Profit-aware coalition formation in fog computing providers: A game-theoretic approach

Received: Added at production | Revised: Added at production | Accepted: Added at production

DOI: xxx/xxxx

RESEARCH ARTICLE

Profit-aware coalition formation in fog computing providers: A game-theoretic approach

Cosimo Anglano[1] | Massimo Canonico[1] | Paolo Castagno[2] | Marco Guazzone*[1] | Matteo Sereno[2]

[1]Computer Science Institute, DiSIT, University of Piemonte Orientale, Italy
[2]Department of Computer Science, University of Torino, Italy

Correspondence
*Marco Guazzone. Viale T. Michel 11, 15121 Alessandria (Italy). Email: marco.guazzone@uniupo.it

Summary

We consider fog computing scenarios where data generated by a set of IoT applications need to be processed locally by a set of fog nodes, belonging to distinct *Fog Infrastructure Providers* (FIPs) sharing the same co-location facility, with the aim of meeting QoS goals despite time-varying workloads.

We argue that these FIPs may find it profitable to cooperate, by mutually sharing their workload and resources, and we show – by using a game-theoretical framework – that this is indeed the case when *stable* coalitions can be formed. Moreover, we present a mathematical model for maximizing the profit obtained for allocating IoT applications to a group of FIPs, that allows to lower costs and, at the same time, to comply with the associated QoS parameters.

Based on these results, we devise a coalition formation algorithm that allows each FIP to decide with whom to cooperate so as to increment its profits. The efficacy of the devised algorithm is assessed by means of an experimental evaluation taking into account different workload intensities. The results from these experiments show the capability of the proposed algorithm to form coalitions of FIPs that are profitable and stable in all the scenarios we take into consideration.

KEYWORDS:
Fog computing, Fog federation, Game Theory, Coalition Formation, Profit Maximization.

1 | INTRODUCTION

Large-scale *Internet of Things* services, such as healthcare [1], smart cities [2], agriculture monitoring [3], and many others [4], are nowadays pervasive in cyber-physical systems. These services are based on the collection of very large volumes of data, produced by an extremely large number of end devices (sensors, smart personal devices, vehicles, etc.), located at the edge of the network, that operate on a 24/7 basis every day of the year and that often need (near) real-time processing [5].

In recent years, *Fog Computing* [6] has emerged as a suitable solution to the processing needs of IoT data [7,8,9]. In a typical Fog Computing architecture, a group of *fog nodes* – that are placed close-by to IoT devices – provide enough computing, storage, and communication resources to enable local processing of IoT data, thus avoiding the high-latency transfers to a cloud system, and reducing congestion at the core network.

Latency reduction can be quite expensive in terms of resources, given that (as argued in [10]), the latency of communications among the devices placed in a given area is proportional to the inverse square root of the number of these devices (i.e., to reduce latency by half in a given area hosting X fog nodes, $4 * X$ fog nodes must be placed in that area). This implies that the cost incurred by enterprises that need purchasing, operating, and managing a suitable number of fog nodes to cover large geographic areas might be excessively high, with the result that they could not exploit the benefits brought by the Fog Computing paradigm.

A solution to this problem has been envisioned in [10], where the emergence of *Fog Infrastructure Providers* (FIPs) has been anticipated. Specifically, a FIP places their resources at the network edge, and rent them on a pay-as-you-go basis to those who require to process their data locally. The emergence of FIPs would enable individual enterprises to eliminate the capital and operational expenses implied by the use of an own fog computing infrastructure, as they would not have to purchase, deploy, and manage their own fog resources. Operators would be instead encouraged to enter the FIP business thanks to the fact that for them it would be easy to amortize these costs, as current virtualization technologies enable them to safely and effectively multiplex the same physical infrastructure among multiple tenants, since each tenant typically needs to use only a fraction of the physical capacity of fog nodes, as consequence of the variability of the volume of generated data [5].

Moreover, as argued in [10], many of these FIPs will be regional operators, providing service in small urban centers or rural areas, that will exploit co-location facilities to reduce their operational costs, in line with current trends for small-to-medium-size enterprises [11]. In these scenarios, the profits of FIPs can be increased if they agree to cooperate among them by mutually sharing their users and their infrastructures. As a matter of fact, the co-located resources of a set of FIPs are indistinguishable from the perspective of latency perceived by a user regardless of the FIPs (s)he is renting resources from, so a FIP may host users of another FIP, or can offload its users to another FIP, without incurring into performance penalties. Consequently, a FIP can increase its net profit by either (a) cutting down its energy consumption costs by turning off some of its fog nodes and offloading its users to fog nodes of other (cooperating) FIPs, or (b) improving its earnings by running workloads generated by users of other FIPs, or (c) exploiting the capacity of fog nodes belonging to other FIPs to serve more users than it what could do by working alone.

In order to motivate FIPs to cooperate among them, suitable benefits must result from cooperation while, at the same time, the risks of monetary losses (due to the inability to comply with the QoS parameters agreed with its customers) are maintained below a suitable threshold.

In this paper we cope with the problem of studying under what conditions a set of FIPs finds profitable to cooperate among them, and of providing a suitable algorithm – based on this study – that allows every FIP to decide whether it is profitable or not to cooperate with other FIPs in the same areas.

We tackle these problems by exploiting game-theoretic techniques, where the establishment of the cooperation among a set of FIPs is modeled as a *cooperative game with transferable utility* [12] (specifically, as a *hedonic game* [13], whereby every FIP bases its decision on its own preferences), and we devise an algorithm that allows a group of FIPs to decide whether to cooperate with other FIPs, and if so with whom to cooperate.

In particular, we propose a game-theoretic framework, that extends our previous work [14], to study the problem of forming *stable* coalitions among FIPs, and a mathematical optimization model to allocate IoT applications to a group of resources, in order to increase profits and, at the same time, to meet application QoS for FIPs inside the same coalition. As a result, we devise a decision algorithm that, taking in input the capacity of fog resources of each FIP, the QoS levels negotiated with its clients, the intensity of the workload submitted to these resources, and the revenues and costs of running its own workloads and the workloads of other FIPs on its own these resources, allows it to assess the profitability of joining or not a coalition formed by other FIPs.

More specifically, we devise a *coalition formation algorithm*, that is able to form stable coalitions by letting each FIP to autonomously and selfishly decide whether to leave the current coalition to join a different one or not on the basis of the net profit it receives for doing so. This algorithm is based on the principle that each FIP pays for the energy consumed to serve each application, whether it belongs to it or to another FIP, but receives a payoff (computed as discussed later) for doing so.

The effectiveness of the coalition formation algorithm we propose is assessed by means of an extensive experimental evaluation, in which we consider a set of realistic operational scenarios. The results of our evaluation show that our algorithm enables a population of FIPs to significantly improve their profits thanks to a suitable combination of energy reduction and satisfaction of QoS requirements.

The contributions of this paper can be summarized as follows:

- we study the problem of improving the profit of a set of FIPs that share the same co-location facility;
- we model the problem as a cooperative game with transferable utility;
- we devise a distributed algorithm enabling a set of FIPs to autonomously decide whether to form a stable coalition or not on the basis of the profits they make by doing so;
- we show its effectiveness through experimental analysis in realistic scenarios.

The remainder of the paper is so organized. In Section 2, we introduce the system model. Section 3 presents the problem statement. In Section 4, we describe the coalition formation process we propose in this paper. In Section 5, we evaluate our proposed algorithm via simulation and discuss achieved results. In Section 6, we discuss the state-of-the-art. In Section 7, we conclude the paper and highlight some future works.

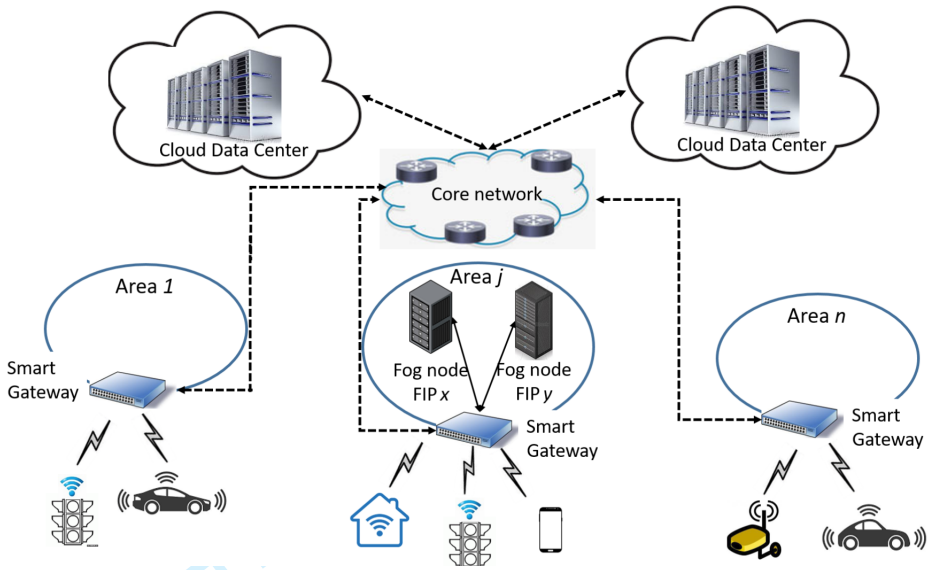

FIGURE 1 System architecture.

2 | SYSTEM MODEL

In this paper, we consider systems, whose architecture is schematically shown in Figure 1, where both users and fog nodes are distributed in a group of n different geographic areas. Within each area, a (possibly large) set of IoT devices, including both stationary (e.g., smart homes, IP cameras, and smart traffic lights) and mobile (e.g., tablets, smartphones, and connected vehicles) devices, generate a stream of data that need to be processed in (near) real-time.

Each area is covered by a single co-location facility that hosts a group of fog nodes running the applications that are responsible for the processing of the data generated by devices located in that area.

We suppose that the communication latency among fog nodes located in the same co-location facility is negligible, while inter-area communication experiences a significantly higher latency, since it requires the traversal of the core network.

The remainder of the section discusses the characteristics of the key system components, namely the fog nodes (Section 2.1), the applications (Section 2.2), and the VMs (Section 2.3).

2.1 | The Fog Nodes

Fog nodes are resource-rich small-scale computing systems [15] (e.g., *Cloudlets* or *Micro Data Centers*) that are located into the co-location facilities of the various areas. In particular, we assume that a population of N FIPs co-exists within each given co-location facility, and that each one of them provides some of the fog nodes located in that facility. For example, the co-location facility of geographic area j in Figure 1 hosts fog nodes that belong to FIPs x and y.

Fog nodes rely on a virtualization platform to run the *Virtual Machines* (VMs) where the various applications are encapsulated together with their software stack (see below). We assume that this platform provides suitable dynamic resource allocation mechanisms (see for instance [16,17,18,19,20]) able to partition the physical capacity of the fog node across the VMs it runs.

We assume that each fog node z is characterized in terms of its CPU capacity c_z, measured through a suitable benchmark (e.g., GeekBench [21]), and of its power consumption $w_z(u)$, which is modeled as in [22] by the following equation:

$$w_z(u) = W_z^{\min} + u \cdot (W_z^{\max} - W_z^{\min}) \quad (1)$$

where $u \in [0, 1]$ is the CPU utilization of the fog node, and $W_z^{\min}$ and $W_z^{\max}$ are its power consumption (in Watts) when its CPU is in the idle state and when it is fully utilized, respectively. [1]

We suppose that, for any fog nodes a and b belonging to the same FIP and located in the same co-location facility, $c_a = c_b$ and $w_a(u) = w_b(u)$ for any value of u (in other words, a and b are identical).

[1] This model, in spite of its simplicity, has been demonstrated to provide accurate estimates of power consumption for different types of systems when running several benchmarks representative of real-world applications [22].

2.2 | The Applications

As discussed above, the data generated in the system are processed by a group of M applications $\mathcal{S} = \{S_1, S_2, \ldots, S_M\}$, each one encapsulated into a group of VMs. To cover all the various geographic areas of its interest, application S_i is present with one or more identical *instances* in the different areas, each one corresponding to a VM; the instances located in geographic area j are responsible of processing the data generated by the devices in that area.

We suppose that the load of data processing requests sent to application S_i in any geographic area j changes with time and is described by its *load profile curve* $\ell_{i,j}(t)$ expressing, as function of time t, the rate at which requests are submitted (e.g., see [23,24,14]). Also, we suppose that, for each application S_i and area j, $\ell_{i,j}(t)$ is known in advance. The load profile curve can be indeed accurately built by estimating both the request rates generated by stationary devices and mobile users, and by aggregating them into a single measure.

Each application S_i is associated with its *reference FIP* $Ref(S_i)$ (i.e., the FIP that runs the VMs encapsulating the instances of S_i). The same FIP k can be the reference FIP of many distinct applications, that we denote as $App(k)$.

We suppose that each application S_i is characterized by its QoS objective, quantified by the maximum value Q_i that the average request processing time D_i can take (in other words, it must be ensured that $D_i \leq Q_i$). To meet its QoS goal, the owner of application S_i makes an agreement with $Ref(S_i)$ stating that (a) $Ref(S_i)$ will charge a given amount of money for each instance of S_i it runs, computed according to an agreed-upon *revenue rate* $R_{Ref(S_i),i}$, per each unit of time, and (b) $Ref(S_i)$ will remunerate the owner of S_i with an amount of money per each unit of time during which the QoS goal of S_i is not achieved, computed according to the agreed-upon *penalty rate* $L_{Ref(S_i),i}$.

2.3 | The Virtual Machines

The instances of any application S_i are encapsulated into a group of identical VMs, each of which hosting a single instance, that are instantiated from a common *template* VM, denoted as VM_i.

VM_i is characterized by the rate μ_i at which requests are processed in the unit of time. Without loss of generality, we suppose that μ_i depends only on the quantity of physical CPU capacity U_i allotted to that VM, [2] and that U_i is set to a constant value and is the same for all the instances of VM_i.

To make sure that μ_i is the same for all the instances of VM_i, we suppose that every of them receives, on the fog node k on which it runs, a suitable amount of CPU capacity $U_{k,i}$ computed as discussed below.

First, a profiling experiment is carried out by running VM_i on a reference fog node x (e.g., using the methodology described in [18,19,20]) in which $U_{x,i}$ is progressively increased until its measured request processing rate reaches the value μ_i, and the corresponding value $U^*_{x,i}$ of allocated physical CPU capacity is recorded.

Then, the amount of physical capacity $U_{k,i}$ that must be allocated to VM_i on fog node $k \neq x$ is computed as [25]:

$$U_{k,i} = U^*_{x,i} \frac{c_x}{c_k} \tag{2}$$

where c_k and c_x represent the physical CPU capacity of fog node k and x, respectively, that are measured as described in Section 2.1. Doing so, we make sure that all the instances of VM_i will proceed at the same rate on the corresponding fog nodes, so they all will exhibit the same processing rate μ_i.

For instance, if $U^*_{x,i} = 0.6$, $c_x = 1$ and $c_k = 2$, then $U_{k,i} = 0.6 \cdot 0.5 = 0.3$ (i.e., if the physical CPU capacity doubles, μ_i is obtained by allocating half of the CPU capacity with respect to the reference fog node).

The choice of imposing that μ_i remains constant for all the lifetime of the instances of VM_i implies that, to achieve the QoS goal Q_i of S_i in geographic area j, it is needed to appropriately choose the number $N_{i,j}$ of VMs allocated on fog nodes placed in that area in order to assure that $D_i \leq Q_i$. However, $N_{i,j}$ depends on the value of the load intensity $\lambda_{i,j}(t)$, which is not constant but varies according to the load profile $\ell_{i,j}(t)$, as already discussed in Section 2.2.

To determine $\lambda_{i,j}(t)$, we proceed as follows. First, likewise existing works (e.g., [24,26,14,27]), we discretize $\ell_{i,j}(t)$ by splitting the time axis into uniform disjoint intervals $[t, t + \Delta t)$ of length Δt time units. Then, for any time interval τ, we approximate the value $\lambda_{i,j}(\tau)$ as a constant value set to the peak load in that interval.

The values of $\lambda_{i,j}(\tau)$ are then fed as input into an M/M/c-FCFS queueing model [28], where $c = N_{i,j}(\tau)$ represents the group of identical VMs (instances of VM_i) associated to application S_i allocated in a given area j to process a stream of incoming requests which is fairly distributed among them. The solution of this model yields the minimum number $N_{i,j}(\tau)$ of VMs, instances of VM_i, in time interval τ that satisfies $D_i \leq Q_i$ as follows

[2] The extension to multiple types of physical resources (e.g., RAM and storage) and to multiple classes of VMs, each one with different physical resource requirements, is straightforward (e.g., see [25]).

(for readability purposes, we drop the dependence on τ):

$$D_i = \frac{G}{\mu_i N_{i,j} - \lambda_{i,j}} + \frac{1}{\mu_i}, \tag{3}$$

$$N_{i,j} \geq \frac{G}{Q_i - \frac{1}{\mu_i}} + \frac{\lambda_{i,j}}{\mu_i}. \tag{4}$$

where:

- $\rho = \frac{\lambda_{i,j}}{\mu_i N_{i,j}}$ is the offered load to the queueing station;
- $G = [1 + (1-\rho)(\frac{N_{i,j}!}{(\rho N_{i,j})^{N_{i,j}}}) \sum_{k=0}^{N_{i,j}-1} \frac{(N_{i,j}\rho)^k}{k!}]^{-1}$ is the probability of a request to be enqueued before being served.

In addition, to assure the stability of the system, the following inequality must hold:

$$N_{i,j} > \frac{\lambda_{i,j}}{\mu_i} \tag{5}$$

3 | PROBLEM STATEMENT

An FIP i aims at increasing its *net profit* (i.e., the difference between its revenues and costs) as much as possible, in face of the request for allocating, for every application $S_k \in App(i)$, the corresponding set of VMs on its fog nodes $FN(i,j)$ located in a given geographic area j. In the rest of this paper, we consider, without loss of generality, a single geographic area, as FIPs allocate VMs for instances of applications submitted in a given area independently from those submitted to other areas. The extension to multiple geographic areas is thus straightforward, given that the overall net profit earned by FIP is therefore simply the sum of the net profits it earns in each single area.

The net profit rate $P_{i,j}$ (i.e., the net profit that FIP i makes per unit of time) in a given area j is given by the following difference (to improve readability, we drop the dependency on time interval):

$$P_{i,j} = \sum_{S_k \in App(i)} R_{i,k} n_{k,j} - \left(\sum_{f \in \mathcal{FN}(i,j)} x_f w_f(u_f) E_{i,j} + \sum_{S_k \in App(i)} \mathbf{1}_{[0,N_{k,j})}(n_{k,j}) L_{i,k} \right) \tag{6}$$

where x_f tells whether fog node f is switched on ($x_f = 1$) or off ($x_f = 0$), $n_{k,j} \leq N_{k,j}$ denotes the number of VMs for S_k that are really allocated in area j (see below), $E_{i,j}$ is the electricity price charged to FIP i in area j (expressed as a cost rate per unit of time), u_f is the total CPU capacity of fog node $FN(i,j)$ allocated to the VMs it hosts, and $\mathbf{1}_\Omega(x)$ represents the *indicator function* which has value 1 if $x \in \Omega$ and 0 otherwise.

In Eq. (6), the first term of the difference represents the sum of the revenue rates $R_{i,k}$ that FIP i earns (per unit of time) for hosting $n_{k,j}$ of its VMs where to run instances of each application S_k, whereas the second term of the difference denotes the costs that FIP i must pay (per unit of time) to host the above VMs. In turn, these costs are computed as the sum of (a) the *energy cost rates* resulting from the fraction of CPU capacity of its powered-on fog nodes to the hosted VMs (see Eq. (1)), and (b) the *monetary penalty rates* $L_{i,k}$ that FIP i must pay if the QoS of some application $S_k \in App(i)$ is violated (i.e., if $n_{k,j} < N_{k,j}$).

The maximization of the net profit rate requires solving an optimization problem to find those values of x_i that maximizes $P_{i,j}$, and that must consider the application penalties, the current workload, and the electricity price. This is a challenging task whose solution is discussed in detail in Section 4.2.

Intuitively, when the number of VMs to allocate on a fog node $FN(i,j)$ is so small that results in a negative net profit, FIP i must determine whether not allocating any VM on $FN(i,j)$ (hence paying the penalties for violating the QoS of the associated applications) would result in a higher profit than allocating the VMs anyway (for not paying high QoS penalties). Moreover, when the number of VMs is so large that it needs more than one fog node to allocate them, FIP i must determine whether allocating all of them would be more profitable than allocating only the ones that results in a positive profit (thus paying the monetary penalties for those applications whose QoS is not achieved).

A way to increase the net profit of a FIP is via cooperation, by means of which two or more FIPs in the same geographic area of interest join to form a coalition where they share their workloads and their fog nodes to serve them.

In particular, with cooperation, FIP i can seek to reduce its energy consumption costs by allocating (some of) its VMs to the fog nodes of other FIPs, so that its fog nodes can be switched off. Furthermore, FIP i can seek to increase its revenues either by hosting VMs from other FIPs (so as to amortize its energy consumption costs) or by relying on fog nodes of other FIPs to allocate VMs that, if working alone, it could not host (thus incurring into monetary penalties for violating the QoS of the corresponding applications).

It is worth pointing out that each FIP is inclined to cooperate with other FIPs only if it receives enough incentives to do so that make cooperation at least as profitable as working alone. The lack of these suitable incentives leads to the so called unstable coalitions, that is to coalitions where a participating FIP prefers either to leave its current coalition to move to a more profitable one or to work alone. In Section 4, we better formalize this

cooperation process in the framework of the game theory and we propose a distributed algorithm to form stable coalitions among a group of FIPs, so that no FIP in the same coalition has incentive to leave its current coalition to join a better one.

4 | THE COALITION FORMATION GAME

We suppose that the FIPs are rational agents (also referred to as players) being able to make strategic decisions aiming at increasing their profits. To pursue these goals the FIPs can cooperate among them. To this aim, a set of FIPs, may form a *coalition*, i.e., all the FIPs of the coalition must agree to share their own resources and users among them. The coalition formation is a dynamic process where the FIPs move from one to another coalition to improve their utility. One way to describe such a process is to model it as a *coalition formation game*.

In order to join a coalition, a FIP must indeed find it profitable, i.e., it must be sure that the profit it earns by joining the coalition is no worse of the one it obtains by working alone. Furthermore, in order to be sure that this profit is not ephemeral, a FIP must seek other properties that guarantee the suitability of a coalition, namely:

- *Coalition stability*. A coalition is said to be *stable* if no player (or possibly no coalition of players) can deviate from the outcome so as to reach a subjectively better outcome. Several notions of stability have been defined in literature some of which allow to guarantee stability against single player moves (Nash stability) while others also allows group movements (the core, see [13,29] for details). Lack of stability causes possible monetary losses for the following reasons: *i)* a player (in our case a FIP) that has joined a coalition with the expectation of receiving users from other players is penalized if, after switching on additional resources to accommodate these users, these other players leave the coalition; *ii)* a player that has accepted more users than those that it can serve without incurring into a penalty, expecting to use the resources of other players to accommodate them, is penalized if these players leave the coalition.

- *Fairness*. When joining a coalition, a player expects that the resulting profits are fairly divided among participants. As an unfair division leads to instability, a fair profit allocation strategy is mandatory.

From these considerations, it follows that a way must be provided to each FIP to decide whether to participate to a coalition or not and, if so, which one among all the possible coalitions is worth joining. In this paper, we address this issue by modeling the problem of coalition formation as a *coalition formation cooperative game with transferable utility* [12], where each FIP cooperates with the other ones to maximize its net profit rate, and by devising an algorithm to solve it.

In particular, we use the *hedonic coalition formation games* (also referred to as hedonic games) [13,29,30]. A hedonic game is a game where: *i)* the gain of any player depends solely on the members of the coalition to which the player belongs, and *ii)* the coalitions arise as a result of the preferences of the players over their possible set of coalitions. In other words, in this type of coalition games every player is only interested in which players are in its coalition and does not take into account how players in other coalitions are grouped together.

Given the set $\mathcal{N} = \{1, 2, \ldots, N\}$ of FIPs (i.e., the players of the cooperative game), a coalition $\mathcal{C} \subseteq \mathcal{N}$ represents an agreement among the FIPs in $\mathcal{C}$ to act as a single entity: they must agree to share their own resources and users among them.

At any given time, the set of players is partitioned into a *coalition partition* Π, defined as the set $\Pi = \{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_l\}$. That is, for $k = 1, \ldots, l$, each $\mathcal{C}_k \subset \mathcal{N}$ is a disjoint coalition such that $\cup_{k=1}^{l} \mathcal{C}_k = \mathcal{N}$ and $\mathcal{C}_i \cap \mathcal{C}_j = \emptyset$ for $i \neq j$. Given a coalition partition Π, for any player $i \in \mathcal{N}$, we denote by $\mathcal{C}_\Pi(i)$ the coalition that contains player i.

In its partition form, a coalition game is defined on the set $\mathcal{N}$ by associating a utility value $u(\mathcal{C} \mid \Pi)$ to each subset of any partition Π of $\mathcal{N}$. For hedonic games, the utility value of a coalition is independent of the other coalitions and therefore, $u(\mathcal{C} \mid \Pi) = u(\mathcal{C})$. In particular, we define the coalition value $u(\mathcal{C})$ as the net profit rate of coalition $\mathcal{C}$ that we compute as the solution of the profit maximization problem that is presented in Section 4.2.

Obviously, each FIP $i \in \mathcal{C}$ must receive a fraction $\phi_i(\mathcal{C})$ of the coalition value, that we call the *payoff* of i in $\mathcal{C}$. The coalition game we propose aims at getting coalitions in which the FIPs obtain payoffs as high as possible, without violating the fairness requirement, so that stability is achieved.

To this end, we use the Shapley value [31], a payoff allocation rule based on the concept of players' marginal contribution (i.e., the change in the coalition value when a player joins that coalition). This implies that, in a given coalition, the FIP that brings a higher contribution will be rewarded by other FIPs with smaller contributions. In particular, we use the Aumann-Dréze version [32], which is an extension of the Shapley value for games with coalition structures, and it is defined as:

$$\phi_i(\mathcal{C}) = \sum_{\mathcal{B} \subseteq \mathcal{C} \setminus \{i\}} \frac{|\mathcal{B}|!(|\mathcal{C}| - |\mathcal{B}| - 1)!}{|\mathcal{C}|!} \left(u(\mathcal{B} \cup \{i\}) - u(\mathcal{B})\right), \tag{7}$$

where the sum is over all subsets $\mathcal{B}$ not containing i, including the empty set.

For the definition of the coalition formation process, we need to define, for each FIP i, a preference relation $\succeq_i$ that the FIP i can use to compare all the possible coalitions it may join. This requires the definition of a complete, reflexive, and transitive binary relation over all the possible coalitions

that the FIP i can form (see [29]). In other words, for any FIP i, and for any pair of coalitions $\mathcal{C}_1, \mathcal{C}_2 \subseteq \mathcal{N}$ (with $i \in \mathcal{C}_1$ and $i \in \mathcal{C}_2$) the notation $\mathcal{C}_1 \succeq_i \mathcal{C}_2$ denotes that player i prefers being a member of $\mathcal{C}_1$ over $\mathcal{C}_2$, or at least i prefers both coalitions equally. In the coalition formation game we define, for any FIP i we use the following preference relation:

$$\mathcal{C}_1 \succeq_i \mathcal{C}_2 \iff \Phi_i(\mathcal{C}_1) \geq \Phi_i(\mathcal{C}_2) \tag{8}$$

where $\mathcal{C}_1$ and $\mathcal{C}_2$ are any two coalitions that contain FIP i and $\Phi_i(\cdot)$ is a preference function defined as follows:

$$\Phi_i(\mathcal{C}) = \begin{cases} \phi_i(\mathcal{C}) & \text{if } \mathcal{C} \notin h(i) \\ -\infty & \text{otherwise,} \end{cases} \tag{9}$$

where $h(i)$ is a history set where FIP i caches the identity of the coalitions that have been already evaluated so that we avoid generating twice the same candidate coalition (this pruning strategy has been used in several papers, see for instance [33] and [24]).

By replacing the relation $\geq$ with $>$ in Eq. (8), we obtain the preference relation $\succ_i$ that represents the strict counterpart of $\succeq_i$.

4.1 | The Hedonic Coalition Formation Algorithm

In this section, we derive a coalition formation algorithm (see Algorithm 1) that, starting from the initial setting where there are no coalitions (i.e., each FIP is alone) and the history set is empty, allows FIPs to make distributed decisions (asynchronously with respect to the other FIPs) for choosing which coalitions to join at any time. To cope with the distributed nature of our algorithm, we assume the use of suitable distributed state management algorithms (e.g., see [34,35]).

Algorithm 1 is based on the *hedonic shift rule* "$\rightarrow$" (introduced in [36]). Given a coalition partition $\Pi = \{\mathcal{C}_1, \ldots, \mathcal{C}_l\}$ on the set $\mathcal{N}$ and a preference relation $\succ_i$, any FIP $i \in \mathcal{N}$ decides to leave its current coalition $\mathcal{C}_\Pi(i)$ to join another one $\mathcal{C}_k \in \Pi \cup \emptyset$ if and only $\mathcal{C}_k \cup \{i\} \succ_i \mathcal{C}_\Pi(i)$. If the change of coalition from $\mathcal{C}_\Pi(i)$ to $\mathcal{C}_k$ takes place, we refer to it as the hedonic shift from $\mathcal{C}_\Pi(i)$ to $\mathcal{C}_k$ and we denote it by $\{\mathcal{C}_\Pi(i), \mathcal{C}_k\} \rightarrow \{\mathcal{C}_\Pi(i) \backslash \{i\}, \mathcal{C}_k \cup \{i\}\}$.

In other words, player i leaves its current coalition if it receives a greater payoff in another coalition (note that i can decide to go alone, this is the case where $\mathcal{C}_k = \emptyset$). This rule captures the selfish behavior made by a FIP to move from its current coalition to a new one, regardless of the effects of this move on the other FIPs.

The rationale of Algorithm 1 is to let each FIP i search, asynchronously with respect to the other FIPs, the state space of possible coalitions it may join and, for each of them, check whether it is preferable (according to the corresponding $\succ_i$ relation) to join it, or instead to remain in its current coalition. When a FIP decides to leave a coalition to join another one, it inserts the coalition it is leaving into its history set $h(i)$, so that it will not be visited again during the coalition state-space search. A FIP repeats the steps of Algorithm 1 until no more hedonic shift rules can be performed.

Algorithm 1 accepts as parameters the identity i of the calling FIP and the global state $state$, storing the current shared coalition partition Π_c. Initially, before the algorithm is invoked by any FIP, there are no coalitions, meaning that $\Pi_c = \Pi_0 = \big\{\{1\}, \{2\}, \ldots, \{N\}\big\}$.

For each invocation of the algorithm, the calling FIP i initializes its history set h as well as other auxiliary variables (lines 2–4), and then iteratively applies the hedonic shift rules until no more of them can be evaluated from the last coalition partition it considered.

Specifically, after acquiring a lock to get exclusive access to the shared state $state$ (line 6) to ensure its atomic update (through a suitable distributed mutual exclusion algorithm [34]), FIP i iteratively retrieves the current coalition partition Π_c and evaluates (by means of the hedonic shift rule) all the possible coalitions it can form from Π_c, to search for the one with the higher payoff.

To this aim, given the current coalition partition Π_c, for each coalition $\mathcal{C} \in \Pi_c \cup \emptyset$, not contained by its history set h and different from its current one $\mathcal{C}_c$, FIP i evaluates the hedonic shift rule to move to the new coalition $\mathcal{C}_n = \mathcal{C} \cup \{i\}$, and assesses its preference to join the new coalition $\mathcal{C}_n$ against the current coalition $\mathcal{C}_c$ (lines 10–17), according to Eq. (8) and Eq. (9).

If a coalition $\mathcal{C}_b$ with the higher payoff is found (lines 18–22), FIP i inserts into its history set h the coalition $\mathcal{C}_c \setminus \{i\}$ it is leaving, and changes the shared coalition partition by updating both $\mathcal{C}_b$ (now containing also i) and $\mathcal{C}_c$ (now not containing i anymore).

After releasing the exclusive lock to the shared state $state$ (line 23), FIP i repeats the above steps (lines 6–23) to search for a better coalition, in case some other FIP has meantime changed the coalition partition stored in the shared state.

When no other more profitable coalition is found, FIP i terminates the execution of the algorithm (line 24), until a new invocation is performed again.

Given that FIPs can work asynchronously and independently from each other, Algorithm 1 can be easily implemented as a distributed algorithm by means of suitable distributed mechanisms for: *i) state retrieval*, and *ii) atomic state update*. The former mechanism ensures that any FIP is able to obtain the current coalition partition Π_c; while the latter establishes that the current coalition partition Π_c does not change while FIP i is making its decisions. The literature of distributed algorithms provides several proposals to address similar issues (see for instance [37,35] and [34]).

Furthermore, the algorithm we propose ensures two important properties, that are proved below, namely *convergence* and *stability* (in particular we use the notion of *Nash-stability* [38]). The *convergence* ensures that the algorithm always terminates in a finite number of steps, while the *stability*

Algorithm 1 The Hedonic Coalition Formation Algorithm

procedure HEDONICCOALITIONFORMATION(i, $state$)
  h $\leftarrow \emptyset$ ▷ History set
  $\mathcal{C}_c \leftarrow \emptyset$ ▷ Current coalition
  $\mathcal{C}_b \leftarrow \emptyset$ ▷ Best coalition
  repeat
    ACQUIRELOCK($state$)
    $\Pi_c \leftarrow$ GETCURRENTCOALITIONPARTITION($state$)
    $\mathcal{C}_c \leftarrow \mathcal{C}_{\Pi_c}(i)$
    $\mathcal{C}_b \leftarrow \mathcal{C}_c$
    for all $\mathcal{C} \in (\Pi_c \setminus \{\mathcal{C}_c\}) \cup \emptyset$ **and** $\mathcal{C} \notin$ h **do**
      $\mathcal{C}_n \leftarrow \mathcal{C} \cup \{i\}$
      $\phi_{\mathcal{C}_b} \leftarrow$ COMPUTESHAPLEYVALUE($\mathcal{C}_b$, i) ▷ See Eq. (7)
      $\phi_{\mathcal{C}_n} \leftarrow$ COMPUTESHAPLEYVALUE($\mathcal{C}_n$, i) ▷ See Eq. (7)
      if $\phi_{\mathcal{C}_n} > \phi_{\mathcal{C}_b}$ **then** ▷ See Eq. (8) and Eq. (9)
        $\mathcal{C}_b \leftarrow \mathcal{C}_n$
      end if
    end for
    if $\mathcal{C}_b \neq \mathcal{C}_c$ **then**
      UPDATEHISTORY(h, $\mathcal{C}_c \setminus \{i\}$)
      $\Pi_b \leftarrow \left(\Pi_c \setminus \{\mathcal{C}_c, \mathcal{C}_b \setminus \{i\}\}\right) \cup \{\mathcal{C}_c \setminus \{i\}, \mathcal{C}_b\}$ ▷ Hedonic shift: $\{\mathcal{C}_c, \mathcal{C}_b \setminus \{i\}\} \rightarrow \{\mathcal{C}_c \setminus \{i\}, \mathcal{C}_b\}$
      SETCURRENTCOALITIONPARTITION($state$, Π_b)
    end if
    RELEASELOCK($state$)
  until $\mathcal{C}_b = \mathcal{C}_c$
end procedure

ensures that in a stable coalition partition no FIP can benefit from leaving the current coalition to join another one. More formally a partition $\Pi = \{\mathcal{C}_1, \ldots, \mathcal{C}_l\}$ is *Nash-stable* if $\forall i \in \mathcal{N}, \mathcal{C}_\Pi(i) \succeq_i \mathcal{C}_k \cup \{i\}$ for all $\mathcal{C}_k \in \Pi \cup \emptyset$.

Although the proof of convergence and of the Nash-stability are classical results of hedonic games, for completeness we include them.

Proposition 1 (Convergence). Starting from any initial coalition structure $\Pi_0 = \{\{1\}, \{2\}, \ldots, \{N\}\}$, the proposed algorithm always converges to a final partition Π_f.

Proof. The coalition formation process can be mapped to a sequence of hedonic shift rule operations that transforms the current partition Π into another partition Π'. Thus, starting from the initial step, the algorithms yields the following transformations:

$$\Pi_0 \rightarrow \Pi_1 \rightarrow \cdots \rightarrow \Pi \rightarrow \Pi' \tag{10}$$

where the symbol $\rightarrow$ denotes the application of a shift operation. Every application of the shift rule generates two possible cases: (a) $\mathcal{C}_k \neq \emptyset$, so it leads to a new coalition partition, or (b) $\mathcal{C}_k = \emptyset$, so it yields a previously visited coalition partition with a single FIP (i.e., with a coalition of size 1). In the first case, the number of transformations performed by the shift rule is finite (at most, it is equal to the number of partitions, that is the Bell number), and hence the sequence in Eq. (10) will always terminate and converge to a final partition Π_f. In the second case, starting from the previously visited partition, at certain point in time, the non-cooperative FIP must either join a new coalition and yield a new partition, or decide to remain non-cooperative. From this, it follows that the number of re-visited partitions will be limited, and thus, in all the cases the coalition formation stage of the algorithm will converge to a final partition Π_f. □

Proposition 2 (Nash-stability). Any final partition Π_f resulting from Algorithm 1 is Nash-stable.

Proof. We prove it by contradiction. Assume that the final partition Π_f is not Nash-stable. Consequently, there exists a FIP $i \in \mathcal{N}$ and a coalition $\mathcal{C}_k \in \Pi_f \cup \emptyset$ such that $\mathcal{C}_k \cup \{i\} \succ_i \mathcal{C}_{\Pi_f}(i)$. Then, FIP i will perform a hedonic shift operation and hence $\Pi_f \rightarrow \Pi'_f$. This contradicts the assumption that Π_f is the final outcome of our algorithm. □

Hence, the Nash stability is a fundamental guarantee for the convergence of the proposed method. Moreover, the introduction of players' history in the preference relation provides an easy way to prune the state space at each decision step of the algorithm. Indeed, at each step the number of alternatives a player faces fades monotonically to eventually ends up in a stable coalition configuration. Obviously, the process of coalition formation may terminate in a specific configuration just because all further steps are restricted by the history. That is, our algorithm may stop in a local minimum, hence not reaching the best coalition configuration for the specific scenario — which might also be unreachable for different reasons (e.g., coalition instability). However, any approach rising a Nash stable solution of the game would suffer from the local minima solutions. Furthermore, it is worth to point out that Nash-stability also implies the so called *individual-stability* [38]. A partition $\Pi = \{\mathcal{C}_1, \ldots, \mathcal{C}_l\}$ is *individually-stable* if do not exist a player $i \in \mathcal{N}$ and a coalition $\mathcal{C}_k \in \Pi \cup \emptyset$ such that $\mathcal{C}_k \cup \{i\} \succ_i \mathcal{C}_\Pi(i)$ and $\mathcal{C}_k \cup \{i\} \succeq_j \mathcal{C}_k$ for all $j \in \mathcal{C}_k$, i.e., if no player can benefit by moving from her/his coalition to another existing (possibly empty) coalition while not making the members of that coalition worse of. Thus, we can conclude that our algorithm always converges to a partition Π_f which is both Nash-stable and individually stable.

4.2 | Computation of the Optimal Coalition Allocation Profit

In order to derive the payoffs that FIPs receive in a given coalition (see Eq. (7)), our coalition formation algorithm needs to compute the coalition value $u(\mathcal{C})$ (i.e., the coalition net profit rate) for any coalition $\mathcal{C}$ of FIPs that may form.

This requires solving a maximization problem that, given a coalition $\mathcal{C}$ of FIPs located in a specific geographic area g, looks for the best allocation of the VMs $\mathcal{V}$ where to run instances of applications $\mathcal{A} = \bigcup_{i \in \mathcal{C}} App(i)$ to host in this area onto the fog nodes $\mathcal{F} = \bigcup_{i \in \mathcal{C}} FN(i, g)$, in order to maximize the net profit rate $u(\mathcal{C})$ of coalition $\mathcal{C}$. The set $\mathcal{V}$ is obtained by merging the $N_{j,g}(t)$ VMs needed by each application $j \in \mathcal{A}$ to achieve its target QoS in the time interval t.

To this end, we propose the *Mixed Integer Linear Program* (MILP) of Figure 2 to model the problem of allocating a set $\mathcal{V}$ of VMs onto a set $\mathcal{F}$ of fog nodes so that the net profit rate $u(\mathcal{C})$ of the coalition $\mathcal{C}$ is maximized.

In this figure, we adopt the same notation introduced in Section 2 and we denote by $s(j)$ (where $s : \mathcal{V} \rightarrow \mathcal{A}$) the instance of the application run by VM j, and by $p(i)$ (where $p : \mathcal{F} \rightarrow \mathcal{C}$) the FIP which owns fog node i in the specific geographic area g. Furthermore, for readability purposes, we remove from the model any dependence on the time interval t (e.g., we denote by $N_{j,g}$, instead of $N_{j,g}(t)$, the needed number of VMs to allocate so as to achieve the target QoS of application j).

In the optimization model,

- the binary decision variable x_i is set to 1 if fog node i is turned on, and 0 otherwise;
- the binary decision variable $y_{i,j}$ is set to 1 if VM j is allocated on fog node i, and 0 otherwise;
- the non-negative real decision variable u_i denotes the overall share of CPU capacity of fog node i that has been allotted to the hosted VMs;
- the non-negative integer decision variable $n_{k,g}$ represents the number of VMs allocated for executing instances of application k.

The objective function $u(\mathcal{C})$ of the optimization model (defined by Eq. (11a)) represents the net profit rate obtained by the coalition $\mathcal{C}$ of FIPs, which is computed as an extension to coalitions of the net profit rate $P_{i,g}$ defined for a single FIP i (in fact, $u(\{i\}) = P_{i,g}$; see Eq. (6)), that is as the difference between the revenues earned by the allocation of VMs and the costs resulting from the energy consumed by the switched-on fog nodes and from the violations (if any) of applications' QoS. Its maximization must satisfy the following constraints:

- Constraints Eq. (11b) ensures that VMs are not allocated on fog nodes that will be switched off.
- Constraints Eq. (11c) guarantee that each VM is hosted at most on one fog node.
- Constraints Eq. (11d) define the value of variables u_i as the sum of the CPU capacity demands of the VMs allocated on fog node i.
- Constraints Eq. (11e) assure that the allocated CPU capacity of a switched-on fog node is not exceeded.
- Constraints Eq. (11f) define the value of the variable $n_{k,g}$ as the number of allocated VMs where to execute instances of application k.
- Constraints Eq. (11g) guarantee that for each application no more VMs are allocated than required;
- Constraints Eq. (11h) to Eq. (11k) define the domain of decision variables x_i, $y_{i,j}$, u_i and $n_{k,g}$, respectively.

$$\text{maximize } u(\mathcal{C}) = \sum_{k \in \mathcal{A}} R_{Ref(k),k} n_{k,g} - \left[\sum_{i \in \mathcal{F}} \left(x_i W_i^{\min} + (W_i^{\max} - W_i^{\min}) u_i \right) E_{p(i),g} + \sum_{k \in \mathcal{A}} \left(n_{k,g} < N_{k,g} \right) L_{Ref(k),k} \right] \tag{11a}$$

subject to

$$\sum_{j \in \mathcal{V}} y_{i,j} \leq |\mathcal{V}| x_i, \quad \forall i \in \mathcal{F}, \tag{11b}$$

$$\sum_{i \in \mathcal{F}} y_{i,j} \leq 1, \quad \forall j \in \mathcal{V}, \tag{11c}$$

$$u_i = \sum_{j \in \mathcal{V}} y_{i,j} U_{i,j}, \quad \forall i \in \mathcal{F}, \tag{11d}$$

$$u_i \leq x_i, \quad \forall i \in \mathcal{F}, \tag{11e}$$

$$n_{k,g} = \sum_{i \in \mathcal{F}} \sum_{\substack{j \in \mathcal{V}, \\ s(j)=k}} y_{i,j}, \quad \forall k \in \mathcal{S}, \tag{11f}$$

$$n_{k,g} \leq N_{k,g}, \quad \forall k \in \mathcal{A}, \tag{11g}$$

$$x_i \in \{0, 1\}, \quad \forall i \in \mathcal{F}, \tag{11h}$$

$$y_{i,j} \in \{0, 1\}, \quad \forall i \in \mathcal{F}, j \in \mathcal{V}, \tag{11i}$$

$$u_i \in \mathbb{R}^*, \quad \forall i \in \mathcal{F}, \tag{11j}$$

$$n_{k,g} \in \mathbb{N}, \quad \forall k \in \mathcal{A}. \tag{11k}$$

FIGURE 2 The optimization model for maximizing the net profit rate $u(\mathcal{C})$ of coalition $\mathcal{C}$.

TABLE 1 Parameters used in the experimental scenarios. Subscripts i and j take values on the set $\{1, 2, 3, 4\}$.

	Parameter	Value
$\lvert App(i) \rvert$	Number of applications associated to FIP i	1
$E_{i,j}$	Electricity price charged to FIP i in area j	0.0001 $/Wh
$\lvert FN(i,j) \rvert$	Number of fog nodes owned by FIP i in area j	4
$L_{i,j}$	Penalty rate paid by FIP i for violating QoS of application j	0.022 $/h
N	Number of FIPs	4
M	Number of applications	4
Q_i	Maximum request processing time defined by the QoS of application i	0.7 sec
$R_{i,j}$	Revenue rate earned by FIP i for running instances of application j	0.0022 $/h
$U_{k,j}$	CPU requirement of template VM_j on fog node k	0.05
$W_k^{\max}$	Maximum power consumption of fog node k	200 W
$W_k^{\min}$	Idle power consumption of fog node k	100 W
μ_j	Request processing rate of template VM_j	2 req/sec

5 | EXPERIMENTAL EVALUATION

In this section, we present the experimental evaluation we performed through simulation to assess the performance and the efficacy of our algorithm to form profitable and stable coalitions of FIPs. To do so, we run Algorithm 1 for different scenarios, where we vary the workload of a

set of applications. The results of this evaluation show that our algorithm is able to form coalitions of FIPs that are profitable and stable in all the considered scenarios.

To perform our experiments, we implemented a specific simulator in C++ which uses the IBM ILOG CPLEX solver 12.8 [39] to solve the maximization problem of Section 4.2.

The remainder of this section is organized as follows. First, in Section 5.1, we provide the experimental settings we use in our scenarios. Then, in Section 5.2, we present the results we obtain by running our coalition formation algorithm in these scenarios (Section 5.2).

5.1 | Experimental Setup

In this section, we present the experimental settings and scenarios considered in our evaluation. To ease readability, they are summarized in Table 1, Table 2, and Table 3.

In a specific geographic area g, we consider $N = 4$ identical FIPs, each of which is in charge of running instances of a different application (i.e., $App(i) = \{S_i\}$, for $i = 1, \ldots, 4$). The physical infrastructure of each FIP consists of 4 identical fog nodes whose idle and maximum power consumptions W_k^{min} and W_k^{max} are set to 100 W and 200 W, respectively (for $k = 1, \ldots, 4$). This number of fog nodes ensures that, in the scenarios we consider, each FIP, when working in cooperation, is able to accept on its fog nodes all the workload of the other FIPs that are member of the same coalition. The template VM VM_j of each application j is characterized by a request processing rate μ_j of 2 requests per second and, to achieve this value, it needs a physical CPU capacity $U_{k,j}$ of 0.05 for each fog node k (i.e., when allocated to a fog node k, every VM consumes 5% of its physical CPU capacity).

We suppose that the electricity price $E_{i,g}$ is charged hourly to every FIP i and we set it to 0.0001 \$/Wh [40] for every FIP. Furthermore, we fix the revenue rate $R_{i,j}$ that each FIP i receives for hosting a VM where to run an instance of application j to 0.0022 \$/hour, which is 22 times the electricity price $E_{i,g}$. We derived this value by assuming that each FIP reaches the break-even point when the load of one of its fog node (i.e., the total CPU capacity of the fog node allocated to VMs) is 30%. This value depends on the specific parameters we set for our experimental evaluation, and such parameters have been selected to study the formation of coalitions with different structures as function of the load. On the one hand, this value is sufficiently large that each FIP is willing to cooperate with all the other FIPs at low loads (by hosting the VMs of all FIPs on a single fog node), but on the other hand, it is sufficiently small that an FIP is willing to join a coalition only with some other FIP at medium loads or to not join at all at high loads (to avoid paying energy consumption costs that it cannot amortize).

Finally, for every application j, we fix its QoS target Q_j to 0.7 sec and the associated penalty rate $L_{i,j}$ to 0.022 \$/hour for every FIP i (i.e., $L_{i,j} = 10R_{i,j}$). We select this value for the penalty rate to ensure that an FIP always prefers allocating VMs for the applications it hosts than refusing them for cutting down energy costs.

We study the impact of the workload on the coalition formation process by varying, in a controlled way (i.e., by considering each discretization interval of the traffic load curve separately), the workload intensity of each application.

To this aim, we consider two set of scenarios, namely the *homogeneous scenarios set* and the *heterogeneous scenarios set*, respectively, whose settings are given in Table 2 and Table 3.

In the *homogeneous scenarios set*, we vary the workload intensity $\lambda_{k,g}$ of each application k (in the given area g) so that the induced load α_i, experienced by each FIP i (when it works alone) on the fog node where the required VMs have been allocated, ranges from 0.1 (i.e., only 10% of the CPU capacity of the fog node is allocated to VMs) to 0.9 (i.e., the total allocated CPU capacity on the fog node is 90%), with increments of 10%. The resulting scenarios are summarized in Table 2, where the first column represents the scenario name, the second column contains the load level of each application k stated in terms of its workload intensity $\lambda_{k,g}$ in the geographic area of interest g, the third column gives the minimum number $N_{k,g}$ of VMs required to satisfy the QoS parameter Q_k of application k in face of the workload $\lambda_{k,g}$, and the last column contains the load α_i induced on a fog node of FIP i by the workload intensity $\lambda_{k,g}$.

To better evaluate the game dynamics, we also consider the *heterogeneous scenarios set*, where the reference scenario for the experiments has been changed but, in order to be able to compare different scenarios, the average load offered to all FIPs has been kept fixed and equal to the previous set, as reported in Table 3. Specifically, fixed a given average load $\hat{\alpha}$ offered to all FIPs, we compute the load α_i associated to each FIP i as follows: $\alpha_1 = \left\lceil \frac{(3/2)\hat{\alpha}}{U_{1,j}} \right\rceil U_{1,j}$, $\alpha_2 = \left\lceil \frac{(5/4)\hat{\alpha}}{U_{2,j}} \right\rceil U_{2,j}$, $\alpha_3 = \left\lceil \frac{(1/2)\hat{\alpha}}{U_{3,j}} \right\rceil U_{3,j}$, and $\alpha_4 = \left\lceil \frac{(3/4)\hat{\alpha}}{U_{4,j}} \right\rceil U_{4,j}$, where $\lceil \cdot \rceil$ is the ceiling operator, and the use of $U_{i,j}$ in these equations is to assure that the resulting load is a multiple of the CPU demand $U_{i,j}$ of VM_j on a fog node of FIP i.

$\lambda_{k,g}$	$N_{k,g}$	α_i
2.1	2	0.1
5.7	4	0.2
9.4	6	0.3
13.3	8	0.4
17.2	10	0.5
21.1	12	0.6
25.0	14	0.7
28.9	16	0.8
32.8	18	0.9

TABLE 2 The experimental settings corresponding to the homogeneous scenarios set. Subscripts i and k take values on the set $\{1, 2, 3, 4\}$.

$\hat{\alpha}$	FIP i	$\lambda_{k,g}$	$N_{k,g}$	α_i
0.2	1	9.4	6	0.30
	2	7.6	5	0.25
	3	2.1	2	0.10
	4	3.9	3	0.15
0.5	1	26.9	15	0.75
	2	23.0	13	0.65
	3	7.6	5	0.25
	4	13.3	8	0.40
0.8	1	44.7	24	1.20
	2	36.8	20	1.00
	3	13.3	8	0.40
	4	21.1	12	0.60

TABLE 3 The experimental settings corresponding to the heterogeneous scenarios set. Subscripts i and k take values on the set $\{1, 2, 3, 4\}$.

5.2 | Experimental Results

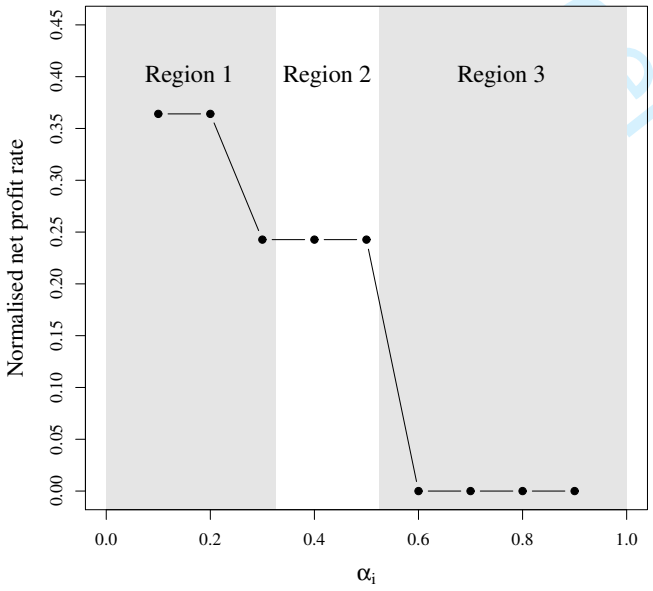

FIGURE 3 Changes in the net profit rate of FIP i (with $i = 1, \ldots, 4$) for different values of load level α_i when using the algorithm in a homogeneous load scenario.

$\hat{\alpha}$	FIP i	u({i})	E[ϕ_i]	Normalised Δ(E[ϕ_i], u({i}))	Average Normalised Δ(E[ϕ_i], u({i}))	Formed Coalition Partition
0.2	1	0.0002	0.0077	0.9740	0.9740	{1, 2, 3, 4}
	2	-0.0015	0.006	0.9740		
	3	-0.0066	0.0009	0.9740		
	4	-0.0049	0.0026	0.9740		
0.5	1	0.0155	0.0172	0.0971	0.1456	{1, 2, 3, 4}
	2	0.01	0.0138	0.0971		
	3	-0.0015	0.0035	0.2913		
	4	0.0036	0.0053	0.0971		
0.8	1	0.0208	0.0241	0.1381	0.1036	{2}, {1, 3, 4}
	2	0.024	0.024	0		
	3	0.0036	0.0069	0.1381		
	4	0.0104	0.0137	0.1381		

TABLE 4 The experimental scenarios corresponding for the heterogeneous loads scenario. Subscript i takes values on the set $\{1, 2, 3, 4\}$. The symbol E[·] denotes the arithmetic mean operator.

The configuration summarised in Table 2 applies to all FIPs, smoothing the effect induced by the payoff allocation. Indeed, as stated before, the Shapley value keeps into account the contribution each player brings to the coalition and compute rewards accordingly. The assumption of homogeneous loads allows us to focus on the effects induced by the cooperation among FIPs, neglecting the dynamics leading to it – any player joining a specific coalition brings the same contribution as the others. Specifically, Figure 3 depicts the ratio between the net profit rate – i.e., the difference between payoff obtained applying the proposed methodology and the case in which all FIPs act independently (i.e., without cooperation) – and the maximum payoff the players can get among all scenarios. Clearly, such normalisation aims to provide a measure describing the benefits induced by the methodology decoupling it from the specific costs and revenues in the scenario. Figure 3 shows three different regions, each one of

which corresponds to a specific load range – from low to high traffic load. Furthermore, each region identifies different coalition configurations and net profit rates – except for the point $\alpha_i = 0.3$.

In **Region 1** service requests arrive with a low rate to all FIPs allowing them to coalesce in a single coalition. Such configuration holds until the point $\alpha_i = 0.3$ after which optimising the use of resources within the grand coalition it is not any longer possible. Hence, further increments of loads generate scenarios in the **Region 2**. Here players autonomously organise themself in smaller coalitions to rationalise the use of resources. Eventually, as the computational capacity available decreases also FIPs propensity to cooperate fades. Indeed, within **Region 3** the cooperation costs grow higher than benefits – i.e., FIPs would incur in penalties – and each one act selfishly. The step behaviour observed in Figure 3 is induced by the discrete granularity of VMs allocation in the maximisation problem and would smooth in case of smaller VMs scenarios.

In Table 4, we present the results of the heterogeneous scenarios set whose configuration is summarized in Table 3. Here, only one point for each region of load has been reported but is enough to observe the effects of heterogeneity. In the first place it is possible to notice that some of the observations made in the previous case still hold: an increment in the loads decreases the benefits of cooperation — i.e., the number of fog nodes switched off decreases as the incoming service requests increases. On the other hand, heterogeneous loads allow new coalitions to arise. Indeed, in **Region 3** it is possible to notice that FIPs are now able to coalesce and obtain a non-zero payoff also where in the homogeneous scenario was not possible.

This set of experiments allows to point out that the cooperation brings greater benefits in case of low loads. On the other hands, in case of high loads the advantages decrease as load increases. This behaviour is due to the reduction of waste of resources that occurs in case of under utilized FIPs. Eventually, evaluating the algorithm in a more realistic scenario — i.e., the one with heterogeneous loads — it has been showed that the cooperation is beneficial also in cases of biased traffic load, where overloaded FIPs take advantage of other FIPs with low traffic loads and vice versa.

Despite of its simplicity, the set of experiments summarized by Figure 3 and in Table 4, illustrates how to use the distributed coalition formation algorithm. The FIPs, based on its own traffic profile estimates, can agree the timing and the activation frequency of the distributed algorithm. The goal should be an appropriate trade-off between the benefits due to the algorithm (e.g., obtaining coalitions that allow the reduction of costs) and the overhead derived from too frequent and not very effective activations.

6 | RELATED WORK

The idea of forming coalitions of different operators, using a game theoretical approach, with the aim of improving their net profit or to obtain different benefits, has been already investigated in the past in various scenarios.

Profit maximization has been investigated both for cloud computing [41,25,42,43] and cellular networks [24]. Coalition formation frameworks have been used instead for femtocell networks for purposes different from profit maximization, such as interference mitigation, and resource and power allocation [44,45,46].

Compared to these works, in our contribution we consider a much different system architecture, which is characterized by different properties and, hence, requires a different solution.

Approaches based on game theory have also been used in the field of fog computing (or edge computing/femto-cloud) for resource optimization [47,48], latency and/or the energy consumption reduction [49,50], computation offloading [51,52], and resource aggregation into a single distributed system [53,54].

In this work we target a goal which is different from those addressed by the above papers. In particular, we focus on the problem of increasing the profit of different FIPs in the presence of applications characterized by different QoS targets and time-varying workloads.

Last but not least, this paper extends our previous work [14] by introducing a different coalition formation allocation strategy that improves the stability of the formed coalitions, guarantees the efficiency of the payoff allocation, and ensures fairness in the payoff allocation thanks to the use of the Shapley value (note that a fair payoff allocation is a mandatory property to convince autonomous agents (often competing) to adopt the solution we propose).

7 | CONCLUSIONS

In this paper we have considered the problem of studying under what conditions a set of FIPs, each one needing to allocate its fog nodes to a set of IoT applications with the goal of meeting specific QoS targets in face of time-varying workloads, finds profitable to cooperate among them, and of devising a suitable decision algorithm allowing each one of them to decide independently whether to join a coalition with other FIPs or not.

We addressed the above problems by means of game-theoretical framework, that allowed us to study the problem of forming coalitions of FIPs, and by devising a mathematical optimization model for the computation of the allocation of IoT applications on the resources of the various FIPs so as to improve their net profits.

In the proposed scheme, we model the cooperation among FIPs as a cooperative game with transferable utility and we design a distributed algorithm to form coalitions of FIPs. With the proposed algorithm, each FIP individually decides whether to leave the current coalition to join a different one according to his preference, meanwhile improving the perceived net profit. Moreover, we proved that the devised algorithm converges to a Nash-stable coalition partition which determines the resulting coalition structure. Numerical results demonstrate the effectiveness of our approach.

As future work, we plan to proceed along different directions. First, we plan to explore ways to improve the coalition value function in order to account for possible request losses due to lack of physical resources. Second, we plan to explore ways to improve the game-theoretic and optimization models in order to include costs like the loss of revenues, as well as other aspects like the ones related to trustworthiness among FIPs. Finally, we plan to implement and validate the proposed solution on a real testbed.

ACKNOWLEDGMENTS

This research is original and has a partial financial support of the Università del Piemonte Orientale.

References

1. Hassanalieragh M, Page A, Soyata T, et al. Health monitoring and management using Internet-of-Things (IoT) sensing with cloud-based processing: Opportunities and challenges. In: Proc. of the 2015 IEEE International Conference on Services Computing (SCC). IEEE; 2015: 285–292
2. Perera C, Qin Y, Estrella JC, Reiff-Marganiec S, Vasilakos AV. Fog Computing for Sustainable Smart Cities: A Survey. *ACM Computing Surveys* 2017; 50(3): 32:1–32:43. doi: 10.1145/3057266
3. Vasisht D, Kapetanovic Z, Won J, et al. FarmBeats: An IoT Platform for Data-Driven Agriculture. In: Proc. of the 14th USENIX Symposium on Networked Systems Design and Implementation (NSDI'17). USENIX Association; 2017: 515–529.
4. Byers CC. Architectural Imperatives for Fog Computing: Use Cases, Requirements, and Architectural Techniques for Fog-Enabled IoT Networks. *IEEE Communications Magazine* 2017; 55(8): 14–20. doi: 10.1109/MCOM.2017.1600885
5. Bonomi F, Milito R, Natarajan P, Zhu J. Fog Computing: A Platform for Internet of Things and Analytics. In: Bessis N, Dobre C., eds. *Big Data and Internet of Things: A Roadmap for Smart Environments*. vol. 546 of *Studies in Computational Intelligence*. Springer. 2014 (pp. 169–186)
6. Bonomi F, Milito R, Zhu J, Addepalli S. Fog Computing and Its Role in the Internet of Things. In: 1st Workshop on Mobile Cloud Computing (MCC). ACM; 2012; Helsinki, Finland: 13–16
7. Yousefpour A, Fung C, Nguyen T, et al. All One Needs to Know about Fog Computing and Related Edge Computing Paradigms: A Complete Survey. *arXiv preprint arXiv:1808.05283* 2018.
8. Hu P, Dhelim S, Ning H, Qiu T. Survey on fog computing: architecture, key technologies, applications and open issues. *Journal of Network and Computer Applications* 2017; 98(Supplement C): 27–42. doi: 10.1016/j.jnca.2017.09.002
9. Liu Y, Fieldsend JE, Min G. A Framework of Fog Computing: Architecture, Challenges, and Optimization. *IEEE Access* 2017; 5. doi: 10.1109/ACCESS.2017.2766923
10. Weinman J. The Economics of the Hybrid Multicloud Fog. *IEEE Cloud Computing* 2017; 4(1): 16–21. doi: 10.1109/MCC.2017.13
11. 451 Research . Customer Insight: Future-proofing your colocation business. white paper, Schneider Electric; Rueil-Malmaison, France: 2017.
12. Peleg B, Sudhölter P. *Introduction to the Theory of Cooperative Games*. Springer-Verlag Berlin Heidelberg. 2nd ed. 2007
13. Drèze J, Greenberg J. Hedonic Coalitions: Optimality and Stability. *Econometrica* 1980; 48(4): 987–1003. doi: 10.2307/1912943
14. Anglano C, Canonico M, Castagno P, Guazzone M, Sereno M. A game-theoretic approach to coalition formation in fog provider federations. In: 2018 Third International Conference on Fog and Mobile Edge Computing (FMEC). IEEE; 2018; Barcelona, Spain: 123–130

15. Yi S, Li C, Li Q. A Survey of Fog Computing: Concepts, Applications and Issues. In: Proc. of the 2015 Workshop on Mobile Big Data (Mobidata). ACM; 2015; Hangzhou, China: 37–42

16. Barham P, Dragovic B, Fraser K, et al. Xen and the Art of Virtualization. In: Proc. of the 19th ACM Symposyum on Operating Systems Principles (SOSP). ACM; 2003; Bolton Landing, NY, USA: 164–177

17. Albano L, Anglano C, Canonico M, Guazzone M. Fuzzy-Q&E: Achieving QoS Guarantees and Energy Savings for Cloud Applications with Fuzzy Control. In: Proc. of the 2013 International Conference on Cloud and Green Computing. IEEE; 2013; Karlsruhe, Germany: 159–166

18. Anglano C, Canonico M, Guazzone M. FC2Q: exploiting fuzzy control in server consolidation for cloud applications with SLA constraints. *Concurrency and Computation: Practice and Experience* 2015; 27(17): 4491–4514. doi: 10.1002/cpe.3410

19. Anglano C, Canonico M, Guazzone M. FCMS: A fuzzy controller for CPU and memory consolidation under SLA constraints. *Concurrency and Computation: Practice and Experience* 2017; 29(5). doi: 10.1002/cpe.3968

20. Anglano C, Canonico M, Guazzone M. Prometheus: A flexible toolkit for the experimentation with virtualized infrastructures. *Concurrency and Computation: Practice and Experience*; 30(11): e4400. doi: 10.1002/cpe.4400

21. Primate Labs, Inc. . GeekBench: Next-Generation Processor Benchmark. Available from: https://www.geekbench.com; 2018. Accessed: Sept. 27, 2018.

22. Rivoire S, Ranganathan P, Kozyrakis C. A Comparison of High-level Full-system Power Models. In: 2008 Conference on Power Aware Computing and Systems (HotPower). USENIX Association; 2008; San Diego, CA, USA: 1–5.

23. Boiardi S, Capone A, Sansó B. Radio planning of energy-aware cellular networks. *Computer Networks* 2013; 57(13): 2564–2577. doi: 10.1016/j.comnet.2013.05.003

24. Anglano C, Guazzone M, Sereno M. Maximizing profit in green cellular networks through collaborative games. *Computer Networks* 2014; 75, Part A: 260–275. doi: 10.1016/j.comnet.2014.10.003

25. Guazzone M, Anglano C, Sereno M. A Game-Theoretic Approach to Coalition Formation in Green Cloud Federations. In: 2014 14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid). IEEE; 2014; Chicago, IL, USA: 618–625

26. Bahreini T, Grosu D. Efficient Placement of Multi-component Applications in Edge Computing Systems. In: Proc. of the 2nd ACM/IEEE Symposium on Edge Computing (SEC). ACM; 2017; San Jose, CA, USA: 5:1–5:11

27. Anglano C, Canonico M, Guazzone M. Profit-aware Resource Management for Edge Computing Systems. In: Proc. of the 1st International Workshop on Edge Systems, Analytics and Networking (EdgeSys). ACM; 2018; Munich, Germany: 25–30

28. Harchol-Balter M. *Performance Modeling and Design of Computer Systems: Queueing Theory in Action.* Cambridge University Press . 2013.

29. Bogomolnaia A, Jackson MO. The Stability of Hedonic Coalition Structures. *Games Economic Behavior* 2002; 38: 201–230. doi: 10.1006/game.2001.0877

30. Aziz H, Savani R. Hedonic Games. In: Brandt F, Conitzer V, Endriss U, Lang J, Procaccia AD. , eds. *Handbook of Computational Social Choice*Cambridge University Press. 2016 (pp. 356–376)

31. Shapley LS. A Value for n-person Games. In: Kuhn H, Tucker A., eds. *Contributions to the Theory of Games (AM-28)*. Volume II. Princeton University Press. 1953 (pp. 307–318)

32. Aumann R, Dréze J. Cooperative games with coalition structures. *International Journal of Game Theory* 1974; 3(4): 217–237. doi: 10.1007/BF01766876

33. Saad W, Han Z, Başar T, Debbah M, Hjørungnes A. Hedonic Coalition Formation for Distributed Task Allocation among Wireless Agents. *IEEE Transactions on Mobile Computing* 2011; 10(9): 1327–1344. doi: 10.1109/TMC.2010.242

34. Kshemkalyani A, Singhal M. *Distributed Computing: Principles, Algorithms, and Systems.* Cambridge University Press . 2008.

35. Weiss G., ed.*Multiagent Systems.* MIT Press. 2nd ed. 2013.

36. Saad W, Han Z, Başar T, Debbah M, Hjørungnes A. A selfish approach to coalition formation among unmanned air vehicles in wireless networks. In: Proc. of the 2009 International Conference on Game Theory for Networks (GameNets). IEEE; 2009

37. Coulouris G, Dollimore J, Kindberg T, Blair G. *Distributed Systems: Concepts and Design.* Addison Wesley. 5th ed. 2011.

38. Bogomolnaia A, Jackson M. The Stability of Hedonic Coalition Structures. *Game Econ Behav* 2002; 38: 201–230.

39. IBM . ILOG CPLEX Optimization Studio. Available from: https://www.ibm.com/products/ilog-cplex-optimization-studio; 2018. Accessed: Sept. 27, 2018.

40. U.S. Energy Information Administration (EIA) . Average Price of Electricity to Ultimate Customers. Available from: https://www.eia.gov/electricity; 2018. Accessed: Sept. 27, 2018.

41. Kaewpuang R, Niyato D, Wang P, Hossain E. A Framework for Cooperative Resource Management in Mobile Cloud Computing. *IEEE Journal on Selected Areas in Communications* 2013; 31(12): 2685–2700. doi: 10.1109/JSAC.2013.131209

42. Lee CA. Cloud Federation Management and Beyond: Requirements, Relevant Standards, and Gaps. *IEEE Cloud Computing* 2016; 3(1): 42–49. doi: 10.1109/MCC.2016.15

43. Mashayekhy L, Nejad MM, Grosu D. Cloud Federations in the Sky: Formation Game and Mechanism. *IEEE Transactions on Cloud Computing* 2015; 3(1): 14–27. doi: 10.1109/TCC.2014.2338323

44. Pantisano F, Bennis M, Saad W, Debbah M. Spectrum leasing as an incentive towards uplink macrocell and femtocell cooperation. *IEEE Journal on Selected Areas in Communications* 2012; 30(3): 617–630. doi: 10.1109/JSAC.2012.120411

45. Zhang Z, Song L, Han Z, Saad W. Coalitional games with overlapping coalitions for interference management in small cell networks. *IEEE Transactions on Wireless Communications* 2014; 13(5): 2659–2669. doi: 10.1109/TWC.2014.032514.130942

46. Pantisano F, Bennis M, Saad W, Debbah M, Latvaaho M. Interference alignment for cooperative femtocell networks: a game-theoretic approach. *IEEE Transactions on Mobile Computing* 2013; 12(11): 2233–2246. doi: 10.1109/TMC.2012.196

47. Barbarossa S, Sardellitti S, Di Lorenzo P. Joint allocation of computation and communication resources in multiuser mobile cloud computing. In: Proc. of the IEEE 14th Workshop on Signal Processing Advances in Wireless Communications (SPAWC). IEEE; 2013: 26–30

48. Meng S, Wang Y, Miao Z, Sun K. Joint optimization of wireless bandwidth and computing resource in cloudlet-based mobile cloud computing environment. *Peer-to-Peer Networking and Applications* 2017; 11(3): 462–472. doi: 10.1007/s12083-017-0544-x

49. Chen MH, Dong M, Liang B. Joint offloading decision and resource allocation for mobile cloud with computing access point. In: Proc. of the 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE; 2016: 3516–3520

50. Dinh T, Tang J, La Q, Quek T. Offloading in mobile edge computing: Task allocation and computational frequency scaling. *IEEE Transactions on Communications* 2017; 65(8). doi: 10.1109/TCOMM.2017.2699660

51. Barbarossa S, Sardellitti S, Di Lorenzo P. Computation offloading strategies based on energy minimization under computational rate constraints. In: Proc. of the 23rd European Conference on Networks and Communications (EuCNC). IEEE; 2014: 1–5

52. Munoz-Medina O, Pascual-Iserte A, Vidal J. Optimization of radio and computational resources for energy efficiency in latency-constrained application offloading. *IEEE Transactions on Vehicular Technology* 2014; 64(10). doi: 10.1109/TVT.2014.2372852

53. Oueis J, Calvanese-Strinati E, Barbarossa S. Small cell clustering for efficient distributed cloud computing. In: Proc. of the 2014 IEEE 25th Annual International Symposium on Personal, Indoor, and Mobile Radio Communication (PIMRC). IEEE; 2014: 1474–1479

54. Tanzil S, Gharehshiran O, Krishnamurthy V. A Distributed Coalition Game Approach to Femto-Cloud Formation. *IEEE Transactions on Cloud Computing* 2016; PP(99). doi: 10.1109/TCC.2016.2594175