Energy and QoE aware Placement of Applications and Data at the Edge

Matteo Mordacchini[1], Luca Ferrucci[2], Emanuele Carlini[2], Hanna Kavalionak[2], Massimo Coppola[2] and Patrizio Dazzi[3]

[1]IIT-CNR, Pisa, 56124, Italy
[2]ISTI-CNR, Pisa, 56124, Italy
[3]Department of Computer Science, University of Pisa, 56127, Italy

Abstract

Recent years are witnessing extensions of cyber-infrastructures towards distributed environments. The Edge of the network is gaining a central role in the agenda of both infrastructure and application providers. Following the actual distributed structure of such a computational environment, nowadays, many solutions face resource and application management needs in Cloud/Edge continua. One of the most challenging aspects is ensuring highly available computing and data infrastructures while optimizing the system's energy consumption. In this paper, we describe a decentralized solution that limits the energy consumption by the system without failing to match the users' expectations, defined as the services' Quality of Experience (QoE) when accessing data and leveraging applications at the Edge. Experimental evaluations through simulation conducted with PureEdgeSim demonstrate the effectiveness of the approach.

Keywords

Edge Computing, Self-organization, QoE

1. Introduction

Cloud solutions are spreading and getting momentum, being used in a vast majority of fields and environments. In spite of the large availability and variety of cloud solutions, we witness increasing difficulties in coping with next generation applications, like latency-sensitive ones. Edge/Cloud continua have all the possibilities to overcome these limits by seamlessly integrating one (or more) Cloud(s) and large numbers of Edge resources that are geographically distributed. Despite being conceptually feasible, many challenges emerge for the actual management, coordination and optimization of these huge sets of heterogeneous and dispersed resources [1, 2, 3]. A key issue is the efficient management of the overall energy consumption, memory and computational resource usage of the system. As a matter of fact, one way to achieve these results

SEBD 2022: The 30th Italian Symposium on Advanced Database Systems, June 19-22, 2022, Tirrenia (PI), Italy
matteo.mordacchini@iit.cnr.it (M. Mordacchini); luca.ferrucci@isti.cnr.it (L. Ferrucci); emanuele.carlini@isti.cnr.it (E. Carlini); hanna.kavalionak@isti.cnr.it (H. Kavalionak); massimo.coppola@isti.cnr.it (M. Coppola); patrizio.dazzi@unipi.it (P. Dazzi)
https://www.iit.cnr.it/matteo.mordacchini/ (M. Mordacchini)
0000-0002-1406-828X (M. Mordacchini); 0000-0003-4159-0644 (L. Ferrucci); 0000-0003-3643-5404 (E. Carlini); 0000-0002-8852-3062 (H. Kavalionak); 0000-0003-4159-0644 (M. Coppola); 0000-0001-8504-1503 (P. Dazzi)

is to optimize the placement of the instances of the applications requested by the users, by taking into account the functional needs of the applications, the computational limits of Edge resources and the non-functional requirements associated with the users' Quality of Experience (QoE). Distributed [4, 5], self-organizing [6, 7] and adaptive [8, 9] solutions have been proposed for facing such challenges at the Edge. In this paper, we describes a decentralized, self-organizing and QoE-centric scheme for the optimization of the energy consumed by the system. The approach enables Edge entities to interact with one another to exchange information in an attempt to determine whether the users of each application can be served using a lower number of instances. This behaviour allows to reduce the number of instances executed in the system, thus reducing the energy consumed and resource usage. When taking the decision to shut down a potential redundant instance, the entities exploit the data they have exchanged to evaluate whether this decision is in accordance with the services' QoE and the computational limits of Edge resources. Experimental results through simulation show that the proposed solution is able to reduce the energy required by the system up to nearly 40%. The rest of this paper is organized as follows. Section 2 contextualizes this work in the related scientific literature. Section 3 presents our the definition of the problem and the approach we propose. Section 4 describes the experimental evaluation of the proposed solution. Finally, Section 5 draws concluding remarks and highlights future work directions.

2. Related Work

Edge-based systems are the object of many researches that try to optimize the overall performances of a system by limiting the communications with centralized Clouds [10, 11, 12, 13, 14, 15]. In fact, data exchange between Cloud and Edge systems introduce significant overhead and could potentially degrade the performance of applications running at the Edge, like locality and context-based services. A well-known approach to overcome this problem is to use decentralized and/or self-organizing solutions [16, 17]. These solutions achieve their goal by moving the applications [18, 19, 20] and/or data closer to users. When the data is moved in the system, the aim is to make it easy for the users to access it [21, 22]. In this case, the general strategy is to shorten the distance between the data storage devices or the data producers and their respective consumers [23, 24]. To achieve an optimization of the energy consumption levels of the entities at the Edge, we use a method which does not move data and/or applications closer to each other and/or closer to their users. Beraldi *et al.* propose CooLoad [25], a scheme where Edge datacenters re-direct their requests to other adjacent data centers whenever they become congested. Carlini *et al.* [14] propose a decentralized system, where autonomous entities in a Cloud Federation communicate to exchange computational services, trying to maximize the profit of the whole Federation. Differently from the previous solutions, in this paper, the efficient exploitation of the resources at the Edge is obtained by optimizing the energy consumption of the system as a whole. As we explain in depth in the rest of the paper, this result is achieved through point-to-point interactions between Edge entities, known as Edge Miniclouds (EMCs). These entities use their communications to detect potential redundant instances of the applications requested by the users. As a result, the users are directed to use only a limited set of instances, thus allowing to shut down the others. However, an user request could be

served by a different instance running on another EMC only if the associated QoE constraints remain satisfied. The outcome of the collective behaviour of the entities at the Edge is a notable reduction of the energy needed by the system performing the computational tasks requested by its users.

3. Problem Definition and Proposed Solution

Consistently with the definitions of the EU ACCORDION project [1], we consider that the system at the Edge is a federation of so-called *Edge mini-clouds* (EMCs). Each EMC is an entity that supervises a set of other devices with limited resources, like IoT devices, sensors, etc. Applications are sent to an EMC, which is in charge to orchestrate their execution among the devices it controls. We consider that $E = \{EMC_1, \ldots, EMC_m\}$ is the set of all the EMCs in the system, with $|E| = M$. The set $A = \{A_1, \ldots, A_n\}$ is the set of all the types of applications that can be executed in the system, with $N = |A|$. Each $A_i \in A$ represents a distinct type of service, with specific requirements in term of resources. In order to meet the requests of the users, several instances of an application A_i can be deployed on the EMCs. The symbol a_{ij} denotes the instance of the application A_i executed by EMC_j. Running a_{ij} implies a resource occupancy equal to w_{ij}. This weight is composed of a base weight w_i^{fix} and a variable component w_{ij}^{var}, where the variable component depends on the number of users served by a_{ij}. Therefore, if we denote with u_{ij} the number of users of a_{ij}, we have that $w_{ij}^{var} = u_{ij} w_i^u$, where w_i^u is the weight-per-user of A_i. Thus, $w_{ij} = w_i^{fix} + u_{ij} w_i^u$. The overall number of users served by all the instances of A_i is U_i, while W_j is the maximum weight that can be supported by EMC_j for running all the instances that are assigned to it. In addition to the functional requirements, in order to meet the required QoE, each application has also additional non-functional requirements. These requirements limit the EMCs where an instance can be deployed. QoE is expressed in terms of latency that each service A_i constraints to be lower than a value L_i. In fact, latency is one of the main factors that influence a user's perception of the quality of a service. We assume that each time a user requests a service, an instance of such service is activated in the user's closest EMC (in terms of latency), thus latency is initially minimized but this allocation scheme also generate a set of redundant instances of the same service. In fact, users initially assigned to different EMCs could be served by just one, properly selected instance, without violating the service's QoE. This allows to shut down the other instances and reduce the amount of energy consumed for serving the same users. Always relaying on the direct intervention of a distant Cloud orchestrator to reach this result could be a source of delay and degradation of the QoE. To overcome this limit, we developed an adaptive decentralized approach that identifies and stop redundant instances of the running applications. In the next, following the pseudocode given in Algorithm 1, we describe the steps executed by a generic EMC_j. EMC_j has a set $\mathcal{N}$ of neighboring EMCs (EMCs within the communication range of EMC_j). The latency between EMC_j and any of its neighbors $EMC_k \in \mathcal{N}$ is $\mathcal{L}(j,k)$. I_j^t is the set of application types running on the instances on EMC_j at time t. Each application $A_i \in I_j^t$ has a maximum agreed latency L_i, and a set of users u_{ij}, which experiences a maximum latency l_{ij}. At regular time intervals, EMC_j randomly chooses one neighbor $EMC_k \in \mathcal{N}$.It

[1] https://www.accordion-project.eu/

Algorithm 1 Actions performed by a generic EMC_j at each time step t

Input: $\mathcal{N}$ = set of neighbors of EMC_j
Randomly choose $EMC_k \in \mathcal{N}$
Request I_k^t, W_k, W_k^t to EMC_k
Compute $I_{jk} = \{A_i | A_i \in I_j^t \cap I_k^t\}$
if $I_{jk} \neq \emptyset$ **then**
  if $W_j^t \geq W_k^t$ **then**
    $\mathcal{A}_{jk} = \{A_i \in I_{jk} | l_{ij} + \mathcal{L}(j,k) \leq L_i\}$
    Order $\mathcal{A}_{jk}$ in ascending order using w_{ij}
    Let m be the index of the first application $A_m \in \mathcal{A}_{jk}$ s.t. $W_k^t + w_m^u u_{mj} \leq W_k$
    Direct the users of a_{mj} to use a_{mk}
    Turn off a_{mj}
  else
    $\mathcal{A}_{kj} = \{A_i \in I_{jk} | l_{ik} + \mathcal{L}(j,k) \leq L_i\}$
    Order $\mathcal{A}_{kj}$ in ascending order using w_{ik}
    Let m be the index of the first application $A_m \in \mathcal{A}_{kj}$ s.t. $W_j^t + w_m^u u_{mk} \leq W_j$
    Direct the users of a_{mk} to use a_{mj}
    Tell EMC_k to turn off a_{mk}
  end if
end if

then asks EMC_k for the list of its running applications I_k^t with their number of users, its maximum capacity W_k and its actual resource occupancy W_k^t. The solution tries to gather the users of both the instances on the EMC with the lowest actual occupancy. If $I_{jk} \neq \emptyset$, s is the *source* EMC (the one from which the users will be moved), with d the EMC receiving that users. Thus, $W_s^t \geq W_d^t$. EMC_j builds a set $\mathcal{A}_{sd} = \{A_i \in I_{sd} | l_{is} + \mathcal{L}(s,d) \leq L_i\}$ containing the instances whose users can be moved without violating the QoE. EMC_j traverses $\mathcal{A}_{sd}$ in descending order (on the basis of the instances weights in EMC_s), choosing for the exchange the first application whose users can be transferred without exceeding W_d. In the special case where both the EMCs select each other for an exchange, having equal loads and selecting the same service, the EMC with the lowest ID rejects to receive the exchange, asking for another application. Once the users are directed to another instance, the instance on EMC_s is turned off. This action allows both to save energy and to free space for other potential exchanges or new instances.

4. Experimental Evaluation

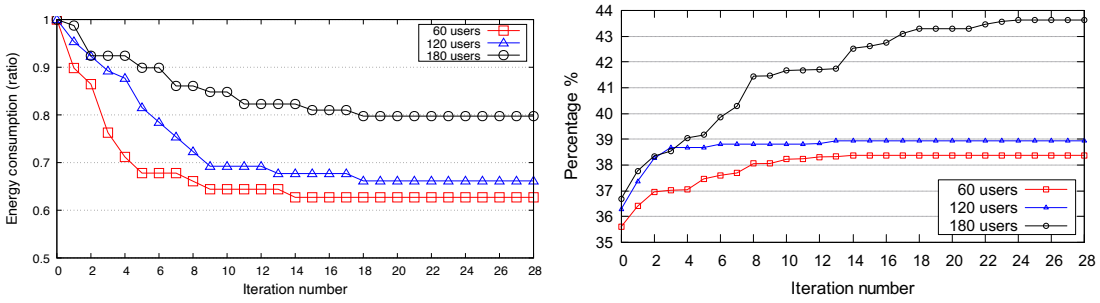
The experimental evaluation has been conducted through a simulation of a target scenario using PureEdgeSim [26], a discrete-event simulator for Edge environments, that well matches the EMC-based structure of our scenario, allowing also to measure energy consumptions. At the beginning of the simulation, each user requests a single application to its closest EMC. In case an instance of the requested application type already exists on that EMC, the user

Table 1
Values of fixed weights w^{fix} and per-user w^u for each application type and resource

App. type	VCPU	Ram	BW	VCPU (per user)	RAM (per user)	BW (per user)
Balanced	1	200	20	1 each 10 users	20	2
Comp Bound	2	200	20	1 each 5 users	20	2
Mem Bound	1	400	20	1 each 10 users	40	2
I/O Bound	1	200	40	1 each 10 users	20	4

is simply added to the instance's local set of users. In the experiments the number of users varies in the set $\{60, 120, 180\}$. Each user device is placed randomly in a bi-dimensional area of 200×200 metres. In all the experiments the number of EMCs is fixed to 4. They are placed at predefined locations inside the simulation space. We assume that any EMC can host any type of application. Moreover, each EMC is able to communicate with the others. There are three types of resources available in the system (at the EMCs): the number of VCPU; the amount of RAM; the amount of network bandwidth (BW). In the simulations, we use four different types of applications. Application types differ on the resources they request and, as a consequence, the energy footprint they produce when their instances are executed. The application types are divided as *Computational Bound* (i.e, computational intensive), *Memory Bound* (memory intensive) and *I/O Bound* (networking intensive) applications, where "intensive" means having double the requirements of the basic *Balanced* application type. The load of an EMC is calculated as the mean of the percentage of availability of the three resources. The fixed weight w_i^{fix} and the weights per user w^u associated with the different application types is shown in Table 1. RAM and BW are in Mbytes and Mbit/s, respectively. In addition to these parameters, we also use three different values for the maximum application latency L_i: 0.2, 0.3, 0.5 seconds. Therefore, we have 12 possible combinations of parameters for the applications: 4 types of applications times 3 different latency constraints. All the results presented in the next are the average of 10 independent runs. The first and main result of our evaluation is presented in Figure 1a. This figure presents the evolution over time of the energy required by all the EMCs in the system, including the energy needed for inter-EMCs communications. The results are presented as the ratio between the energy needed at a time $t > 0$ and the energy consumed by the system at the beginning of the simulation. It is possible to observe that the level that is required to serve the very same number of users drops by a minimum of 20% (with 180 users) up to nearly 40% (with 60 users); this drastic reduction in the energy footprint of the system demonstrates the high level of efficiency of the proposed approach.

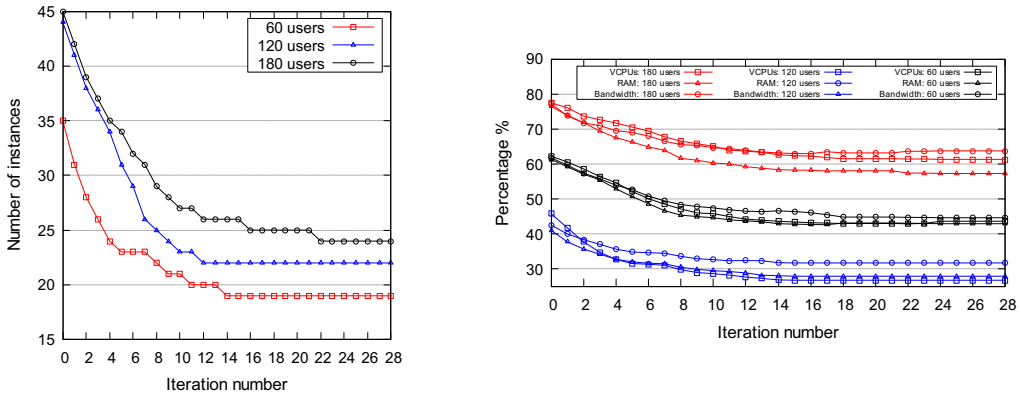
Figure 1b depicts how the configurations adopted by the system are able to remain compliant with the applications' QoE. A simulated latency function is calculated for each user's device, which is the composition of a fixed part, which is dependent from the communication channel type, and a linear part, proportional to the Euclidean distance between the EMC hosting the instance of the serving application and the user's device. The average latency is measured as the percentage of the maximum average latency, as constrained by the applications' limits. It is possible to note that there is only a limited increase on the average latency. Therefore, the proposed solution shows its ability to remarkably reduce the energy needed to run the instances



(a) (b)

Figure 1: Temporal evolution of the levels of energy (a) and average latency (b)

that serve a given population of users, while remaining well below the limits of the required QoE. In order to better understand how these results are achieved, the next set of figures analyses how the system collectively adapts its behaviour and how it changes the exploitation of the available resources. Figure 2a presents the variation over time of the number of running instances in the system. Clearly, these entities are the source of energy consumption. The ability of the system to detect and eliminate redundant instances is the basis for the energy minimization scheme. It is possible to observe a clear and sharp decrease of this quantity. The final number is nearly the half of the original number of instances. Figure 2b presents the global level of exploitation



(a) (b)

Figure 2: Variation of the total number of instances (a) and system resource loads (b), over time

of the resources. The y axis presents the percentage of all the resources that are required to run the application instances. As in the previous case, we can observe a clear reduction. The amount of this reduction is lower than that of the number of instances, since users are moved from a redundant instance to an active one. As we highlighted in Section 3, each user bring an additional cost in terms of resources. Despite this fact, the overall level of occupied resources is decremented, since the fixed costs needed for running redundant instances are saved.

5. Conclusions

This paper presents a solution for application placement performing edge-to-edge exchanges to reduce the energy consumption and resource usage, while guaranteeing the QoE of applications by keeping the communication latency below given thresholds. The paper provides a definition of the problem and the pseudo code of the proposed approach. An experimental evaluation via simulation shows the validity of our solution. While the solution is quite a promising one, there is space to improve the results in the near future. It is worth e.g. considering alternative local search criteria and heuristics for the selection criteria of the EMC and application for the swap proposal. This may improve the asymptotic cost savings and is likely to improve the achieved savings as well as the convergence speed of our algorithm.

References

[1] C.-H. Youn, M. Chen, P. Dazzi, Cloud Broker and Cloudlet for Workflow Scheduling, Springer, 2017.

[2] T. Taleb, B. Samdanis, Kand Mada, H. Flinck, S. Dutta, D. Sabella, On multi-access edge computing: A survey of the emerging 5g network edge cloud architecture and orchestration, IEEE Communications Surveys Tutorials 19 (2017) 1657–1681.

[3] N. Kumar, A. Jindal, M. Villari, S. N. Srirama, Resource management of iot edge devices: Challenges, techniques, and solutions, Software: Practice and Experience 51 (2021) 2357–2359. doi:https://doi.org/10.1002/spe.3006.

[4] G. F. Anastasi, E. Carlini, P. Dazzi, Smart cloud federation simulations with cloudsim, in: Proceedings of the first ACM workshop on Optimization techniques for resources management in clouds, 2013, pp. 9–16.

[5] M. Marzolla, M. Mordacchini, S. Orlando, A p2p resource discovery system based on a forest of trees, in: 17th International Workshop on Database and Expert Systems Applications (DEXA'06), IEEE, 2006, pp. 261–265.

[6] L. Ferrucci, L. Ricci, M. Albano, R. Baraglia, M. Mordacchini, Multidimensional range queries on hierarchical voronoi overlays, Journal of Computer and System Sciences (2016).

[7] M. Conti, M. Mordacchini, A. Passarella, L. Rozanova, A semantic-based algorithm for data dissemination in opportunistic networks, in: 7th International Workshop on Self-Organizing Systems (IWSOS 2013), LNCS 8221, Springer, 2013, pp. 14–26.

[8] R. Baraglia, P. Dazzi, B. Guidi, L. Ricci, Godel: Delaunay overlays in p2p networks via gossip, in: IEEE 12th International Conference on Peer-to-Peer Computing (P2P), IEEE, 2012, pp. 1–12.

[9] M. Mordacchini, A. Passarella, M. Conti, S. M. Allen, M. J. Chorley, G. B. Colombo, V. Tanasescu, R. M. Whitaker, Crowdsourcing through cognitive opportunistic networks, ACM Transactions on Autonomous and Adaptive Systems (TAAS) 10 (2015) 1–29.

[10] F. Salaht, F. Desprez, A. Lebre, An overview of service placement problem in fog and edge computing, ACM Comput. Surv. 53 (2020).

[11] G. Z. Santoso, Y.-W. Jung, S.-W. Seok, E. Carlini, P. Dazzi, J. Altmann, J. Violos, J. Marshall,

Dynamic resource selection in cloud service broker, in: 2017 Int. Conf. on High Perform. Comput. Simul. (HPCS), IEEE, 2017, pp. 233–235.
[12] U. Ahmed, A. Al-Saidi, I. Petri, O. F. Rana, Qos-aware trust establishment for cloud federation, Concurrency and Computation: Practice and Experience 34 (2022) e6598.
[13] J. Altmann, B. Al-Athwari, E. Carlini, M. Coppola, P. Dazzi, A. J. Ferrer, N. Haile, Y.-W. Jung, J. Marshall, E. Psomakelis, et al., Basmati: an architecture for managing cloud and edge resources for mobile users, in: International Conference on the Economics of Grids, Clouds, Systems, and Services, Springer, Cham, 2017, pp. 56–66.
[14] E. Carlini, M. Coppola, P. Dazzi, M. Mordacchini, A. Passarella, Self-optimising decentralised service placement in heterogeneous cloud federation, in: 2016 IEEE 10th International Conference on Self-adaptive and Self-organizing Systems (SASO), IEEE, 2016, pp. 110–119.
[15] G. F. Anastasi, M. Coppola, P. Dazzi, M. Distefano, Qos guarantees for network bandwidth in private clouds, Procedia Computer Science 97 (2016) 4–13.
[16] M. Mordacchini, M. Conti, A. Passarella, R. Bruno, Human-centric data dissemination in the iop: Large-scale modeling and evaluation, ACM Trans. Auton. Adapt. Syst. 14 (2020).
[17] C. Gennaro, M. Mordacchini, S. Orlando, F. Rabitti, Mroute: A peer-to-peer routing index for similarity search in metric spaces, in: 5th VLDB International Workshop on Databases, Information Systems and Peer-to-Peer Computing (DBISP2P 2007), 2007.
[18] Z. Ning, P. Dong, X. Wang, S. Wang, X. Hu, S. Guo, T. Qiu, B. Hu, R. Kwok, Distributed and dynamic service placement in pervasive edge computing networks, IEEE Transactions on Parallel and Distributed Systems (2020).
[19] A. M. Maia, Y. Ghamri-Doudane, D. Vieira, M. F. de Castro, Optimized placement of scalable iot services in edge computing, in: 2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM), 2019, pp. 189–197.
[20] P. Dazzi, M. Mordacchini, Scalable decentralized indexing and querying of multi-streams in the fog, Journal of Grid Computing 18 (2020) 395–418.
[21] E. Carlini, M. Coppola, P. Dazzi, D. Laforenza, S. Martinelli, L. Ricci, Service and resource discovery supports over p2p overlays, in: 2009 International Conference on Ultra Modern Telecommunications & Workshops, IEEE, 2009, pp. 1–8.
[22] M. Mordacchini, P. Dazzi, G.Tolomei, R. Baraglia, F. Silvestri, S. Orlando, Challenges in designing an interest-based distributed aggregation of users in p2p systems, in: 2009 IEEE ICUMT, IEEE, 2009, pp. 1–8.
[23] A. Aral, T. Ovatman, A decentralized replica placement algorithm for edge computing, IEEE Trans. on Network and Service Management 15 (2018) 516–529.
[24] C. Li, Y. Wang, H. Tang, Y. Zhang, Y. Xin, Y. Luo, Flexible replica placement for enhancing the availability in edge computing environment, Computer Communications 146 (2019) 1–14.
[25] R. Beraldi, A. Mtibaa, H. Alnuweiri, Cooperative load balancing scheme for edge computing resources, in: 2017 Second International Conference on Fog and Mobile Edge Computing (FMEC), IEEE, 2017, pp. 94–100.
[26] C. Mechalikh, H. Takta, F. Moussa, Pureedgesim: A simulation toolkit for performance evaluation of cloud, fog, and pure edge computing environments, in: 2019 Int. Conf. on High Perform. Comput. Simul. (HPCS), 2019, pp. 700–707.