Elastic autoscaling for distributed workflows in MEC networks

Benedetta Picano, Riccardo Reali, Leonardo Scommegna and Enrico Vicario

Abstract With the recent advancements in computing technologies, new paradigms have emerged enabling users to access a large variety of distributed resources, overcoming several limitations of localized applications and information storage. Among these paradigms, Mobile Edge Computing (MEC) places storage and computing capabilities at the edge of the network, significantly decreasing congestion and service response times, at the cost of limited capacities. Within this context, the emergence of novel computationally intensive services has triggered the necessity to design algorithms that adaptively scale resources, achieving solutions tailored to traffic demand. In this paper, we present a preliminary scaling method to determine the resource provisioning of complex workflows of web services that are distributed on a MEC infrastructure, with the intent of improving the distribution of the end-to-end response time of the workflow. The method is designed to run compositionally, exploiting a structured hierarchical workflow representation, enabling efficient top-down determination of the resource provisioning. The method is also formalized to act considering the inherent limitations and complexities of an MEC network landscape. In so doing, we demonstrate the applicability of the approach on two synthetic application scenarios, confirming the validity of the proposed elastic scheme in optimizing resource management within a resource-constrained MEC network.

1 Introduction

Recent advancements in networking and computing technologies have sparked a growing interest in considering computation and communication in a collaborative and distributed manner, aligning with the network-computing paradigm [4]. Network computing introduces a novel computing paradigm wherein all informa-

Benedetta Picano, Riccardo Reali, Leonardo Scommegna, Enrico Vicario,
Department of Information Engineering, University of Florence, e-mail: {benedetta.picano, riccardo.reali, leonardo.scommegna, enrico.vicario}@unifi.it

tion, data, and software applications exist on a network accessed by users on demand, whose infrastructure is typically deployed within the edge-to-cloud continuum. This computing approach promises to enable users to access a comprehensive range of resources from any location, eliminating the limitations associated with localized storage of information and applications [5]. This is due to the presence of Mobile Edge Computing (MEC) nodes which move storage and computing facilities at the edge of the network, close to end-users, resulting in significant performance enhancements, especially in terms of latency and response times. At the same time, the upcoming next-generation networks will enable the new era of computational intensive service classes, characterized by heterogeneous quality-of-service (QoS) and quality-of-experience requirements. To ensure effective service provisioning, the network resources need to be optimized and handled properly. Due to the computational-hungry nature of next-generation applications, and because of the high-velocity links expected to serve the new-generation networks, the usage of computational resources, if not optimized, risks becoming the bottleneck of novel applications. When resource exploitation is optimized, service providers only incur costs for the resources they utilize at a given moment. When managed effectively, this approach can lead to lower costs and a higher quality of service compared to hosting on traditional hardware [3]. Due to the dynamic and unpredictable nature of shared resources, autoscaling mechanisms must be designed to handle the complexity and time-varying nature of resource demand. The goal is to achieve a runtime scaling system that is self-aware, self-adaptive, and reliable in the face of changing demands [3].

Recently, autoscaling, originally designed and confined to cloud-based solutions, is gaining attention for its extension into distributed MEC contexts. For example, the paper [11] proposes a latency-optimal scheme to solve the monitoring function placement problem, considering an edge-to-cloud landscape. In order to favor the system scalability, authors design a hierarchical monitoring system topology, where a metaheuristic algorithm is exploited to adaptively scale of resource pool of the MEC infrastructure. Then, an online scaling scheme was developed to perform real-time and on-demand resource allocation in such a monitoring system. In [9], the authors introduced deep learning models that encompass both centralized and federated strategies. These models are designed to execute both horizontal and vertical autoscaling across multi-domain networks. Authors in [6], propose an innovative auto-scaling method using a deep reinforcement learning-based algorithm. This method aims to dynamically adjust the number of instances assigned to an atomic microservice that composes a service, thereby optimizing resource allocation and potentially enhancing service performance.

An innovative approach to meet the dynamically changing network service demands in 5G networks was developed in [8]. The authors applied machine learning models for auto-scaling and predicting the required number of virtual network function instances based on traffic demand. Furthermore, Integer Linear Programming techniques were exploited to solve a joint user association and Service Function Chain placement problem. To address the scalability concern of the ILP model, they proposed a heuristic algorithm. However, existing solutions do not consider the ne-

cessity to realize coordinated scaling among distributed MEC resources for service workflows, i.e., services composed of atomic elementary tasks structured as directed acyclic graphs. In [1], a coordinated compositional approach is proposed to scale the resource provisioning of a workflow of services. The approach minimizes the workflow e2e response time, by considering topological information and exploiting a stochastic characterization of service durations, for an environment where any transmission costs occur. In this reference, the main contributions of this paper can be summarized as follows:

- The design and development of an effective coordinated vertical scaling scheme of MEC node resources to execute stochastic workflows within the distributed edge network. The objective is to meet QoS constraints by improving the end-to-end (e2e) response time distribution of the workflow.
- Definition of a heuristic that, through performing an efficient compositional analysis, deduces the resource provisioning of each sub-workflow. The proposed algorithm employs a structured workflow model to scale resources in a top-down fashion, including inter-MEC node communication costs.
- Application scenarios aimed at demonstrating the applicability of the method to workflow of services that are deployed on a MEC network, at illustrating a methodology to explore the solution space of the problem, and at highlighting the complexity of the problem which is opened to many relevant challenges.

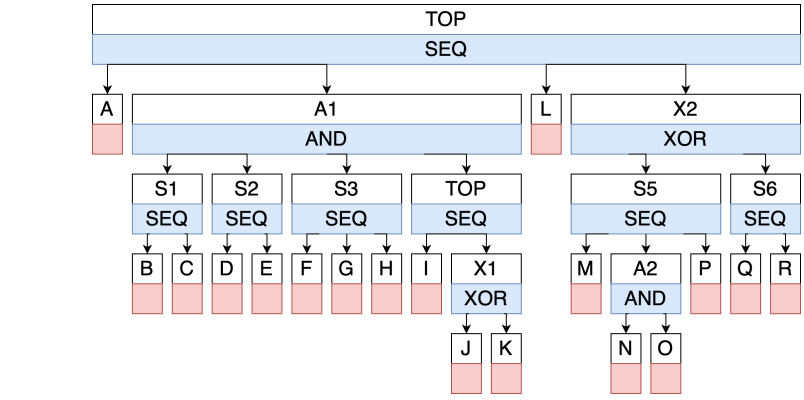
2 Workflow Modelling

We model workflows by combining Stochastic Time Petri Net (STPN) [10] blocks. An STPN block is a single-entry/single-exit model, which receives a token when workflow execution starts, and eventually ends with probability 1 (w.p.1). Blocks are recursively combined with sequential, split/join, and choice/merge workflow patterns [7]. We reference workflows according to the following EBNF syntax:

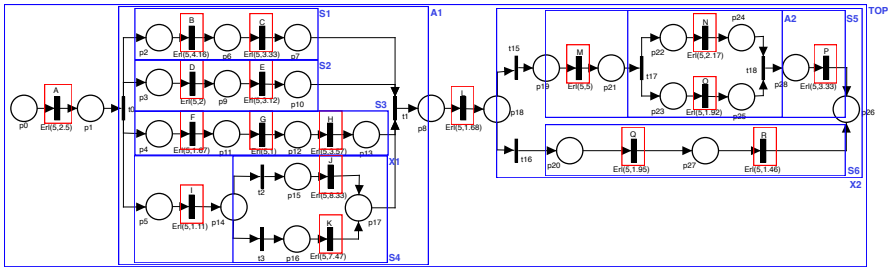
$$
\begin{aligned}
&< block > ::= \\
&\qquad \mathrm{SEQ}\,(< block > \{, < block >\}) \mid \\
&\qquad \mathrm{AND}\,(< block > \{, < block >\}) \mid \\
&\qquad \mathrm{XOR}\,(\langle < block >, prob\rangle \{, \langle < block >, prob\rangle\}) \mid \\
&\qquad \mathrm{ACT}
\end{aligned}
$$

where ACT is an elementary activity (e.g., activity B in Fig. 1b), SEQ models sequential behaviors (e.g.,activity S1 in Fig. 1), AND models concurrent behaviors (e.g., activity A1 in Fig. 1), and XOR models alternative behaviors that occur with different probabilities (e.g., activity X1 in Fig. 1). A workflow is thus modeled as a *structure tree* [2] $S = \langle N, n_0 \rangle$, where N is the set of nodes (i.e., blocks) and $n_0 \in N$ is the root node (i.e., the entire workflow). Each structure tree node $n_i \in N$ is characterized by the tuple $\langle n_i := R_i, Z_i, T_i, X_i \rangle$, where R_i is the amount of provisioned

resources, Z_i is the generally distributed random variable characterizing the node e2e response time that arises from the given provisioning, $T_i = E[Z_i]$ is the expected value of the response time, and $X_i = R_i T_i$ is the *job size*, representing the amount of work required to complete the node with the assigned resources. As workflows are recursive compositions of STPN blocks, topological complexity can notably increases. Complexity is further exacerbated by the general characterization of the activity durations, which lead to the unfeasibility of many effective analysis methods. In such cases, to evaluate the e2e response time distribution of a node, we leverage on a compositional technique [2]. In particular, workflows are evaluated by performing a top-down visit of the structure tree to estimate the analysis complexity of blocks, evaluating the response time distribution of the identified sub-workflows in isolation, and finally performing a bottom-up recomposition of the obtained results. In particular, when workflows are defined by well-nested composite blocks (i.e., composition of AND, SEQ, and XOR blocks), the exact e2e response time distribution can be evaluated by recursive numerical analysis.



(a)



(b)

Fig. 1: (a) Structure tree and (b) STPN modeling a synthetic workflow.

3 A communication costs aware scaling approach

3.1 A Performance Model

We propose a coordinated approach to determine the optimal resource provisioning of a workflow of services, that can be subject to communication costs incoming from distinct placements of the services on an edge network. The approach determines how to distribute an arbitrary amount of resources R^{in} to the elementary activities of a workflow, i.e., the leaves of the workflow structure tree, with the intent of minimizing the e2e response time distribution of the workflow. Resource allocations are determined by performing a top-down approach that splits the input resources to the child nodes, exploiting both topological information and the response time characterization of each node. In particular, the approach leverages on the assumption that the job size of an activity is invariant with respect to the provisioned resources, i.e. given two different resource allocations R_i, R_i' resulting in the expected response times T_i and T_i', then $T_i R_i = T_i' R_i' = X_i$, for each node n_i of the workflow. This enables to express the expected response time of a node as a function of the child node resources, which is employed as the objective function of an optimization problem. To deal with communication costs, we include a probability distribution representing the time spent to reach services that are hosted on different MEC nodes. In particular, D_i is introduced to represent the expected value of such communication time, for each node n_i. Consequently, the performance model of a node n_i is defined as: $T_i = X_i/R_i + D_i$, where X_i/R_i is the computation time of the activity. Communication times can be specified both on simple and composite activities, and are propagated bottom-up on the workflow nodes. In particular:

- For a sequence of k activities, $D_{\text{SEQ}(n_0,n_1,\ldots,n_k)} = \sum_{i=0}^{k} D_i$;
- For a fork/join of k, $D_{\text{AND}(n_0,n_1,\ldots,n_k)} = \max(D_0, D_1, \ldots, D_k)$;
- For a random choice between k activities, $D_{\text{XOR}(\langle n_0,p_0\rangle,\langle n_1,p_1\rangle,\ldots,\langle n_k,p_k\rangle)} = \sum_{i=0}^{k} p_i D_i$;

Note that, since it is not possible to determine the exact expected communication time of a fork/join activity, we select the maximum of its children. In so doing, we design the approach to deal with safety-critical contexts, where worst case scenario is typically considered.

To summarize how the approach proceeds as follows:

- Initially, an arbitrary amount of resources R_0^{in} is assigned to the root node n_0.
- For each non-leaf node n_k, the amount of input resources R_k^{in} is split by assigning an amount $R_j^{\star}$ to each child node n_j, i.e., $\sum_{n_j \in C_k} R_j^{\star} = R_k^{\text{in}}$; the assignment exploits a performance models where the job size of an activity is invariant with respect to any resource variation and communication costs changes the way resources are distributed.

By induction, the sum of the amounts of resources allocated to the leaf nodes is equal to the amount of resources of the root node, i.e., $\sum_{n_k | C_k = \emptyset} R_k^{\star} = R_0^{\text{in}}$.

3.2 Resource allocation decisions

We characterize different rules to determine the resource provisioning of a node, that are based on the workflow pattern represented by the node.

Elementary Activities. Let n_k be an elementary activity, when a new resource allocation $R^\star$ is evaluated by the approach, then its response time changes as

$$T_k^\star = \frac{R^\star}{R_k} T_i \tag{1}$$

which leads to a transformation of the node distribution parameters.

Sequential Activities. Let $n_k = SEQ(n_i, n_j)$ be the sequence of n_i and n_j. Then:

$$T_k = T_i + T_j = \frac{X_i}{R_i} + D_i + \frac{X_j}{R_k^{\text{in}} - R_i} + D_j \tag{2}$$

has a minimum with the following allocation

$$R_i^\star = \frac{\sqrt{X_i}}{\sqrt{X_i} + \sqrt{X_j}} R_k^{\text{in}} . \tag{3}$$

The result is obtained by imposing $\frac{dT_k}{dR_i} = 0$, and it can be extended by induction to the sequence of $K > 2$ activities, $SEQ(n_1, \ldots, n_K)$:

$$R_i^\star = \frac{\sqrt{X_i}}{\sum_{i=1}^{K} \sqrt{X_i}} R_k^{\text{in}} . \tag{4}$$

Concurrent Activities. Let $n_k = \text{AND}(n_i, n_j)$ be a fork/join between n_i and n_j. Then:

$$T_k = \max(T_i, T_j) = \max\left(\frac{X_i}{R_i} + D_i, \frac{X_j}{R_k^{\text{in}} - R_i} + D_j \right) \tag{5}$$

Since response time is not defined as an explicit function of R_i, the minimum can not be evaluated exploiting the Fermat theorem. Hence, we provide a heuristics evaluation of $R_i^\star$ which imposes equality between response times of n_i and n_j:

$$\frac{X_i}{R_i} + D_i = \frac{X_j}{(R_k^{\text{in}} - R_i)} + D_j . \tag{6}$$

This leads to the allocation:

$$R_i^\star = \begin{cases} \frac{X_i}{X_i + X_j} R_k^{\text{in}} & D_i = D_j \\ \frac{\Delta D R_k^{\text{in}} - X_i - X_j \pm \sqrt{4 \Delta D R_k^{\text{in}} X_i + (X_i + X_j - \Delta D R_k^{\text{in}})^2}}{2 \Delta D} & D_i \neq D_j \end{cases} \tag{7}$$

where $\Delta D = D_i - D_j$. In particular, when $D_i \neq D_j$, it is chosen the solution for which $R_i^\star > 0$ and $R_i^\star < R_k^{\text{in}}$. The solution is extended to m activities by re-

arranging the topology of the fork/join into a 2-children fork/join pattern, i.e., $\mathrm{AND}(n_1, n_2, ...n_N) = \mathrm{AND}(n_1, \mathrm{AND}(n_2, ..., n_N))$.

Alternative Activities. Let $n_k = \mathrm{XOR}(\langle n_i, p_i \rangle, \langle n_j, p_j \rangle)$ be a random alternative choice n_i and n_j, with probabilities p_i and $p_j = 1 - p_i$. Then:

$$T_k = p_i T_i + p_j T_j = p_i \left(\frac{X_i}{R_i} + D_i \right) + p_j \left(\frac{X_j}{R_k^{\mathrm{in}} - R_i} + D_i \right) \tag{8}$$

has the minimum:

$$T_k^\star = \frac{\left(\sqrt{p_i X_i} + \sqrt{p_j X_j} \right)^2}{R_k^{\mathrm{in}}} \tag{9}$$

for the following resource allocation:

$$R_i^\star = \frac{\sqrt{p_i X_i}}{\sqrt{p_i X_i} + \sqrt{p_j X_j}} R_k^{\mathrm{in}} \tag{10}$$

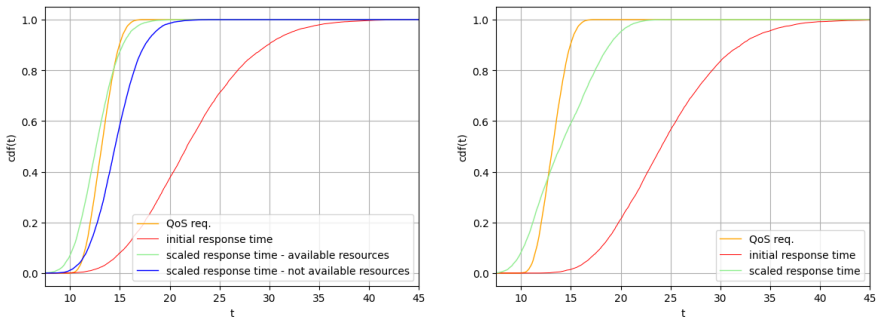
which is in turn obtained by exploiting the Fermat theorem. As the solution shows, the optimal allocation of a XOR node is a generalization of the optimal allocation for a SEQ node. Hence, resource allocation for n activities, can be derived as done for sequential nodes. Note that resources R_k^{in} are split among activities i and j, independently from how they occur or not. This typically occurs in service-oriented applications, where each service has a reserved amount of resources. In contrast, for Function as a Service (FaaS) solutions, resources are allocated on-demand only when a service is executed: in this case, resource costs are accrued only for the selected service, and the expected cost is $p_i R_i + p_j R_j$ instead of $R_i + R_j$, resulting in a different optimal allocation.

4 Application Scenarios

We illustrate a preliminary methodology to explore the space design of a MEC network hosting services of a workflow. We consider a synthetic and well-nested workflow (see Figs. 1a and 1b), combining 19 web services, each distributed as a 5-phases Erlang distribution with rate randomly selected in $[0, 10]$, for a maximum concurrency degree equals to 4. It is assumed that the considered response time arises by provisioning 1 resource to each service. We consider a QoS requirement obtained as 5-phase Erlang distribution whose rate is chosen to fit an expected value of 12.96ms, which is the half of the workflow expected response time. Then, we consider two scenarios. In the first, each service of the workflow is deployed on a single MEC node; in the second, services B, C, M, N, P, Q are placed in a different MEC node, thus introducing some communication costs in the QoS requirement fit. In particular, communication times are characterized as uniform distributions, where support bounds are randomly selected in $[0, 5]$. In both scenarios, we assume that each node has an residual availability of 20 resources. Fig. 2a illustrates the im-

pact of limited MEC node resource availability to the problem of resource scaling. By applying the proposed technique, the QoS requirement could be met without significant effort (green line), by provisioning a total amount of 22.91 resources. However, this quantity overcomes the considered availability, requiring to adjust the computed provisioning, but ending up worsening the e2e response time distribution (blue line), which results not to fulfill the QoS requirement. Figure 2b illustrates how communication costs enable to mitigate the impact of limited resource availability. The presence of transmission costs implies a higher resource demand for the entire workflow to fulfill the QoS requirement, which is met with a total amount of 27.01 resources (green line). However, transmission costs affect the ways resources are distributed among the services and allocated to the considered MEC nodes. In particular, the total amount of resources allocated on the node that hosts services that causes transmission costs is 8.46, while in the other node 18.64 resources are allocated. Despite the higher resource costs, transmission costs mitigate MEC node saturation, which may be a desirable implication in the prospect of solving other problems such as service offloading or dynamic service placement. Table 1 reports the resource allocations evaluated for the considered scenarios. Column 2 provides the provisioning of resources when no constraints are given on the resource availability of the nodes. Column 3 reports the resource provisioning when all services are deployed on the single MEC node *A*. Finally, column 5 reports the resource allocation in the case services B, C, M, N, P, Q are deployed on the MEC node *B*. The last row of the table reports the cumulative resources provisioning of each considered scenario.

The proposed scenarios allow us to highlight the inherent complexity of the problem of provisioning resources to services hosted on MEC nodes, by illustrating a methodology through which exploring the design space of a workflow of services that is deployed at the edge of the network.



(a) Single MEC node

(b) Multiple MEC nodes

Fig. 2: Application of the proposed approach to different MEC network scenarios.

SERVICE	OPTIMAL PROVISIONING	RESOURCES	NODE	RESOURCES	NODE
A	1.87	1.66		2.03	
D	0.80	0.69		1.00	
E	1.96	1.68		0.80	
F	0.56	0.48		1.97	
G	2.53	2.17		2.54	
H	1.34	1.15		1.34	A
I	1.52	1.31		1.53	
J	0.35	0.30		0.35	
K	0.45	0.39	A	0.46	
L	2.29	2.03		2.49	
Q	1.49	1.32		1.91	
R	1.88	1.52		2.21	
B	0.63	0.54		1.36	
C	1.00	0.85		1.52	
M	0.93	0.82		1.19	B
N	1.08	0.95		1.38	
O	1.22	1.08		1.56	
P	1.14	1.01		1.46	
TOTAL	22.91	20.00		27.01	

Table 1: Resources provisioning obtained by applying the proposed approach without constraint on the resource availability (column 2), when services are deployed on a single MEC node (column 3) and when services are deployed on different MEC nodes (column 5).

5 Conclusions

In this paper, we propose a coordinated approach to evaluate the resource provisioning of a workflow of web services that are deployed at the edge of a network. Tha approach is experimented on a workflow that exhibits a well-nested topology with a non-negligible degree of complexity, with a maximum concurrency degree equals to 4. Evaluation of resource provisioning is performed in a top-down fashion, exploiting a hierarchical formalism, termed structure tree. To identify resource allocations for elementary web services, the approach solves an optimization problem which aims at minimizing the response time of structure tree nodes. In so doing, we leverage on an assumption of invariance of the job size of each node. We also characterize transmission costs to deal with services deployed on different MEC nodes.

Finally, the experimented workflow allows us to illustrate a methodology to explore the design space of a workflow that is deployed on an MEC network, illustrating how the approach can be exploited to analyze different scenarios. The results prove the applicability of the method, and highlight the inherent complexity of evaluating the optimal resource provisioning for a workflow deployed on an edge network.

References

1. Carnevali, L., Paolieri, M., Picano, B., Reali, R., Scommegna, L., Vicario, E.: A quantitative approach to coordinated scaling of resources in complex cloud computing workflows. In: European Workshop on Performance Engineering. pp. 309–324. Springer (2023)
2. Carnevali, L., Paolieri, M., Reali, R., Vicario, E.: Compositional safe approximation of response time probability density function of complex workflows. ACM Transactions on Modeling and Computer Simulation (2023)
3. Espadas, J., Molina, A., Jiménez, G., Molina, M., Ramírez, R., Concha, D.: A tenant-based resource allocation model for scaling software-as-a-service applications over cloud computing infrastructures. Future Generation Computer Systems **29**(1), 273–286 (2013). https://doi.org/https://doi.org/10.1016/j.future.2011.10.013, https://www.sciencedirect.com/science/article/pii/S0167739X1100210X, including Special section: AIRCC-NetCoM 2009 and Special section: Clouds and Service-Oriented Architectures
4. Fantacci, R., Picano, B.: Edge-based virtual reality over 6g terahertz channels. IEEE Network **35**(5), 28–33 (2021). https://doi.org/10.1109/MNET.101.2100023
5. Jiang, J., Lu, J., Zhang, G., Long, G.: Optimal cloud resource auto-scaling for web applications. In: 2013 13th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing. pp. 58–65 (2013). https://doi.org/10.1109/CCGrid.2013.73
6. Lee, D.Y., Jeong, S.Y., Ko, K.C., Yoo, J.H., Hong, J.W.K.: Deep q-network-based auto scaling for service in a multi-access edge computing environment. International Journal of Network Management **31**(6), e2176 (2021)
7. Russell, N., Ter Hofstede, A.H., Van Der Aalst, W.M., Mulyar, N.: Workflow control-flow patterns: A revised view. BPM Center Report BPM-06-22, BPMcenter. org **2006** (2006)
8. Subramanya, T., Harutyunyan, D., Riggio, R.: Machine learning-driven service function chain placement and scaling in mec-enabled 5g networks. Computer Networks **166**, 106980 (2020)
9. Subramanya, T., Riggio, R.: Centralized and federated learning for predictive vnf autoscaling in multi-domain 5g networks and beyond. IEEE Transactions on Network and Service Management **18**(1), 63–78 (2021). https://doi.org/10.1109/TNSM.2021.3050955
10. Vicario, E., Sassoli, L., Carnevali, L.: Using stochastic state classes in quantitative evaluation of dense-time reactive systems. IEEE Transactions on Software Engineering **35**(5), 703–719 (2009)
11. Yuan, Q., Ji, X., Tang, H., You, W.: Toward latency-optimal placement and autoscaling of monitoring functions in mec. IEEE Access **8**, 41649–41658 (2020). https://doi.org/10.1109/ACCESS.2020.2976858