A Location-Allocation model for Fog Computing Infrastructures

Thiago Alves de Queiroz[1], Claudia Canali[2], Manuel Iori[3], Riccardo Lancellotti[2]

[1]*Institute of Mathematics and Technology, Federal University of Goiás, Catalão-Goiás, Brazil*

[2]*Department of Engineering "Enzo Ferrari", University of Modena and Reggio Emilia, Modena, Italy*

[3]*Department of Science and Methods for Engineering, University of Modena and Reggio Emilia, Reggio Emilia, Italy*

taq@ufg.br, claudia.canali@unimore.it, manuel.iori@unimore.it, riccardo.lancellotti@unimore.it

Keywords: Fog computing, Facility location-allocation problem, Optimization model

Abstract: The trend of an ever-increasing number of geographically distributed sensors producing data for a plethora of applications, from environmental monitoring to smart cities and autonomous driving, is shifting the computing paradigm from cloud to fog. The increase in the volume of produced data makes the processing and the aggregation of information at a single remote data center unfeasible or too expensive, while latency-critical applications cannot cope with the high network delays of a remote data center. Fog computing is a preferred solution as latency-sensitive tasks can be moved closer to the sensors. Furthermore, the same fog nodes can perform data aggregation and filtering to reduce the volume of data that is forwarded to the cloud data centers, reducing the risk of network overload. In this paper, we focus on the problem of designing a fog infrastructure considering both the location of how many fog nodes are required, which nodes should be considered (from a list of potential candidates), and how to allocate data flows from sensors to fog nodes and from there to cloud data centers. To this aim, we propose and evaluate a formal model based on a multi-objective optimization problem. We thoroughly test our proposal for a wide range of parameters and exploiting a reference scenario setup taken from a realistic smart city application. We compare the performance of our proposal with other approaches to the problem available in literature, taking into account two objective functions. Our experiments demonstrate that the proposed model is viable for the design of fog infrastructure and can outperform the alternative models, with results that in several cases are close to an ideal solution.

1 Introduction

Fog computing is joining the traditional cloud platforms as the enabling technology for a wide range of applications [13, 16]. Applications that must cope with a large amount of data produced by a wide set of distributed sensors are a typical scenario where fog computing is a winning asset. For example, Internet of Things frameworks, smart city support, and environmental monitoring can benefit from the distributed nature of fog computing. Another class of applications that can take advantage from the fog computing paradigm is that of delay-sensitive tasks, such as the support of autonomous driving.

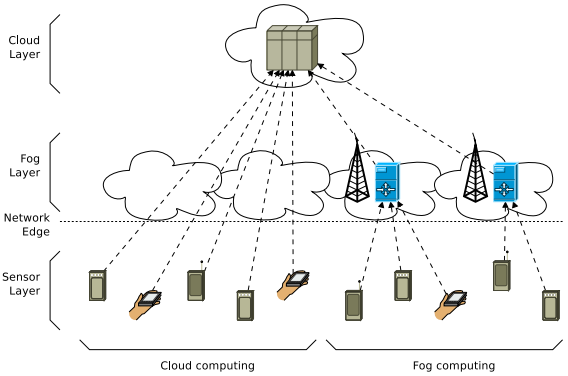


Figure 1: Cloud and fog infrastructures

Figure 1 presents a comparison of fog and cloud infrastructures. In the cloud case (the left part of the figure) a set of sensors (at the bottom of the figure) sends data directly to the cloud data center (at the top of the figure) for processing. In the fog case (on the right part of the figure), a layer of fog nodes is placed close the network edge (and hence to the sensors) to host pre-processing, filtering, and aggregation tasks.

The advantage of fog computing over a traditional cloud computing approach in these scenarios is twofold. First, in a cloud scenario the huge data volume reaching the cloud data center increases the risk of high network utilization and can determine poor performance. Even in the case where the high

network load does not result in a performance degradation, high network utilization is still undesirable due to the non-negligible economic cost related to the cloud pricing model. The distributed nature of fog computing and the ability of fog nodes to reduce the data volume through pre-processing are key features to address this issue. Second, latency-sensitive applications cannot accept a delay that may be in the order of hundreds of milliseconds, due to the potentially high round-trip-time latency with the cloud data center. The fog layer located close to the network edge can guarantee low latency and fast response even for this class of applications.

The additional degree of freedom provided by the introduction of fog nodes opens also new problems for the infrastructure design. In particular, some studies consider a naive approach in the allocation (i.e., mapping) of data flows to fog nodes, assuming that every sensor can reach only the nearest fog node [7, 17]. Recent studies demonstrated that an optimized assignment of sensors to fog nodes can provide a major advantage in solving this problem [2, 3]. However, even when some optimization is performed in the sensor-to-fog mapping, no effort is devoted in understanding whether the whole fog infrastructure is required or some fog node can be switched off to reduce energy consumption. This problem has been widely explored at the level of managing resources in a cloud data center [1, 12], but it has been neglected in the fog computing area.

In this paper, we explicitly address these issues in the area of fog computing (unlike studies such as [4] that focuses on generic distributed stream processing systems). We introduce a performance model for fog computing that considers both network delays and processing time at the level of fog nodes. Furthermore, we propose an optimization model, based on a facility location-allocation problem, aiming to locate fog nodes and allocate sensors to fog nodes. This class of problems have been studied in the area of operational research [5, 6, 8]. In particular, studies concerning the application of such models in urban scenarios [9, 15] and with variable number of nodes [11] have been proposed recently. However, our proposal is characterized by the presence of two objective: minimize the number of used fog nodes while guaranteeing the respect of a service level agreement on response time; and minimize the response time for the given number of selected fog nodes. Furthermore, our model capture the nature of the underlying problem, that is characterized by non-linear functions in the description of the response time. to the best of our knowledge the proposal in this paper is the first attempt to model this dual-objective problem in the area of fog computing.

The experiments are based on a real, georeferenced scenario. We consider the design of a smart-city infrastructure in Modena, Italy, and compare the proposed model with a *simplified* model proposed in [2, 3]. All these comparisons use a linear ideal model as a lower bound for the performance. The results demonstrate that the proposed model is a viable alternative for the design of fog infrastructure and it can outperform the alternative in terms of ensuring adequate performance while minimizing the infrastructure cost. Furthermore, in most cases, the performance of our solution are close to the ideal solution, especially in the most critical cases when the system load is high.

The remainder of the paper is organized as follows. Section 2 presents the theoretical modeling for the considered problem. Section 3 presents the experimental setup and the considered scenarios, and provides a thorough evaluation of the proposed model against the alternatives. Finally, Section 4 presents some concluding remarks and outlines some future work direction.

2 Problem definition

In the proposed model, we assume a stationary scenario where a set $\mathcal{S}$ of similar sensors are distributed over an area. Sensors produce data at a steady rate, with a frequency that we denote as λ_i for the generic sensor i. The fog layer is composed by a set $\mathcal{F}$ of nodes that receive data from the sensors and perform operations on such data. Examples of these operations include filtering and/or aggregation, or some form of analysis to identify anomalies or problems as fast as possible. The rate at which the fog node j processes data is μ_j (hence $1/\mu_j$ is the average processing time for a data unit). We also consider that each fog node j is characterized by a fixed cost c_j if the node is turned on (i.e., a fog is located at position j). We consider a set of cloud data centers $\mathcal{C}$ that collect data from the fog nodes. The model considers also the presence of network delays from sensors to fog nodes and from fog nodes to cloud data centers. In particular, we define as δ_{ij} the delay from sensor i to fog node j, while δ_{jk} is the delay from fog node j to cloud data center k.

For the model, we use three families of binary decision variables. Two families are used to allocate sensors to fog nodes and fog nodes to cloud data center, that is, x_{ij} and y_{jk} model if sensor i sends data to fog node j and if fog node j sends data to cloud data center k, respectively. The last family is E_j and it de-

fines if a fog node is located at position j, that is, if such fog node is turned on and can be used to process data from sensors.

We summarize the main symbols used throughout the model in Table 1.

2.1 Sensor allocation problem

The problem of sensor mapping (i.e., allocation) relies on the definition of the performance metrics that are considered in the optimization problem. The sensor mapping problem was introduced in [2, 3]. In this subsection, we present a revised version of the model. As a minimal metric for the model we focus on the average response time, defined in Eq. (1), that is composed of three components: the network delay due to the sensor to fog latency in Eq. (2), the network delay due to the fog to cloud latency in Eq. (3), and the processing time on the fog nodes in Eq. (4).

$$T_R = T_{netSF} + T_{netFC} + T_{proc} \tag{1}$$

$$T_{netSF} = \frac{1}{\sum_{i\in\mathcal{S}} \lambda_i} \sum_{i\in\mathcal{S}} \sum_{j\in\mathcal{F}} \lambda_i x_{ij} \delta_{ij} \tag{2}$$

$$T_{netFC} = \frac{1}{\sum_{j\in\mathcal{F}} \lambda_j} \sum_{j\in\mathcal{F}} \sum_{k\in\mathcal{C}} \lambda_j y_{jk} \delta_{jk} \tag{3}$$

$$T_{proc} = \frac{1}{\sum_{j\in\mathcal{F}} \lambda_j} \sum_{j\in\mathcal{F}} \lambda_j \frac{1}{\mu_j - \lambda_j} \tag{4}$$

It is worth mentioning that in two delay components, T_{netSF} in (2) and T_{netSF} in (3), the average delay of each sensor and fog node is weighted by the amount of traffic experiencing that delay, which is λ_i for T_{netSF} and λ_j in T_{netFC}. The incoming data rate on each fog node λ_j can be defined as the sum of the data rates of the sensors allocated to that node:

$$\lambda_j = \sum_{i\in\mathcal{S}} x_{ij}\lambda_i, \quad \forall j \in \mathcal{F} \tag{5}$$

The processing time T_{proc} can be modeled using the queuing theory. An estimation of this component of the response time, consistent with other results in literature [1, 2, 3] is used in Eq. (4).

The mathematical model for the sensor allocation problem uses the definition of T_R in (1) as the objective function. As we are not taking into account the problem of locating fog nodes, in this part of the problem definition, we do not consider the decision variable E_j, so we use just $x_{i,j}$ and $y_{j,k}$. Then, we consider a set of constraints defined as follows:

$$\lambda_j < \mu_j, \quad \forall j \in \mathcal{F} \tag{6}$$

$$\sum_{j\in\mathcal{F}} x_{ij} = 1, \quad \forall i \in \mathcal{S} \tag{7}$$

$$\sum_{k\in\mathcal{C}} y_{jk} = 1, \quad \forall j \in \mathcal{F} \tag{8}$$

In particular, constraints (6) ensure that no overload occurs on each fog node, that is the incoming data flow must not exceed the processing rate. Constraints (7) guarantee for each sensor that exactly one fog node processes its data, while constraints (8) ensure for each fog node that exactly one cloud data center receives its processed data.

2.2 Fog nodes location problem

We introduce an additional problem to the sensor allocation problem, which is the location of a subset of fog nodes. For that, we add an additional variable E_j for each location j where a fog node is powered on. For a node that is powered down, no processing must occur. This means that constraints (6) must be re-defined with an additional constraint, such that when E_j is equal to zero, we have λ_j also equal to zero, resulting in the new constrains:

$$\lambda_j \quad < \quad E_j\mu_j \quad \forall j \in \mathcal{F} \tag{9}$$

The optimization problem considers two criteria:

- Minimize the cost associated with the number of fog nodes turned on. Recalling the cost c_j associated to using a fog node in location j, this objective is:

$$C = \sum_{j\in\mathcal{F}} c_j E_j \tag{10}$$

- Minimize the delay in sensor to fog to cloud transit of data. To this aim we can use the cost function introduced as T_R in (1).

It follows that the overall model for the fog node

Table 1: Notation and parameters for the proposed model.

Model parameters	
$\mathcal{S}$	Set of sensors
$\mathcal{F}$	Set of fog nodes
$\mathcal{C}$	Set of cloud data centers
λ_i	Outgoing data rate from sensor i
λ_j	Incoming data rate at fog node j
$1/\mu_j$	Processing time at fog node j
δ_{ij}	Communication latency between sensor i and fog j
δ_{jk}	Communication latency between fog j and cloud k
c_j	Cost for locating a fog node at position j (or for keeping the fog node turned on)
Model indices	
i	Index for a sensor
j	Index for a fog node
k	Index for a cloud data center
Decision variables	
E_j	Location of fog node j
x_{ij}	Allocation of sensor i to fog j
y_{jk}	Allocation of fog node j to cloud k

location-allocation problem is defined as:

Minimize:

$$C = \sum_{j \in \mathcal{F}} c_j E_j \tag{11}$$

$$T_R = T_{netSF} + T_{netFC} + T_{proc} \tag{12}$$

Subject to:

$$T_R \leq T_{SLA} \tag{13}$$

$$\lambda_j < E_j \mu_j, \quad \forall j \in \mathcal{F} \tag{14}$$

$$\sum_{j \in \mathcal{F}} x_{ij} = 1, \quad \forall i \in \mathcal{S}, \tag{15}$$

$$\sum_{k \in \mathcal{C}} y_{jk} = E_j, \quad \forall j \in \mathcal{F} \tag{16}$$

$$E_j \in \{0,1\}, \quad \forall j \in \mathcal{F} \tag{17}$$

$$x_{ij} \in \{0,1\}, \quad \forall i \in \mathcal{S}, j \in \mathcal{F} \tag{18}$$

$$y_{jk} \in \{0,1\}, \quad \forall j \in \mathcal{F}, k \in \mathcal{C} \tag{19}$$

The two objective functions (11) and (12) are related to the minimization of costs and latency in the network. Constraints (14) represent the no-overload condition. Constraints (15) and (16) are the revised version of the constraints (7) and (8) introduced in Section 2.1 where we now consider the variable E_j. Constraints (17), (18) and (19) describe the domain of the decision variables.

One important set of constraints to discuss in this formulation is (13), which introduce a limit such that the average response time does not exceed a *Service Level Agreement* (SLA). The maximum response time is typically defined as a multiple of the average response time $1/\mu$ [1]. Besides that, we introduce an additional term due to the network delays in a distributed architecture (that we consider non-negligible) related to the sensor to fog and fog to cloud network delays. In particular, to this aim we consider this network delay contribution depending on the average network delays that we define as δ. We formalize the value of the SLA limit in (20), where K is a constant defined in accordance with the network requirements.

$$T_{SLA} = \frac{K}{\mu} + 2\delta \tag{20}$$

3 Experimental results

The performance of the proposed model is assessed through a realistic fog computing scenario, where geographically distributed sensors send data to fog nodes. Throughout this section, we start by describing the experimental setup used in the performance evaluation and then we compare the performance of the considered alternatives.

3.1 Experimental setup

The scenario is based on a smart city project for the city of Modena in Italy, which has a population of

around 180.000 inhabitants aiming to correlate air quality and car traffic as in [14]. The application considers a set of 89 sensors located in the main streets of the city. These sensors are wireless devices that collect information related to car and pedestrian traffic (that comprise reading from proximity sensors and, possibly, low-resolution images) and send these data to fog nodes. For the sake of the model, the location of the sensors is obtained by geo-referencing the selected streets. The fog nodes pre-process the received data by filtering the proximity sensor readings and, if available, analyze images from the camera to detect cars and pedestrians. The pre-processed data are then sent to a cloud data center located on the municipality premises. For the fog nodes locations, we select a set of six government buildings, while the location of the municipality cloud data center is known. To summarize, the scenario is composed of 89 sensors, 6 fog nodes and 1 data center.

We assume to have sensors with long-range wireless connectivity, such as LoRA WAN[1] or IEEE 802.11ah/802.11af [10]. Hence, every sensor can connect with every fog node. Due to the growing delay and decreasing bandwidth limitations as the distance from a sensor to the fog node increases, we assume that the network delay depends on the physical distance between two nodes as in [2, 3].

Regarding the parameters for the models, even if the model itself support the description of a highly heterogeneous architecture, we focus on an homogeneous scenario. Hence, the cost c_j of locating a fog node at position j is equal to 1, for all $j \in \mathcal{F}$. This means that, from an operating cost point of view, the fog nodes are similar and the objective function will strive to reduce the overall number of nodes used. In the performance evaluation, we describe each scenario using the following main parameters:

- λ is the data rate of each sensor;
- ρ is the average utilization of the system, defined as $\frac{\sum_{i \in \mathcal{S}} \lambda_i}{\sum_{j \in \mathcal{F}} \mu_j}$;
- $\delta\mu$ is the ratio between the average network delay δ and the average service time of a request that we denote as $1/\mu$. This parameter determine the CPU-bound or network-bound nature of a scenario.

Based on preliminary evaluation of the smart city sensing application for traffic monitoring used in the experiments, we consider that each sensor can provide a reading every 10 seconds. Hence the data rate $\lambda_i = 0.1, \forall i \in \mathcal{S}$. For the parameter ρ, we consider a wide range of values, namely $\rho \in \{0.1, 0.2, 0.5, 0.8, 0.9\}$. For each value of ρ, considering sensors and fog nodes homogeneous and knowing the value of λ_i, we derive the value of $\mu_j = \mu$, which is assumed the same for each $j \in \mathcal{F}$.

We consider values for the parameter $\delta\mu$ ranging multiple orders of magnitude, that is $\delta\mu \in \{0.01, 0.1, 1, 10\}$. This parameter allows us to explore scenarios that can be CPU-bound (e.g., when $\delta\mu = 0.01$) where computing time is much higher than transmission time up to cases that are network-bound (e.g., when $\delta\mu = 10$). We derive the average network delay from the $\delta\mu$ parameter and the previously computed parameter μ_j. It is worth mentioning that, even if in our analysis we may end up with very high network delays, these scenarios can be still considered realistic if we consider that the network contribution may involve the transfer of images over low-bandwidth links. In the definition of the SLA in Eq. (20), the constant K is set to 10, which is a common value in the literature [1].

The evaluation of the proposed model considers a wide range of different scenarios related to the previously introduced parameters. Each scenario is named according to a format ins-ρ-$\delta\mu$ (e.g., the instance ins-0.1-0.01 indicates that the scenario has $\rho = 0.1$ and $\delta\mu = 0.01$). In the experiments, we compare the following models:

- Simplified model (SM): is the simplified version of the problem described in Section 2.1 and presented for the first time in [2] in which all fog nodes are assumed on, that is $E_j = 1, \forall j \in \mathcal{F}$. Although this could represent a situation where the energy consumption may be high, the infrastructure provides better performance from a response time point of view (for the objective function (12));
- Proposed model (PR): is the model introduced in this study and described in Section 2.2;
- Continuous model (CN): consists of the proposed model in which all variables (i.e., E_j, x_{ij}, and y_{jk}) are assumed continuous, ranging in the interval $[0, 1]$. The result of this model is clearly an infeasible solution, but can be used as a lower bound for all the other models.

The main metric used in the comparison is the cost related to the number of fog nodes located (i.e., turned on in the network), which corresponds to (11). The second metric is related to the actual average response time and corresponds to (12). As a baseline for the performance evaluation, we compare each alternative model with the continuous model. Throughout the performance analysis, we evaluate the performance with respect to the continuous model using a devia-

[1] https://lora-alliance.org/

tion measure for each objective function Obj_1 in (11) and Obj_2 in (12). The deviation function is defined as:

$$\epsilon(Obj_1^M) = \frac{Obj_1^M - Obj_1^{CN}}{Obj_1^{CN}} \quad (21)$$

$$\epsilon(Obj_2^M) = \frac{Obj_2^M - Obj_2^{CN}}{Obj_2^{CN}} \quad (22)$$

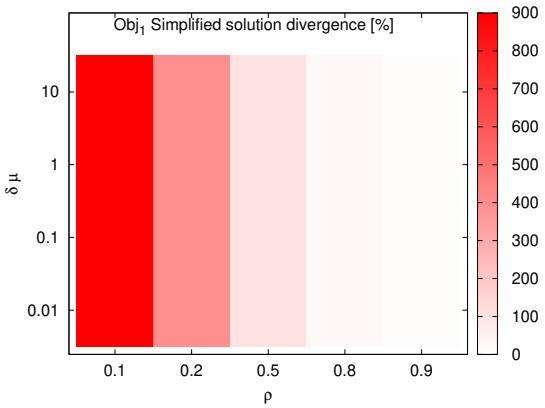
where Obj_1^M and Obj_2^M are the values of the objective functions for the model $M \in \{SM, PR\}$, and Obj_1^{CN} and Obj_2^{CN} are the values of the objective functions for the continuous model (CN). For the numeric results of the models we rely on LocalSolver[2] version 9.0, with a time limit of 300 seconds (5 minutes) as stopping criterion. LocalSolver is a general mathematical programming solver that hybridizes local and direct search, constraint propagation and inference, linear and mixed-integer programming, and nonlinear programming methods. It can handle multi-objective problems, where the objectives are optimized in the order of their declaration in the model.

3.2 Performance evaluation

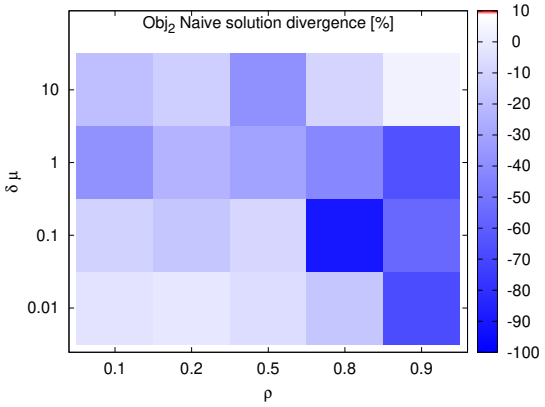
To provide a complete evaluation of the models, we present the numerical values of the solutions (together with the number of iterations required by LocalSolver to reach the value) in Table 2. Moreover, we focus the analysis on the deviation metric previously introduced to compare the pros and cons of each considered model.

Figure 2 shows the deviation as heat maps for the two objective functions of the simplified model. We observe for this model that all scenarios have a feasible solution, confirming the benefit from introducing a mathematical model in the data flow mapping (preliminary experiments carried out with the model used in [7] where the sensors-to-fog mapping is based only on the geographic distance were unable to guarantee feasible solution for high values of ρ). We observe that, for the first objective function, the deviation is driven just by parameter ρ, which determines the number of fog nodes used by the continuous model. Indeed, the simplified model uses all the available fog nodes, while, especially when ρ is low, the processing of sensors data may require just a fraction of the infrastructure computational power, thus motivating the high value of the deviation.

Focusing, instead, on the second objective function in Figure 2b, we observe that a higher number of fog nodes can provide a major reduction in the response time, as testified by the large presence of



(a) $\epsilon(Obj_1^{SM})$



(b) $\epsilon(Obj_2^{SM})$

Figure 2: Performance of the Simplified model.

blue hues in the figure. A deviation close to -100% means that the simplified model halves the average response time. Indeed, in this case we have an abundance of computational power due to the use of all the fog nodes, while the continuous model uses just the minimum amount of resources to satisfy the SLA constraint.

The performance of the proposed model are shown in Figure 3. Unlike the previously considered model, this approach explicitly aims to reduce the number of fog nodes used. The impact of this choice is evident in Figure 3a, where we observe a non-purely vertical pattern in the deviation. Furthermore, we observe that the deviation is, in most of the cases, much lower compared to the other model – indeed the number of fog nodes is is the ceiling value compared to the value of the CN model. There are a few noteworthy exceptions, mainly in the case where the parameter $\delta\mu$ is high. Under these circumstances, the SLA constraint (13) requires more fog nodes because using a network link with above than average delay is likely to result in a SLA violation. Due to the Boolean nature of the variable E_j, a fog node is either

[2]http://www.localsolver.com

Table 2: Results of the model and alternatives.

	Continuous		Simplified			Proposed		
Instance	Obj-1	Obj-2	Iter.	Obj-1	Obj-2	Iter.	Obj-1	Obj-2
ins-0.1-0.01	0.6	0.08	65083	6	0.08	702	1	0.24
ins-0.1-0.1	0.6	0.13	75637	6	0.12	742	1	0.85
ins-0.1-1	0.6	0.80	93112	6	0.49	721	1	6.97
ins-0.1-10	0.6	5.23	99601	6	4.25	46757	6	4.25
ins-0.2-0.01	1.2	0.18	71518	6	0.18	80532	2	0.38
ins-0.2-0.1	1.2	0.32	60473	6	0.27	73102	2	0.74
ins-0.2-1	1.2	1.34	69463	6	1.02	72194	2	4.15
ins-0.2-10	1.2	9.69	96283	6	8.54	37529	6	8.54
ins-0.5-0.01	3.0	0.75	110875	6	0.71	32670	4	1.40
ins-0.5-0.1	3.0	1.09	70181	6	1.00	25875	4	1.86
ins-0.5-1	3.0	4.60	41097	6	3.16	17130	4	4.42
ins-0.5-10	3.0	36.43	58968	6	22.24	21714	6	22.24
ins-0.8-0.01	4.8	3.29	108225	6	2.78	40842	5	16.23
ins-0.8-0.1	4.8	33.39	112087	6	3.30	30397	5	16.74
ins-0.8-1	4.8	14.40	90756	6	8.31	32277	5	21.80
ins-0.8-10	4.8	56.62	95538	6	51.13	26977	6	51.13
ins-0.9-0.01	5.4	19.79	97888	6	6.39	22285	6	6.39
ins-0.9-0.1	5.4	15.68	123337	6	6.97	26125	6	6.97
ins-0.9-1	5.4	37.00	121558	6	12.82	25999	6	12.82
ins-0.9-10	5.4	69.00	45547	6	71.05	37716	6	71.05

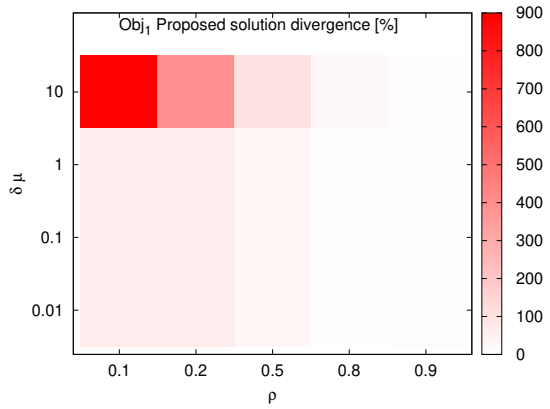
used or not, so the objective function Obj_1^{PR} reaches a high value even when ρ is low. The CN model instead allows the use of just a small fraction of every fog node. This means that the model can use all the available fog nodes turning them on just for the fraction of their capacity required to satisfy the incoming load.

Considering the impact of the second objective function in Figure 3b, we observe that the proposed model is typically able to achieve performance comparable with the continuous model. In some cases the proposed model outperforms even the continuous model. To better understand the reasons for this variable performance, we focus on two extreme cases. For $\rho = 0.1$ and $\delta\mu = 1$, the proposed model provides very poor performance compared to the continuous alternative. To understand this high response time we must consider that both the proposed and the continuous models use the same number of fog nodes (just one). Furthermore, we must factor in the significant impact of network delays (that are not negligible compared to the service time). In this scenario, not being able to use a fraction of the computational power of every fog node results in a high impact of the network delays because we need to make every communication converge on just one fog node. An opposite case is when we have a high load and low contribution of network delays (e.g., $\rho = 0.9$ and $\delta\mu = 0.01$). In this case, having more fog nodes powered on than what is strictly necessary (because $E_j = 1$ for all fog nodes) results in a lower processing time. At the same time, the low impact of network delays makes the problem of achieving a good load balancing quite straightforward because the penalty for reaching a fog node far away is almost negligible.

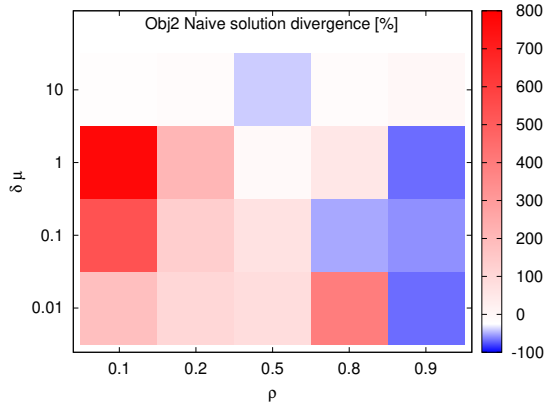
4 Conclusions

In this paper, we focused on a facility location-allocation problem related to the management of a fog infrastructure, with special attention to the mapping of data flows from the sensors to the fog nodes and from the fog nodes to the cloud data centers. Then, we propose a mathematical model that starts with a list of *potential* fog nodes and selects a minimal subset of them to guarantee the satisfaction of a Service Level Agreement.

We test the proposed model against alternative models from the scientific literature. The experiments are based on a realistic situation from a project for a smart city application. We consider a wide range of scenarios characterized by different load levels and by different ratios between the service time (that is the processing time for a set of data from a sensor) and network delay. The results demonstrate that the proposed model can outperform existing alternatives in the literature. We also consider an ideal but unreal-



(a) $\epsilon(Obj_1^{PR})$



(b) $\epsilon(Obj_2^{PR})$

Figure 3: Performance of the proposed model.

istic model and demonstrate that the proposed model can, in several cases, achieve a result that is comparable with this ideal solution.

This paper is a step in a wider research line on fog infrastructure design. We plan to extend our proposal including quickly and effectively heuristic algorithms that can be used to solve our problem, and to introduce dynamic scenarios where the load can change through time.

REFERENCES

[1] D. Ardagna, M. Ciavotta, R. Lancellotti, and M. Guerriero. A hierarchical receding horizon algorithm for QoS-driven control of multi-IaaS applications. *IEEE Transactions on Cloud Computing*, pages 1–1, 2018.

[2] C. Canali and R. Lancellotti. A Fog Computing Service Placement for Smart Cities based on Genetic Algorithms. In *Proc. of International Conference on Cloud Computing and Services Science (CLOSER 2019)*, Heraklion, Greece, May 2019.

[3] C. Canali and R. Lancellotti. GASP: Genetic Algorithms for Service Placement in fog computing systems. *Algorithms*, 12(10), 2019.

[4] V. Cardellini, F. Lo presti, M. Nardelli, and G. Russo russo. Optimal operator deployment and replication for elastic distributed data stream processing. *Concurrency Computation*, 30(June):1–20, 2017.

[5] D. Celik Turkoglu and M. Erol Genevois. A comparative survey of service facility location problems. *Annals of Operations Research*, in press, 2019.

[6] L. Cooper. Location-allocation problems. *Operations Research*, 11(3):331–343, 1963.

[7] R. Deng, R. Lu, C. Lai, T. H. Luan, and H. Liang. Optimal Workload Allocation in Fog-Cloud Computing Toward Balanced Delay and Power Consumption. *IEEE Internet of Things Journal*, 3(6):1171–1181, Dec 2016.

[8] H. A. Eiselt and G. Laporte. Objectives in location problems. In Z. Drezner, editor, *Facility Location: A survey of application and methods*, pages 151–180. Springer, 1995.

[9] R. Z. Farahani, S. Fallah, R. Ruiz, S. Hosseini, and N. Asgari. Or models in urban service facility location: A critical review of applications and future developments. *European Journal of Operational Research*, 276(1):1 – 27, 2019.

[10] E. Khorov, A. Lyakhov, A. Krotov, and A. Guschin. A survey on IEEE 802.11 ah: An enabling networking technology for smart cities. *Computer Communications*, 58:53–69, 2015.

[11] R. Kramer, J.-F. Cordeau, and M. Iori. Rich vehicle routing with auxiliary depots and anticipated deliveries: An application to pharmaceutical distribution. *Transportation Research Part E: Logistics and Transportation Review*, 129:162 – 174, 2019.

[12] A. Marotta and S. Avallone. A Simulated Annealing Based Approach for Power Efficient Virtual Machines Consolidation. In *Proc. of 8th International Conference on Cloud Computing (CLOUD)*. IEEE, 2015.

[13] OpenFog Consortium Architecture Working Group. OpenFog Reference Architecture for Fog Computing. Technical report, OpenFog consortium, Feb. 2017.

[14] L. Po, F. Rollo, J. R. Viqueira, R. Trillo, J. C. Lopez, A. Bigi, M. Paolucci, and P. Nesi. Trafair: Understanding traffic flow to improve air quality. In *Proc. of IEEE International Smart Cities Conference (ISC2 2019)*, Casablanca, Morocco, Oct. 2019.

[15] R. A. C. Silva and N. L. S. Fonseca. On the location of fog nodes in fog-cloud infrastructures. *Sensors*, 19(11), 2019.

[16] S. Yi, C. Li, and Q. Li. A survey of fog computing: Concepts, applications and issues. In *Proceedings of the 2015 Workshop on Mobile Big Data*, Mobidata '15, pages 37–42, New York, NY, USA, 2015. ACM.

[17] A. Yousefpour, G. Ishigaki, and J. P. Jue. Fog computing: Towards minimizing delay in the internet of things. In *2017 IEEE International Conference on Edge Computing (EDGE)*, pages 17–24, June 2017.