmission delay and computation delay together. We will elaborate these two delays later in this section. Then, the service range of an EN $s \in S$ denoted as $R(s)$ is represented by a set of BSs, that are directly connected with s.

BS assignment. Give that ENs may have their service range overlapped, we assign base stations to edge nodes based on the following criteria: (1) a BS can be assigned to one EN only; (2) the selected ENs cover all BSs in the network. We assign EN with enough computation capacity to process all incoming tasks from the assigned BSs and will not further offload the task to other ENs. We represent the assignment with a set of key-value pairs $\mathcal{A}$, where the key is the EN, followed by a set of assigned BS as value, e.g. $\mathcal{A}[s_1] = \{b_1, b_2, b_{19}, ...\}$.

Workload Measurement. Most of existing studies measures the workload of a BS or an EN by task's average requesting [7]. However, in real cases, the workload usually fluctuates dramatically during a day [5], so the peak workload is non-negligible in some cases considering the robustness of the network, especially during the rush hour.

We propose a peak metric to measure the workload. We assume that the tasks transmitted in the network are all data-intensive computing tasks, e.g. HD videos, to guarantee that the MEC network is capable of dealing with overwhelming workload. We define the task size of a single task as ξ in bits. Then the peak workload will appear at the time period that has the largest number of coming tasks. We assume the task can be processed as soon as it arrives. We define tasks that have their processing time overlap as concurrent tasks. Let CT denote the number of concurrent tasks and CT_{max} denote the largest number of concurrent tasks that have occurred.

Thus, with the peak metric, we define the workload of a BS b as:

$$W(b) = \xi \cdot CT_{max} \tag{1}$$

Delay Measurement. Since we assume task offloading between ENs is not allowed, there are two major delays: transmission delay for a task to transmit between a BS and an EN and the computation delay for a task to be computed in an EN [5], which are related to the channel's transmission capacity and EN's computation capacity respectively.

Transmission capacity. We adopt Shannon's channel capacity formula[2] to compute a channel's transmission capacity (denoted as C_{trams}):

$$C_{trans} = \mathcal{B} \log_2 (1 + \frac{SP}{N}) \tag{2}$$

In this equation, $\mathcal{B}$ represents the channel's bandwidth, SP represents the average received signal power over the channel and N represents the average noise power over the channel. We assume that the signal power is identical to all channels. Considering channel noise can be affected by many factors, such as

[2] Shannon theorem: http://www.inf.fu-berlin.de/lehre/WS01/19548-U/shannon.html

distance, environments and quality of cable[3], we use a very common way in the literature by assuming that the channel noise is only affected by the distance [7, 3]. We define the noise as $\mathrm{N} = \alpha \cdot d(s, b)$, where $d(s, b)$ denotes the distance between s and b, and α is a coefficient between N and the distance.

Computation capacity. Adding servers to EN gives it computation capacity. We assume that servers placed to EN have the same computation capacity μ bit/s. Then, for an EN s with $s.n$ servers placed, the computation capacity is:

$$C_{comp} = s.n \cdot \mu \tag{3}$$

Delay. The delay calculation depends on the data size and processing capacity[4]. Since the delay incurred between a BS b and an EN s includes transmission delay and computation delay, we define our delay model as

$$D(b, s) = \frac{W(b)}{C_{trans}} + \frac{W(s)}{C_{comp}} \tag{4}$$

Definition 1. ***Qualified EN Placement Plan.*** *Given a set of BSs B and a delay threshold θ, select a subset $S \subseteq B$ as ENs such that the following constraints hold: (1) $\forall s \in S\ b \in \mathcal{A}[s]$, $D(s, b) \leq \theta$; (2) $\bigcup_{s \in S} \mathcal{A}[s] = B \backslash S$; (3) $\forall s_i, s_j \in S, s_i \neq s_j, \mathcal{A}[s_i] \cap \mathcal{A}[s_j] = \emptyset$.*

Intuitively, these constraints indicate that the total delay experienced by the user does not exceed θ, S should serve all $b \in B \backslash S$, and each BS will be assigned to one and only one EN for task offloading, respectively.

Definition 2. ***Cost Minimization in MEC Edge Node Placement (CMMENP).*** *The CMMENP problem is to find a solution S^* which can minimise the total cost*

$$F(S^*) = \underset{S \subseteq B}{\arg\min} \sum_{s \in S} (p_r + s.n \cdot p_s) \tag{5}$$

where $F(S^*)$ denotes the total cost incurred by selecting S^* as ENs, S is a qualified EN placement plan, p_r is the setup cost, and p_s is the server cost.

3 Methodologies

In this section, we will introduce a greedy-based solution: Coverage First Search (CFS), which is an efficient algorithm that aims to provide a solution in polynomial time.

---

[3] Noise: https://documentation.meraki.com/MR/WiFi_Basics_and_Best_Practices

[4] https://manuals.gfi.com/en/exinda/help/content/exos/how-stuff-works/network-performance-metrics.htm

Algorithm 1: CFS Algorithm

Input : Base Station set B, Delay threshold θ

Output: Edge Node set S

1 $S = \emptyset$, $\mathcal{A} \leftarrow \emptyset$; `//` $\mathcal{A}$`: a set of` $\langle s : \{b_1, b_2, ...\}\rangle$ `for BS assignment`
2 **while** $B \neq \emptyset$ **do**
3     $B \leftarrow$ getConnection(B, θ)
4     $b_s \leftarrow \arg\max\{|R(b)| \mid b \in B\}$; $\mathcal{A}[b_s] \leftarrow R(b_s)$
5     $S \leftarrow S \cap b_s$;
6     $B \leftarrow B \setminus b_s$
7   **foreach** $b \in \mathcal{A}[b_s]$ **do**
8         $B \leftarrow B \setminus b$
9 **return** S

In order to improve the computation efficiency, considering the objective of cost minimization, we devise an approximate algorithm, Coverage First Search (CFS). The core idea of CFS is to minimize the number of ENs being deployed, as the construction cost of edge nodes (e.g., EN setup cost) is usually much greater than the cost of a standard server (e.g., server cost) [6]. As shown in Algorithm 1, we will first model the connections between BSs according to the delay threshold θ (line 3). Then, we iteratively pick the BS which has the highest number of connections as the site to construct an EN, and assign it with all its connected BSs in its service range R (line 4). Finally, we remove the EN and its assigned BSs from the input BS set (lines 6-8). We repeat this process until all the BSs in the input set being assigned.

4 Evaluation

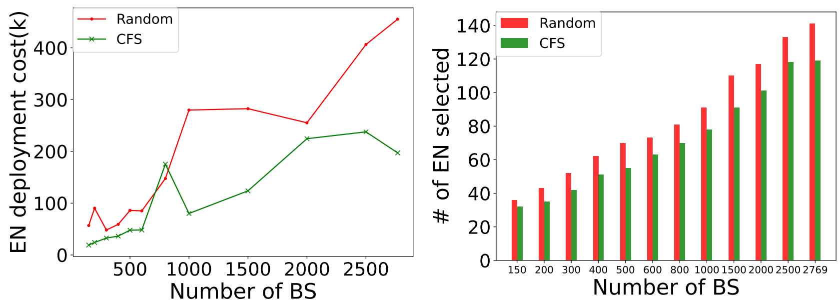
We conduct extensive experiments on CFS and random method to evaluate their effectiveness with a real-world large-scale dataset.

4.1 Experiment Settings

Dataset. Our experiments are conducted on the Shanghai Telecom Dataset [5].
Experiment Environment. All experiments are conducted on MacOS (2.5 GHz Daul-Core Intel i7 processor and 16GB memory). Our methods are implemented in Java.

Parameter Settings. Following [6], we also set the ratio of edge node construction cost and a standard server cost as 4:1 and we set the computing capacity of a standard server as $\mu = 100$ bps. The bandwidth $\mathcal{B}$ is set to 200 Mbps[6]. The Channel signal power SP is set to -35 dBm[7]. The single task size ξ is configured

[5] Shanghai Telecom Dataset: http://sguangwang.com/TelecomDataset.html
[6] https://go.frontier.com/business/internet/200-mbps
[7] https://www.metageek.com/training/resources/wifi-signal-strength-basics.html



(a) Deployment cost

(b) # of EN selected

Fig. 2: Effectiveness with different BS input scale

to 15 bits[8]. Furthermore, all our experiments are conducted with a default delay threshold $\theta = 14$s based on the empirical studies depicted in Section 4.2.

Evaluation Metrics. The effectiveness metrics include the deployment cost and the number of selected ENs. We measure the effectiveness of CFS and Ramdom on different numbers of BSs.

Methods for comparison. We compare the performance of the following two methods: a Random method that randomly picks BSs and our proposed CFS method.

4.2 Experimental Results

The deployment cost and the number of selected ENs of the aforementioned candidate solutions on different BS input scales are shown in Fig. 2a and Fig. 2b.

As shown in Fig. 2a, compared with Random, CFS shows outstanding cost-saving performance especially when the number of participated BSs is high. The cost growth of CFS is relatively smoother than Random method with the increasing number of BSs, which indicates its higher reliability.

We can observe from Fig. 2b that the numbers of ENs selected by CFS is clearly smaller than that is selected by Random. It shows a steady increasing trend for both random and CFS in terms of the number of EN selected, while we can see obvious fluctuations in terms of the deployment cost in Fig. 2a. Such phenomenon reflects the major limitation of CFS that it is incapable of finding all potentially suitable EN locations and optimizing the assignment between BSs and ENs, which causes the following problem: (1) its selected ENs may not be in the optimal locations. (2) the ENs would require high computation capacity to serve distant BSs. It explains the abnormal cost fluctuations experienced by CFS (e.g. when the number of BSs is 800 in Fig. 2a). Similarly, the Random selection also experiences such issue, as we can see obvious fluctuations for random either.

[8] https://www.amaysim.com.au/blog/stuff-made-simple/internet-data-usage-guide

5 Conclusion

In this paper, we defined an MEC Edge Node Placement Problem to address the trade-off between deployment cost and users' delay tolerance. Within this problem, we defined a practical and delicate delay measurement and propose a peak workload metric. We proposed an approximate solution CFS whose effectiveness is demonstrated via our extensive experiments on a real-world dataset. For future works, we will focus on optimizing the proposed solutions to further improve their effectiveness and exploring their performance with respect to the average workload metric.

Acknowledgments. This research is supported in part by ARC DP200102611.

References

1. Chen, L., Wu, J., Zhou, G., Ma, L.: Quick: Qos-guaranteed efficient cloudlet placement in wireless metropolitan area networks. The Journal of Supercomputing **74**(8), 4037–4059 (2018)
2. Fan, Q., Ansari, N.: Cost aware cloudlet placement for big data processing at the edge. In: ICC. pp. 1–6. IEEE (2017)
3. Guo, Y., Wang, S., Zhou, A., Xu, J., Yuan, J., Hsu, C.: User allocation-aware edge cloud placement in mobile edge computing. Softw. Pract. Exp. **50**(5), 489–502 (2020)
4. Ma, L., Wu, J., Chen, L., Liu, Z.: Fast algorithms for capacitated cloudlet placements. In: CSCWD. pp. 439–444. IEEE (2017)
5. Mao, Y., You, C., Zhang, J., Huang, K., Letaief, K.B.: A survey on mobile edge computing: The communication perspective. IEEE Commun. Surv. Tutorials **19**(4), 2322–2358 (2017)
6. Mondal, S., Das, G., Wong, E.: Ccompassion: A hybrid cloudlet placement framework over passive optical access networks. In: INFOCOM. pp. 216–224. IEEE (2018)
7. Wang, S., Zhao, Y., Xu, J., Yuan, J., Hsu, C.: Edge server placement in mobile edge computing. J Parallel Distrib Comput **127**, 160–168 (2019)
8. Xu, X., Xue, Y., Qi, L., Zhang, X., Wan, S., Dou, W., Chang, V.: Load-aware edge server placement for mobile edge computing in 5g networks. In: International Conference on Service-Oriented Computing. pp. 494–507. Springer (2019)