Performance Evaluation of Placement Policies for Cloud-Edge Applications

I. Mongiardo, L. Massari, M. Calzarossa, B. Bermejo, D. Tessera

Abstract The applications commonly deployed in cloud-edge environments consist of multiple inter-dependent modules organized according to the Distributed Data Flow model. Decisions about the mapping between these modules and the available resources are quite difficult because of the resource-constrained nature of devices at the edge of the network and the timing requirements of the applications. In this paper we investigate the problem of application placement by proposing a lightweight heuristic that takes into account the volume of data exchanged between modules. In detail, among all possible devices, the proposed policy allocates a given module to the "best" device, that is, the device that ensures the minimum network delay. The policy has been evaluated in a simulated cloud-edge environment based on the iFogSim toolkit. We consider workloads consisting of applications with different demands in terms of processing and data exchanged between modules. The simulation results are promising and indicate that our policy offers competitive advantages when compared to other heuristics.

1 Introduction

The growing demand of smart pervasive applications capable of monitoring, filtering and analyzing in a timely manner the large volume of data flows produced by

I. Mongiardo, L. Massari, M. Calzarossa
Department of Electrical, Computer and Biomedical Engineering, University of Pavia, Italy
e-mail: ivangiuseppe.mongiardo01@universitadipavia.it, luisa.massari@unipv.it, mcc@unipv.it

B. Bermejo
Department of Computer Science, University of Balearic Islands, Spain
e-mail: belen.bermejo@uib.es

D. Tessera
Department of Mathematics and Physics, Catholic University of Sacred Heart, Italy
e-mail: daniele.tessera@unicatt.it

IoT devices opens new challenges. To ensure low and predictable delays to these applications, it is necessary to employ computation resources in the proximity of the data sources, that is, at the edge of the network [5]. Nevertheless, due the limited processing capacity of these resources, a seamless integration of cloud resources is required, thus a major driver towards the development of these innovative solutions is represented by the cloud-edge continuum [9]. These solutions strongly depend on the characteristics of the workloads being processed [6, 7].

In this paper, we explore the issues related to application placement in the cloud-edge continuum. In particular, we consider applications consisting of multiple interdependent lightweight modules organized according to the Distributed Data Flow (DDF) model and an infrastructure consisting of multiple resources organized in hierarchical layers. We propose a lightweight placement policy that takes into account the volume of data exchanged between modules and allocates a given module to the "best" resource, that is, the resource that ensures the minimum network delay. The experimental results have shown that our heuristic generally outperforms state-of-the-art heuristics, although its performance mainly depends on the characteristics of the applications being placed.

The rest of the paper is organized as follows. In Section 2 we discuss the state of the art in the area of application placement. In Section 3 we describe the proposed heuristic, while in Section 4 we present the experimental setup and results. In Section 5 we summarize the main outcomes and outline possible research directions.

2 Related work

This section presents a brief review the state-of-the-art regarding placement heuristic in the cloud-edge continuum. During the last years, several reviews have been performed (e.g., [2, 4, 12, 14, 16, 17]). These reviews covered various aspects, such as types of IoT application models and strategies for resource allocation, service placement problem, optimal application placement, architectural maintenance and AI-based heuristics.

In the context of placement heuristics, most papers focus on the minimization of various objectives, such as application execution time, resource usage, network latency, energy consumption. To achieve these objectives, different placement techniques have been proposed, mainly applied to fog/edge/cloud architecture. For example, in [18] authors proposed a module mapping algorithm for efficient resource usage in fog/cloud environments. The algorithm considers modules and devices sorted in ascending order with respect to their requirements and capacity, respectively. The mapping of modules to devices follows this order starting from the lightest module that is placed on the least powerful device. Similarly, the algorithm for module placement presented in [3] focuses on reducing the total network usage and the application execution time. According to this algorithm, the shortest modules in terms of processing requirements are placed on the available fog device located in the bottom-most layer.

A different placement approach was presented in [13] where a latency-aware module placement in distributed fog environments was proposed. This approach aims at satisfying service delivery deadlines for latency-sensitive and latency-tolerant applications. Modules can be placed either horizontally or vertically depending on the characteristics of the applications. Interdependent application modules characterized by a high data interaction rate were also considered in [8], where authors proposed a method that tries to place as many modules as possible in the same fog node with the aim of minimizing the network delay.

Other placement heuristics focus on aspects dealing with the impact of the communication requirements of the applications. This is the case of the Communication-based policy proposed in [15] where the placement is based on the minimization of the data volume transferred by each module. More precisely, the modules with the minimum communication impact are placed on the highest layer of the cloud-fog computing infrastructure, thus reducing the network impact on the application delay.

A QoS-aware Greedy-edge placement scheme aimed at minimizing the end-to-end latency of real-time IoT applications in fog environments was proposed in [1]. The approach first assigns to each module a score obtained by combining its processing requirement with a priority that is set starting from the first module. Modules are then sorted in descending order according to their score. Similarly, the fog devices are sorted in ascending order according to their proximity to the IoT layer. Modules with the highest score are placed first by choosing the closest fog node that satisfies their requirements.

This policy, together with the Communication-based policy previously described, will be used as baselines in our experiments because they consider aspects similar to ours.

As a conclusion, to the best of our knowledge, it is important to devise a lightweight heuristic that considers complementary aspects dealing with the data volume exchanged between the application modules together with the processing capacity of various devices.

3 Placement problem

The main goal of placement policies is to identify the most appropriate mapping between application modules and devices. These policies are usually driven by different objectives, e.g., minimizing communication delays, reducing the number of allocated devices, ensuring performance fairness across applications. In what follows we describe the application and architecture models considered in this study and the proposed placement policy.

3.1 Application and architecture models

We consider applications consisting of multiple inter-dependent modules organized according to the Distributed Data Flow model [10]. In this context, modules communicate by pushing and pulling data, thus triggering the data flows. More precisely, a module receives requests for processing the inputs generated by upstream modules and for producing outputs to be stored locally or to be sent to downstream modules. The only exceptions are represented by the modules at the start and at the end of the flow which only produce outputs or consume inputs, respectively.

Figure 1(a) shows an example of an application consisting of seven modules, including the so-called client module which receives the data being sensed and triggers actuators by sending the results of the processing.

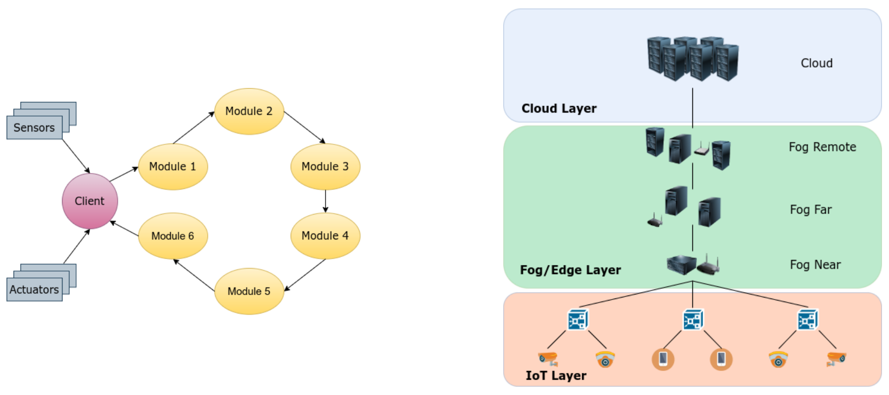


Fig. 1 Models of the applications (a) and of the cloud-edge continuum (b) considered in this study.

We model the cloud-edge continuum as a hierarchical architecture composed of three layers, namely, IoT, fog/edge and cloud layers. An example of such an architecture is sketched in Figure 1(b). As can be seen, the fog/edge layer is characterized by a hierarchy of resources whose latency and processing capacity increase as the proximity from the IoT devices decreases.

3.2 Placement policy

Our placement policy aims at minimizing the network delays over a set of applications by considering the network proximity of the upstream and downstream modules of a given application and placing one module at a time. This minimization problem is addressed by a lightweight heuristic. In fact, unlike optimization problems, heuristics are typically characterized by small computation requirements, thus they can provide timely placement decisions.

For our policy, we assume that placement starts from the first module of each application and proceeds one module at a time until the last module. We also assume that there are no network delays whenever a downstream module is placed in the same device of its upstream counterpart.

For each application the proposed policy focuses on pairs of modules (i.e., a module and its downstream counterpart) and searches for devices able to cope with the processing requirements of both. Under the assumption that module $i-1$ is placed in device k, the policy computes the network delays $D^i_{k,q}$ associated with the previously identified devices as:

$$D^i_{k,q} = L_{k,q} + \frac{V_{i-1,i}}{B_{k,q}}$$

where $L_{k,q}$ and $B_{k,q}$ denote the network latency and bandwidth between devices k and q, while $V_{i-1,i}$ the volume of output data sent by module $i-1$ to module i. The device with the minimum network delay is chosen as a possible candidate for placing modules i and $i+1$.

According to our heuristic, module i is placed on the identified device. This device might have insufficient processing capacity when module $i+1$has actually to be placed, thus a re-evaluation of the placement decision has to be performed. In fact, placement decisions follow a rotation scheme across applications without implementing any resource reservation mechanism. For example, the first modules of all applications are placed before considering the second modules. As a consequence, these re-evaluations will increase the network delays because it will be necessary to place the module in a different device.

In summary, the placement decisions taken by the proposed heuristic are based on two complementary aspects dealing with the network delay that might be experienced by individual modules as well as the processing capacity to be provided in the cloud-edge continuum. These decisions ensure fairness across applications because of the rotation scheme.

4 Experimental results

To evaluate the proposed policy, we performed several experiments in a simulated cloud-edge environment based on the iFogSim toolkit [11]. The simulated environment consists of devices organized according to the layers displayed in Figure 1(b) whose characteristics are summarized in Table 1. Processing capacity refers to individual devices, whereas bandwidth and latency refer to adjacent devices. For example, the fog far device can exploit a bandwidth equal to 25 Mbps to communicate with the for near device, whereas the bandwidth between the cloud and the fog re-

Table 1 Characteristics of the architecture considered in the experiments.

	Processing Capacity [MIPS]	Bandwidth [Mbps]	Latency [ms]
Cloud	400,000	10	70
Fog Remote	25,000	20	30
Fog Far	20,600	25	8
Fog Near	10,600	30	2
Gateway	1,000	5	1

mote device is equal to 10 Mbps. The gateway is dedicated to host the client modules only, thus it will not be considered in the placement decisions.

To test the proposed policy, we initially focus on a toy application composed of six modules plus the client module whose characteristics have been chosen such as the policy needs to perform multiple re-evaluations. These characteristics are presented in Table 2. The output data volume refers to data exchanged by a module

Table 2 Resource requirements of the modules of the toy application.

	Processing Requirement [MIPS]	Output Data Volume [KBytes]
Client	200	100
Module 1	1,000	100
Module 2	2,000	50
Module 3	2,500	80
Module 4	200	80
Module 5	500	50
Module 6	200	20

with its downstream counterpart. Note that the client module is the downstream module of Module 6.

The performance of our policy as a function of the number of applications being placed is shown in Figure 2. Performance refers to the network delay computed as an average over the applications being placed. The figure plots the network delays

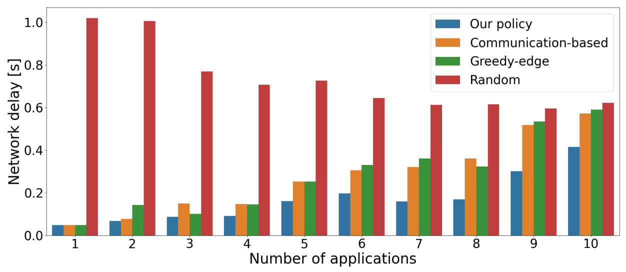


Fig. 2 Average network delays of the applications as a function of the placement policies and of the number of applications being placed.

experienced by applications placed according to our policy and to two state-of-the-art policies, namely, Communication-based and Greedy-edge (see Section 2). For comparison purposes, we also include the results obtained with a random placement. As expected, the network delay increases with the number of applications, the only exception is the random placement whose average delay decreases because of the fewer available resources to choose from. In general, our policy outperforms the others even significantly, despite the re-evaluations that might be necessary because of the insufficient processing capacity on the device previously identified. For example, in these experiments the fraction of these re-evaluations reaches the 46% of the placement decisions.

To further assess the performance of our policy, we consider bigger applications, i.e., consisting of ten modules, whose processing requirements and output data volume are sampled from uniform distributions. Figure 3 displays the average values of the module characteristics together with the corresponding confidence intervals computed at the 95% confidence level.

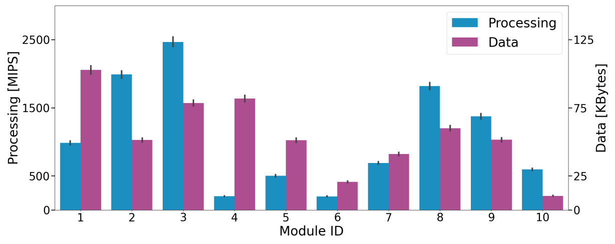
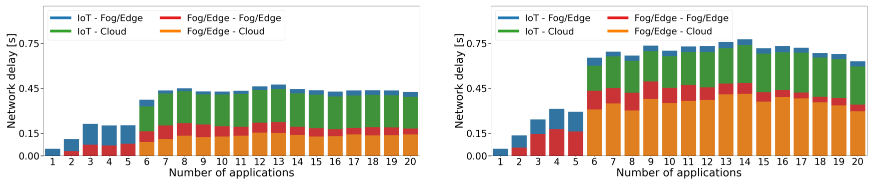


Fig. 3 Average processing requirement and output data volume of each module. Error bars represent the 95% confidence intervals.

The network delays as a function of the number of applications being placed for our policy and the Communication-based policy are shown in Figure 4. Similarly to the previous experiments, these results suggest that our policy outperforms the other. The figure also shows the breakdown of the network delays as a function of

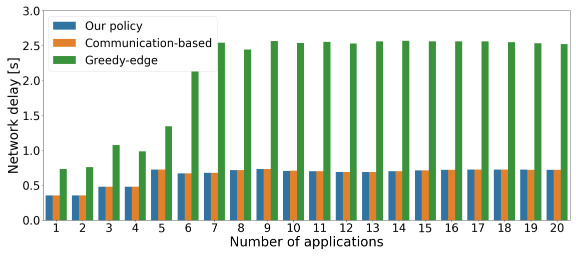


(a) Our policy (b) Communication-based policy

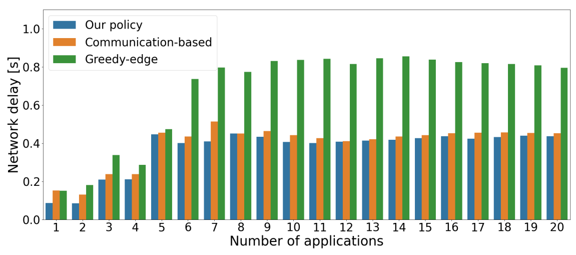
Fig. 4 Breakdown of the network delays as a function of the cloud-edge continuum layers and of the number of applications being placed.

cloud-edge continuum layers. Unlike our policy, the Communication-based policy tends to place modules across layers, thus it is affected by significant delays mainly between the fog/edge and cloud layers.

Another set of experiments focuses on assessing the impact of the policies on the network delay when changing the communication patterns. We consider an application with ten modules exchanging, in total about 5.6 MBytes of data, ranging from 50 KBytes up to 1 MBytes per module. Figure 5 shows the results of these experiments where data volume decreases or increases with the modules, starting from the first one. The figure suggests that, unlike our policy, the start-of-the-art policies



(a) Decreasing data volume

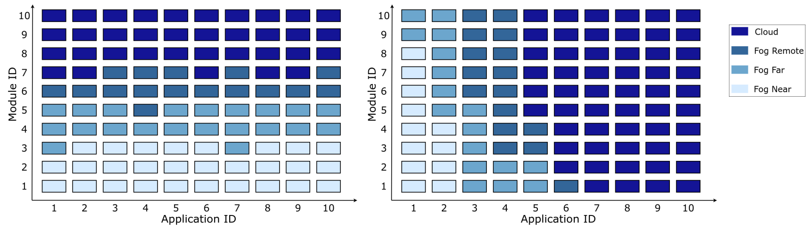


(b) Increasing data volume

Fig. 5 Network delays, as a function of the number of applications, with decreasing (a) and increasing (b) data volume patterns.

considered in this paper are affected by communication patterns, sometimes even considerably.

Finally, we analyze in Figure 6 the mappings between the modules of ten identical applications and the cloud-edge devices obtained when applications are considered all together or one by one starting from their first module. As can be seen, our policy tries to exploit the devices of all layers for all applications independently of their order, thus ensuring fair placements. On the contrary, when applications are considered one by one, their order matters, thus only few applications benefit of the devices located at the fog/edge layer.



(a) (b)

Fig. 6 Mappings between modules and devices obtained with the proposed policy by considering applications all together (a) and one by one (b).

5 Conclusion

In this paper we proposed a lightweight placement policy that considers complementary aspects, that is, the network delay experienced by individual application modules and the processing capacity provided in the cloud-edge continuum. This heuristic generally outperforms state-of-the-art policies. In addition, our experiments have shown that, to minimize the network delays, placement decisions should adapt to the resource requirements of the applications, e.g., communication patterns.

As future research directions, we plan to further analyze the performance of the proposed policy using different mixes of heterogeneous applications. In addition, we will study more sophisticated policies that easily adapt to application requirements and to the cloud-edge continuum characteristics.

Acknowledgments

This work was partly supported by the Italian Ministry of University and Research (MUR) under the PRIN 2022 grant "Methodologies for the Parallelization, Performance Evaluation and Scheduling of Applications for the Cloud-Edge Continuum" (Master CUP: B53D23013090006, CUP: J53D23007110008, CUP: F53D23004300006) and by European Union - Next Generation EU.

References

1. Abu-Amssimir, N., Al-Haj, A.: A QoS-aware resource management scheme over fog computing infrastructures in IoT systems. Multimedia Tools and Applications **82**(18) (2023)
2. Apat, H., Nayak, R., Sahoo, B.: A comprehensive review on Internet of Things application placement in Fog computing environment. Internet of Things **23** (2023)
3. Arora, U., Singh, N.: IoT application modules placement in heterogeneous fog–cloud infrastructure. International Journal of Information Technology **13**(5), 1975–1982 (2021)

4. Bermejo, B., Juiz, C.: Improving cloud/edge sustainability through artificial intelligence: A systematic review. Journal of Parallel and Distributed Computing (2023)
5. Bonomi, F., Milito, R., Zhu, J., Addepalli, S.: Fog Computing and Its Role in the Internet of Things. In: Proceedings of the First Workshop on Mobile Cloud Computing, MCC, p. 13–16. Association for Computing Machinery (2012)
6. Calzarossa, M., Della Vedova, M., Massari, L., Petcu, D., Tabash, M., Tessera, D.: Workloads in the Clouds. In: L. Fiondella, A. Puliafito (eds.) Principles of Performance and Reliability Modeling and Evaluation, Springer Series in Reliability Engineering, pp. 525–550. Springer (2016)
7. Calzarossa, M.C., Massari, L., Tessera, D.: Workload characterization: A survey revisited. ACM Computing Surveys **48**(3), 48:1–48:43 (2016)
8. Dadashi Gavaber, M., Rajabzadeh, A.: BADEP: Bandwidth and delay efficient application placement in fog-based IoT systems. Transactions on Emerging Telecommunications Technologies **32** (2021)
9. Esposito, A., Aversa, R., Barbierato, E., Calzarossa, M., Di Martino, B., Massari, L., Mongiardo, I., Tessera, D., Venticinque, S., Zanussi, L., Zieni, R.: Methodologies for the Parallelization, Performance Evaluation and Scheduling of Applications for the Cloud-Edge Continuum. In: L. Barolli (ed.) Advanced Information Networking and Applications, AINA. Springer (2024)
10. Giang, N., Blackstock, M., Lea, R., Leung, V.: Developing IoT applications in the Fog: A Distributed Dataflow approach. In: Proceedings of the 5th International Conference on the Internet of Things, IOT, pp. 155–162 (2015)
11. Harshit, G., Dastjerdi, A., Ghosh, S., Buyya, R.: iFogSim: A toolkit for modeling and simulation of resource management techniques in the Internet of Things, Edge and Fog computing environments. Software: Practice and Experience **47**, 1275–1296 (2017)
12. Islam, M.M., Ramezani, F., Lu, H.Y., Naderpour, M.: Optimal Placement of Applications in the Fog Environment: A Systematic Literature Review. Journal of Parallel and Distributed Computing **174**(C), 46–69 (2023)
13. Mahmud, R., Ramamohanarao, K., Buyya, R.: Latency-Aware Application Module Management for Fog Computing Environments. ACM Transactions on Internet Technology **19**(1) (2018)
14. Mahmud, R., Ramamohanarao, K., Buyya, R.: Application Management in Fog Computing Environments: A Taxonomy, Review and Future Directions. ACM Computing Surveys **53**(4) (2020)
15. Peixoto, M.L.M., Genez, T.A., Bittencourt, L.F.: Hierarchical scheduling mechanisms in multi-level fog computing. IEEE Transactions on Services Computing **15**(5), 2824–2837 (2021)
16. Salaht, F., Desprez, F., Lebre, A.: An Overview of Service Placement Problem in Fog and Edge Computing. ACM Computing Surveys **53**(3) (2020)
17. Smolka, S., Mann, Z.Á.: Evaluation of fog application placement algorithms: a survey. Computing **104**(6), 1397–1423 (2022)
18. Taneja, M., Davy, A.: Resource aware placement of IoT application modules in Fog-Cloud Computing Paradigm. In: Proceedings of the IFIP/IEEE Symposium on Integrated Network and Service Management, IM, pp. 1222–1228. IEEE (2017)