A Game-Theoretic Approach to Coalition Formation in Fog Provider Federations (Extended Version)

Cosimo Anglano*, Massimo Canonico*, Paolo Castagno†, Marco Guazzone* and Matteo Sereno†

*DiSIT, Computer Science Institute, University of Piemonte Orientale, Italy

†Department of Computer Science, University of Torino, Italy

***Abstract*—In this paper we deal with the problem of making a set of *Fog Infrastructure Providers* (FIPs) increase their profits when allocating their resources to process the data generated by IoT applications that need to meet specific QoS targets in face of time-varying workloads. We show that if FIPs cooperate among them, by mutually sharing their workloads and resources, then each one of them can improve its net profit. By using a game-theoretic framework, we study the problem of forming *stable* coalitions among FIPs. Furthermore, we propose a mathematical optimization model to allocate IoT applications to a set of FIPs, in order to reduce costs and, at the same time, to meet the corresponding QoS targets. Based on this, we propose an algorithm, based on cooperative game theory, that enables each FIP to decide with whom to cooperate in order to increase its profits. The effectiveness of the proposed algorithm is demonstrated through an experimental evaluation considering various workload intensities. The results we obtain from these experiments show the ability of our algorithm to form coalitions of FIPs that are stable and profitable in all the scenarios we consider.**

***Index Terms*—Fog computing, Fog federation, Game Theory, Coalition Formation**

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 require real-time or near real-time processing [5].

Fog Computing [6] has recently emerged as the most suitable solution to the processing needs of IoT data, and has received a lot of attention in the literature [7], [8]. In Fog Computing, compute, storage, and network resources are provided by *fog nodes* that are placed at the edge of the network, in close proximity to where data are generated. In this way, IoT data can be processed at the edge of the network, without the need of transferring it to a centralized data center, thus greatly reducing latency and avoiding to congest the core network, unlike the cloud paradigm that, instead, requires such a transfer.

However, as argued in [9], latency is proportional to the inverse square root of the number of fog nodes, so a very large number of fog nodes is required to reduce it (for instance, it takes four times as many nodes to reduce latency by half). Hence, latency reduction would be prohibitively costly for an enterprise that had to purchase and deploy its own fog nodes, since a very large number of nodes would be necessary to suitably reduce latency for data generated by a population of moving users. Furthermore, these costs would be hardly amortized, since the capacity of these fog nodes would be used only in part for most of the time, because of the variations in volume, variety, and velocity of generated data [5].

The emergence of *Fog Infrastructure Providers* (FIPs), i.e., operators that provide individual enterprises with the computing and networking infrastructure needed to host their fog nodes on a pay-per-use basis, will however eliminate these costs [9], since these enterprises do not need to purchase, deploy, and manage their own fog infrastructures. On the other hand, by multiplexing the same physical infrastructure among multiple tenants, each FIPs can maximize the utilization of its resources, thus amortizing capital and operational costs and making profit.

Furthermore, in many cases (e.g., small urban centers or rural areas) FIPs will be regional providers that exploit co-location facilities [9] to cut down operational costs, a solution that is growing fast in the small-to-medium-size enterprise arena [10].

In the case where several FIPs share the same co-location facility (and, hence, their resources are indistinguishable from the perspective of latency perceived by a user), their profits can be further increased if they cooperate among them by mutually sharing their users and infrastructures. In particular, each FIP can improve its net profit by either (a) reducing energy costs by switching off (a part of) its infrastructure and offloading its users to resources belonging to other cooperating FIPs, or (b) increasing its earnings by attracting users from other FIPs, or (c) relying on resources of other FIPs to accept more users than what could do by working alone.

Obviously, it is unreasonable to expect that each FIP is willing to unconditionally cooperate with the other ones regardless the benefits it receives. Such a cooperation arises indeed only if suitable benefits result from it, and if the risks of monetary losses (caused by violations in the QoS parameters negotiated with its clients) are kept within acceptable limits.

In this paper, we devise a decision algorithm that provides a set of FIPs with suitable means to decide whether to cooperate with other FIPs, and if so with whom to cooperate. Our algorithm is based on game-theoretic techniques, where the process of establishing cooperation among the FIPs is modeled as a *cooperative game with transferable utility* [11] (in particular, as a *hedonic game* [12], whereby each FIP bases its decision on its own preferences).

More specifically, we propose a game-theoretic framework to study the problem of forming *stable* coalitions among FIPs, and a mathematical optimization model to allocate IoT services to a set of resources, in order to improve profits and, at the same time, to meet service QoS for FIPs inside the same coalition. We achieve our goal by devising a *coalition formation algorithm* to form stable coalitions that allows 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.

In our approach, each FIP pays for the energy consumed to serve each service, whether it belongs to it or to another FIP, but receives a payoff (computed as discussed later) for doing so. We prove that the proposed algorithm converges to a *Nash-stable* set of disjoint coalitions [13], whereby no FIP can benefit to leave the current coalition to join a different one.

Our solution adopts an asynchronous approach in which each FIP autonomously makes its own decisions, and where the best solution arises without the need to synchronize them or to resort to a trusted third party. As a consequence, the solution we propose can be readily implemented in a distributed fashion.

To demonstrate the effectiveness of the algorithm we propose, we carry out a thorough experimental evaluation considering real-world traffic traces, and a set of realistic scenarios. The results we obtain indicate that our algorithm allows indeed a population of FIPs to significantly improve their profits thanks to the combination of energy reduction and satisfaction of QoS requirements.

The contributions of this paper can be summarized as follows:

- we consider 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 operators to find the coalition improving their profits under stability concerns.

The rest of this paper is organized as follows. Section 2 describes the details of the system model, and it presents the problem statement. Section 3 introduces the coalition formation game and we evaluate the system in Section 4 using simulations. Finally, we end the paper with related work (in Section 5) and conclusions (in Section 6).

2. System Model and Problem Statement

2.1. System Model

We consider a system, whose architecture is schematically depicted in Figure 1, where a large population of geo-distributed IoT devices, located in a set of n distinct geographic areas, generates very large amounts of data that require real-time or near-real-time processing. These devices may be either stationary (e.g., smart homes, IP cameras, smart traffic lights, etc.) or mobile (e.g., smartphones, tablets, connected cars, etc.).

Without loss of generality, we assume that each area j is covered by a single co-location facility, each one hosting a set of fog nodes that run a set of applications in charge of processing the data generated by the devices. These applications are encapsulated into a set of Virtual Machines (VMs), hosted on the fog nodes. In the rest of this section, we describe the characteristics of the three main components of the system, namely the fog nodes, the applications, and the VMs.

2.1.1. The Fog Nodes. We assume that the various co-location facilities host a set of resource-rich fog nodes [14] (e.g., *Cloudlets* [15] or *Micro Data Centers* [16]) belonging to a population of m FIPs. For instance, the co-location facility of area j in Figure 1 hosts fog nodes belonging to FIPs x and y.

Each fog node z is characterized by its CPU capacity c_z, which is measured by means of a suitable benchmark (e.g., GeekBench [17]), and by its power consumption $w_z(u)$, which is computed as [18]:

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

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

We assume all the fog nodes of the same FIP i located in a given area j are identical among them, i.e. they are characterized by the same CPU capacity and energy consumption.

2.1.2. The Applications. The data generated by the IoT devices are processed by a set of K distributed applications $\mathcal{S} = \{S_1, S_2, \ldots, S_K\}$. Each application S_i is associated with its *owner*, and is present with one or more instances in the various areas.

The instances of S_i located in a given area j process all the data generated for S_i by the devices located in that area, with the possible exception of data that are forwarded to a cloud data center by the corresponding *Smart Gateway* (which is in charge of deciding whether data must be uploaded to the cloud or not [19]). We assume that the load requests arriving to S_i in a given area are fairly split among the instances of S_i allocated in that area.

We assume that the load of data processing requests submitted to application S_i in any area j varies over time, and is described by its *load profile curve* $l(i, j)$ expressing, as function of time, the rate at which requests are submitted. 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 characterized by its QoS target, which is quantified by the maximum value Q_i that the average request processing time T_i can take (in other words, it must be ensured that $T_i \leq Q_i$).

1. This model, albeit simple, has been shown to provide accurate estimates of power consumption for different host types when running several benchmarks representative of real-world applications [18].

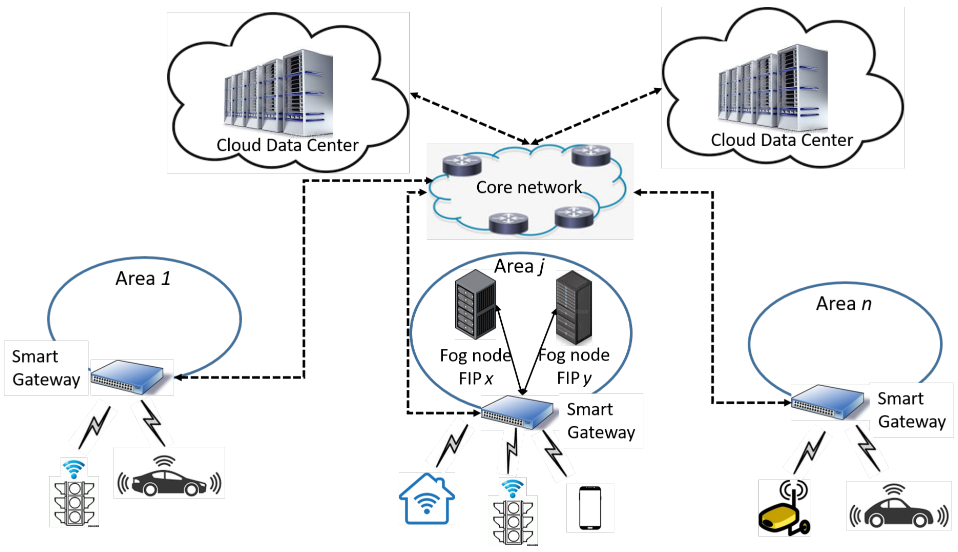

Figure 1. System architecture.

Each application S_i is associated with its *reference FIP* $Ref(S_i)$, i.e., the FIP that hosts the VMs running the instances of S_i. We assume that a given FIP k can be the reference FIP of several distinct applications, that we denote as $App(k)$.

An application S_i and the corresponding $Ref(S_i)$ are bound by a contractual obligation stating that the owner of S_i will pay to $Ref(S_i)$ a certain amount of money per each unit of time, computed according to an agreed-upon *revenue rate* $R_{k,i}$. In addition, $Ref(S_i)$ will pay the owner of S_i an amount of money per each unit of time during which the QoS target of S_i is not met, computed according to the agreed-upon *penalty rate* $L_{k,i}$.

2.1.3. The Virtual Machines.

The instances of any application S_j are embedded into a set of identical VMs, each one hosting a single instance, that are instantiated from a common master VM, denoted as VM_j.

VM_j is characterized by the amount of time τ_j it takes to process a single request that, without loss of generality, it is assumed to be determined only by the amount of physical CPU capacity allocated to that VM, [2] and that this amount is the same for all its instances, and remains constant for all their lifetimes.

To ensure that all the instances of VM_j exhibit the same value of τ_j, we assume that each one of them receives, on the fog node k on which it runs, a suitable amount of CPU capacity $U_{k,j}$ determined as follows.

First, a profiling experiment is carried out by running VM_j on a reference fog node x (e.g., using the methodology described in [21], [22], [23]) in which the amount of physical capacity allocated to VM_j is progressively increased until its measured processing time reaches the value τ_j, and the corresponding value $U_{x,j}$ of allocated physical CPU capacity is recorded. Then, the amount of physical capacity $U_{k,j}$ that must be allocates to VM_j on fog node $k \neq x$ to obtain a response time τ_j is computed as [20]:

$$U_{k,j} = U_{x,j} \frac{c_x}{c_k} \tag{2}$$

where c_k and c_x denote the physical CPU capacity of fog node k and x, respectively, that are measured as discussed in Section 2.1.1.

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

In order to meet the QoS target of S_i in area j, it is necessary to suitably choose the number $N_{i,j}$ of VMs allocated on fog nodes located in that area so as to ensure that $T_i \leq Q_i$. This value, however, depends on the value of the load intensity $\lambda_{i,j}(t)$, which is not constant but varies, as already discussed, according to the load profile $l(i,j)$.

Similarly what has been done for cellular networks in [24] to determine $\lambda_{i,j}(t)$, we proceed as follows: we discretize $l(i,j)$ by splitting the time axis into uniform disjoint sub-intervals $[r, r+\Delta t)$ of length Δt time units. Then, the value $\lambda_{i,j}(s)$ for any sub-interval s is approximated as a constant value set to the peak load in that sub-interval. Once the values of $\lambda_{i,j}(s)$ have been computed as above, they are fed as input into a queuing model representing the set of VMs of S_i allocated in area j. The solution of this model yields the minimum number $N_{i,j}(s)$ of VMs in sub-interval s that results in the satisfaction of the inequality $T_i \leq Q_i$.

In particular, the set of VMs associated to application S_i in area j is modeled as an M/M/c-FCFS queuing station [25] with $c = N_{i,j}(s)$, given that the service times of all the instances of VM_i are identical and the incoming stream of processing requests is fairly distributed among them.

For these queuing systems, it can be shown that in any time interval – and in particular in each sub-interval s – the average response time T_i is given by Eq. (3) (where,

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 [20]).

for readability purposes, we drop the dependence on s):

$$T_i = \frac{\bar{n}}{\lambda_{i,j}} = \frac{C}{\mu N_{i,j} - \lambda_{i,j}} + \frac{1}{\mu} \tag{3}$$

where:

- $\mu = \frac{1}{\tau_i}$ is the request service rate;
- $\bar{n} = \frac{\rho}{1-\rho} C + \rho N_{i,j}$ is the average number of requests in the station, both in the queue and receiving service, and $\rho = \frac{\lambda_{i,j}}{\mu N_{i,j}}$ is the offered load to the station;
- $C = [1 + (1-\rho)(\frac{N_{i,j}!}{(\rho N_{i,j})^{N_{i,j}}}) \sum_{k=0}^{N_{i,j}-1} \frac{(N_{i,j}\rho)^k}{k!}]^{-1}$ is the probability of a request to be enqueued before being served.

From Eq. (3) it follows that, in order to have $T_i \leq Q_i$, $N_{i,j}$ must satisfy the following inequality:

$$N_{i,j} \geq \frac{C}{Q_i - \frac{1}{\mu}} + \frac{\lambda_{i,j}}{\mu} \tag{4}$$

Furthermore, to ensure the stability of the system, we must have that:

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

2.2. Problem Statement

Let us now describe the problem that is faced by a FIP i that has to allocate, for each application $S_k \in App(i)$, the corresponding set of VMs on its fog nodes located in area j (denoted as $FN(i,j)$. Without loss of generality, we focus on a single area, given that FIPs allocate their respective VMs submitted in a given area independently from those submitted to other areas. The extension to other areas is thus straightforward.

FIP i aims at getting a *net profit* (i.e., the difference between its revenues and costs) as high as possible, given the request for the allocation of all the applications $S_k \in App(i)$.

The net profit rate $P_{i,j}$ (i.e., the profit it makes per unit of time) is computed as the following difference:

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

where $n_{k,j} \leq N_{k,j}$ is the number of VMs for S_k that are actually allocated (see below), $E_{i,j}$ is the energy cost for FIP i in area j (expressed as a cost rate per unit of time), u_f is the overall physical capacity of fog node $FN(i,j)$ allocated to the VMs it hosts, and $\mathbf{1}_\Omega(x)$ is the *indicator function* which has value 1 if $x \in \Omega$ and 0 otherwise.

(6) has the following meaning:

- the first term of the difference is the sum of the revenue rates $R_{i,k}$ that FIP i charges (per unit of time) to each application S_k for hosting $n_{k,j}$ of its VMs;
- the second term of the difference represents the costs that FIP i incurs (per unit of time) to run the above VMs. This cost, in turn, is given by the sum of two costs, namely (1) the *energy cost rates* resulting from the execution of overall CPU capacity allocated to all the VMs it hosts (see Eq. 1), and (2) the possible *monetary penalty rates* $L_{i,k}$ that FIP i incurs when the QoS of some application $S_k \in App(i)$ is not met (i.e., when $n_{k,j} < N_{k,j}$).

Maximizing the net profit rate is a challenging task which involves solving an optimization problem that takes into account the current workload, the electricity price and the application penalties. In Section 3.2, we provide more details about the solution to this optimization problem.

Intuitively, when the number of VMs to allocate on a fog node $FN(i,j)$ is so small that the resulting net profit is negative, FIP i must decide whether it is more profitable to not allocate any VM on $FN(i,j)$ (thus opting to pay the monetary penalties for violating the QoS of the related applications), or it is instead better to allocate the VMs anyways to avoid paying high application penalties. Also, when the number of VMs is so large that it needs more than one fog node to allocate them, FIP i must decide whether it is more profitable to allocate all of them, or it is instead better to allocate only the ones that leads to a positive profit (thus paying the monetary penalties for those applications whose QoS is not met).

In this paper, we show that a way to improve the net profit of a FIP is through cooperation, meaning that two or more FIPs in the same area of interest join to form a coalition where they share their workloads and their fog nodes to serve them.

Specifically, with cooperation, FIP i can try to reduce her/his energy consumption costs by allocating (some of) its VMs to the fog nodes of other FIPs, so that its fog nodes can be turned off. Also, FIP i can try to increase its revenues either by hosting VMs from other FIPs (so that it can better 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 related applications).

Clearly, FIPs are willing to cooperate with each other only if they receive suitable 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 her/his current coalition to move to a more profitable one or to work alone.

In Section 3, we better formalize this cooperation process in the framework of the game theory and we propose a decentralized algorithm to form stable coalitions among a group of FIPs, so that no FIP in the same coalition has incentive to leave her/his current coalition to join a better one.

3. The coalition formation game

We assume that the agents (also called players) may join/leave a coalition without any permission requirements, that is, a player is always accepted by a coalition to which it is willing to join, and it can leave a coalition without any permission. One way to describe such a process is to model it as a *coalition formation game*.

In particular, in this paper we use a type of coalition formation games, known as *hedonic coalition formation games* (also called hedonic games) [13].

An 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.*

The hedonic games are usually analyzed in terms of the stability *of coalition structures: the focus lies on finding the conditions for the existence of stable outcomes (i.e., a coalition or a set of coalitions). An outcome (e.g., a set of coalitions) 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 allows to guarantee stability against single player moves, i.e. Nash stability, while other and also allows group movements, i.e.the core (see [12], [13] for details).*

Lack of stability causes possible monetary losses for the following reasons: *i)* a player (in our case a FP) 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.

A coalition $\mathcal{C} \subseteq \mathcal{N}$ represents an agreement among the players in $\mathcal{C}$ to act as a single entity (i.e., 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, that we define 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 $\bigcup_{k=1}^{l} \mathcal{C}_k = \mathcal{N}$ and $\mathcal{C}_j \cap \mathcal{C}_k = \emptyset$ for $j \neq k$. Given a coalition partition Π, for any player $i \in \mathcal{N}$, we denote by $\mathcal{C}_\Pi(i)$ the coalition $\mathcal{C}_k \in \Pi$ such that $i \in \mathcal{C}_k$.

An hedonic coalition formation game is a pair $(\mathcal{N}, \succeq)$, where $\mathcal{N}$ is the set of players, and $\succeq_i$ a preference profile that specifies, for every player $i \in \mathcal{N}$, a reflexive, complete, and transitive binary relation $\succeq_i$ on $\mathcal{N}_i$ (set of all coalitions that include player i). The binary relation $\succeq_i$ is called *preference relation.*

In its partition form, a coalition game is defined on the set $\mathcal{N}$ by associating a utility value $u(\mathcal{C}|\Pi)$ to each subset of any partition Π of $\mathcal{N}$. For hedonic games the utility value of a coalition $\mathcal{C}$ is independent of the other coalitions, that is, and therefore, $u(\mathcal{C}|\Pi) = u(\mathcal{C})$. 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 3.2.

To set up the coalition formation process, we need to specify the preference relation so that each player can order and compare all the possible coalitions it belongs to, and hence it can build preferences over them. This can be done by using a preference function $\pi_i(\mathcal{C})$ that describes the preference of player i (with $i \in \mathcal{N}$) for any coalition $\mathcal{C} \in 2^{\mathcal{N}}$. Formally the preference function for player i can be defined as $\pi_i : 2^{\mathcal{N}} \longrightarrow \mathbb{R}$. In this manner we can say that a player i prefers the coalition $\mathcal{C}$ to $\mathcal{T}$ iff,

$$\pi_i(\mathcal{C}) \geq \pi_i(\mathcal{T}) \Longleftrightarrow \mathcal{C} \succeq_i \mathcal{T}. \tag{7}$$

We assume that the preference relation is chosen to be equal to the utility allocated to the player in a coalition. In other words, we have that $\pi_i(\mathcal{C}) = \phi_i(\mathcal{C})$, where $\phi_i(\mathcal{C})$ is the utility received by player i in coalition $\mathcal{C}$.

An allocation is said to be *efficient* if for any coalition $\mathcal{C} \in \Pi$, the sum of the individual utilities allocated to the coalition participants is equal to the coalition utility, that is

$$\sum_{i \in \mathcal{C}} \phi_i(\mathcal{C}) = u(\mathcal{C}). \tag{8}$$

The key point in hedonic games concerns the stability of this kind of games. The literature on hedonic games leads to different definition of stability concepts (see [13] for details). In this paper we focus the attention on the *Nash-stability* because we are interested in those games where the players only perform individual moves. In other words, we assume that moves of groups of players are not allowed. A partition Π is said to be Nash-stable if no player can benefit from moving from his coalition $\mathcal{C}_\Pi(i)$ to another existing coalition $\mathcal{C}_k$, that is,

$$\forall i, k : i \in \mathcal{C}_\Pi(i) \succeq_i \mathcal{C}_k \cup \{i\}, \text{where } \mathcal{C}_k \in \Pi \cup \{\emptyset\}. \tag{9}$$

Note that, as pointed out in [13], the Nash-stability is a noncooperative notion of stability in the sense that players do not need permission to leave/join a coalition.

Identifying conditions (sufficient and/or necessary) for the existence of stability in hedonic games has been active area of research. In particular, [13] shows that *symmetric additively separable* preferences guarantee the existence of a Nash-stable partition. *In this class of hedonic game each player, in order to express its preference over the set of possible coalitions, assigns a value to any other player. The value of a coalition, therefore, is simply the sum of the values he assigns to the members of his coalition.* A player is preferences are *additively separable* if there exists a function $v : \mathcal{N} \times \mathcal{N} \longrightarrow \mathbb{R}$ such that $\forall \mathcal{C}_1, \mathcal{C}_2$ such that $i \in \mathcal{C}_1$ and $i \in \mathcal{C}_2$,

$$\mathcal{C}_1 \succeq_i \mathcal{C}_2 \Longleftrightarrow \sum_{j \in \mathcal{C}_1} v(i, j) \geq \sum_{j \in \mathcal{C}_2} v(i, j), \tag{10}$$

where $v(i, i) = 0$. Additively separable preferences are *symmetric* if $v(i, j) = v(j, i)$, for every $i, j \in \mathcal{N}$.

In [26] a variation of stability that accounts for the Nash-stability (as defined by Equation (9)), and for the notion of *efficiency*. In that paper the set of efficient allocations of a Nash-stable coalition partition has been called *Nash-stable core*.[3]

In [26] the Nash-stability has been rephrased to an optimization problem aiming at deriving the functions $v(\cdot, \cdot)$ (for any player $i \in \mathcal{N}$) satisfying the property defined by Equation (10). In other words, if it is possible to derive the functions $v(\cdot, \cdot)$ satisfying the property defined

3. Although, the rationale behind the name 'Nash-stable core' reminds to the idea of finding all the efficient allocation methods in a Nash-stable coalition, the word *core* might be confuse with the completely different concept of *core stability*.

by Equation (10) then the preference relations $\succeq_i$ satisfy the satisfying symmetric additively separable property, and hence there exists a Nash-stable coalition partition.

Since symmetric additively separable is a strong condition, it is not surprising that the recasting of the stability problem into an optimization problem does not provide any dramatic improvement to the class of hedonic games satisfying the Nash-stability condition. However, the rephrasing has been used to go beyond the limits of the symmetric additively separable property. In [26] the optimization problem has been relaxed and this allows the definition of a Nash-stability condition that overcomes all the limitations of the symmetric additively separable property. The price to pay for the relaxed version of the Nash-stability is the efficiency. *That is, the method does not guarantee that coalition's utility is totally allocated.*

3.1. The Coalition Formation Algorithm

In this section, we now develop a decentralized algorithm to reach a Nash-stable partition. We use the results presented in [26] where the problem of finding a Nash-stable partition in a hedonic coalition formation has been formulated as a non-cooperative game. According to the discussion presented in [26] the hedonic coalition formation problem can be seen as a weakly acyclic game [27], [28]. That is, the game evolves in turns and players are allowed to choose their strategies once in each turn. In [29] is showed that such games always converge to a pure-strategy Nash equilibrium, meaning that at some point the Nash stable equilibrium will be played. Once such strategy profile has been played it will be played forever. However, randomizing players' strategies is one possible solution to weekly acyclic games; a more suitable approach for an algorithmic solution is described in [27]. Here players are forced to make their decision subsequently, selected in each turn according to a specific order. Specifically by using a random scheduler, the sequence of players choices (each of which represents the player's best response) reaches an equilibrium if there exists at least one.

To describe the decentralized coalition formation algorithm we denote a scheduler by $\Sigma = \{\mathrm{s}(1), \mathrm{s}(2), \ldots, \mathrm{s}(N)\}$. A scheduler is a random permutation of players' indices and, in particular, the scheduler of l-th round denoted by $\mathrm{s}(l)$ is a tuple $\mathrm{s}(l) = \{s_1(l), s_2(l), \ldots, s_N(l)\}$, where $s_i(l)$ identifies the i-th player selected to play in the round l. Hence, each round includes N *steps* that correspond to single players' decisions. It is important to notice that in this game, the set of actions available to each player is equal to the number of coalitions in the game in that moment. For example, in the case of all players apart and considering singletons as coalitions a player may join any other agent or not to move.

A strategy tuple in step s is denoted as $\sigma^{(s)} = \{\sigma_1^{(s)}, \sigma_2^{(s)}, \ldots, \sigma_N^{(s)}\}$, where $\sigma_i^{(s)}$ is the strategy of player i in step s. The strategy tuples $\sigma^{(s)}$ and $\sigma^{(s-1)}$ differ in at most one position. That is, the position corresponding to the player that takes its turn in step s. On the other hand $\sigma^{(s)}$ and $\sigma^{(s-1)}$ are identical if the player that takes its turn in step s does not change its previous strategy. We denote by $\Pi_l^{(s)}$ the partition in step s of the l-th round, and by Π_l the partition obtained at the end of the l-th round.

Furthermore, we denote by $\mathcal{C}_i^{(s)} = \{j : \sigma_j^{(s)} = \sigma_i^{(s)}, \forall j \in \mathcal{N}\}$ the set of players that share the same strategy with player i. In this manner we have that preference relation of player i, denoted by $\pi_i(\mathcal{C})$, verifies the following relation

$$\pi_i(\mathcal{C}_i^{(s)}) > \pi_i(\mathcal{C}_i^{(s-1)}) \Longleftrightarrow \mathcal{C}_i^{(s)} \succ_i \mathcal{C}_i^{(s-1)}.$$

It is easy to verify that a Nash-stable partition is reached, at a given $l+1$-th round, when $\Pi_l = \Pi_{l+1}$.

A three players example (cont'd). The algorithm starts from the $\Pi_0 = \{(1), (2), (3)\}$, that is the partition where each player is alone. This corresponds to the strategy tuple $\{\mathrm{s}_1, \mathrm{s}_2, \mathrm{s}_3\}$ (each player chooses a different strategy). Table 1 shows two possible evolutions of the example in the previous section.

Round / Schedule	Step	Player	Strategy tuple	Partition
0			$\{\mathrm{s}_1, \mathrm{s}_2, \mathrm{s}_3\}$	$\{(1), (2), (3)\}$
1 - (2,1,3)	1	2	$\{\mathrm{s}_1, \mathrm{s}_{1,2}, \mathrm{s}_3\}$	$\{(1, 2), 3)\}$
	2	1	$\{\mathrm{s}_{1,2}, \mathrm{s}_{1,2}, \mathrm{s}_3\}$	
	3	3	$\{\mathrm{s}_{1,2}, \mathrm{s}_{1,2}, \mathrm{s}_3\}$	
2 - (1,3,2)	1	1	$\{\mathrm{s}_{1,2}, \mathrm{s}_{1,2}, \mathrm{s}_3\}$	$\{(1, 2), 3\}$
	2	3	$\{\mathrm{s}_{1,2}, \mathrm{s}_{1,2}, \mathrm{s}_3\}$	
	3	2	$\{\mathrm{s}_{1,2}, \mathrm{s}_{1,2}, \mathrm{s}_3\}$	
0			$\{\mathrm{s}_1, \mathrm{s}_2, \mathrm{s}_3\}$	$\{(1), (2), (3)\}$
1 - (3,1,2)	1	3	$\{\mathrm{s}_1, \mathrm{s}_2, \mathrm{s}_{2,3}\}$	$\{1, (2, 3)\}$
	2	1	$\{\mathrm{s}_{1,2,3}, \mathrm{s}_2, \mathrm{s}_{2,3}\}$	$\{(1, 2, 3)\}$
	3	2	$\{\mathrm{s}_{1,2,3}, \mathrm{s}_{1,2,3}, \mathrm{s}_{2,3}\}$	
2 - (2,3,1)	1	2	$\{\mathrm{s}_{1,2,3}, \mathrm{s}_{1,2,3}, \mathrm{s}_{2,3}\}$	$\{(1, 2), 3\}$
	2	3	$\{\mathrm{s}_{1,2,3}, \mathrm{s}_{1,2,3}, \mathrm{s}_{1,2,3}\}$	
	3	1	$\{\mathrm{s}_{1,2,3}, \mathrm{s}_{1,2,3}, \mathrm{s}_{1,2,3}\}$	

TABLE 1. TWO SAMPLE EXECUTIONS OF THE ALGORITHM

In the first example reported in Table 1, the extracted scheduling for the first round is $(2, 1, 3)$. Hence, according to its preference profile, Player 2 forms a coalition with Player 1 choosing strategy $\mathrm{s}_{1,2}$. Then is Player 1 turn, and choosing to coalesce with Player 2 is its *best response* strategy. Therefore after the second step in turn one the only coalition formed is $\{1, 2\}$ and player 3 apart. In turn 3 Player 3 has no any available options other than staying alone, given that join the grand coalition does not improve its payoff. After that, the first run is finished and a new scheduling for the upcoming round is extracted. However, from the current partition structure does not matter the order in which players are scheduled. That is, strategy profile $\{\mathrm{s}_{1,2}, \mathrm{s}_{1,2}, \mathrm{s}_3\}$ is a Nash equilibrium and players have no incentive to deviate from that.

Again, in the second example, the game begins with all players apart and the extracted scheduler is $(1, 3, 2)$. Player 3 begins and coalesces with Player 2. Player 1 is the next to play, and the only move available to him better than playing alone is to join the grand coalition. Therefore, at this point of the game, the formed coalition gathers all the players. For Player 2, the coalition $\{1, 2, 3\}$ is the most preferable and it is happy to be part of it. After Player 2 moves the round is over and the second round begins with scheduling $(2, 3, 1)$. Now is Player 1 turn and it faces the same choice as before; since at the previous step Player 1 chose the best it could and now the situation is unchanged, it will choose again to stay in the grand coalition. Eventually is Player 3 turn which does not get any benefit from leaving the grand coalition an so it will remain there. Given that there is no any player who benefits from unilaterally deviate from the strategy profile $\{s_{1,2,3}, s_{1,2,3}, s_{1,2,3}\}$ a Nash-stable equilibrium is reached.

3.2. Computation of the Optimal Coalition Allocation Profit

The coalition formation process described in the above sections requires the computation of the coalition net profit rate $u(\mathcal{C})$ for any coalition $\mathcal{C}$ of FIPs that may form.

As discussed in Section 2.2, this computation involves solving an optimization problem that, given a geographic area h and a coalition $\mathcal{C}$ of FIPs located in this area, seeks to find the optimal allocation of the set $\mathcal{V}$ of VMs where to run instances of the applications $\mathcal{A} = \bigcup_{i \in \mathcal{C}} App(i)$ to host in this area onto the set $\mathcal{F} = \bigcup_{i \in \mathcal{C}} FN(i,h)$ of fog nodes, so as to maximize the overall net profit rate of coalition $\mathcal{C}$. The set $\mathcal{V}$ is given by the union of the $N_{j,h}(t)$ VMs required by each application $j \in \mathcal{A}$ to meet its target QoS in the discretization interval t.

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

The resulting optimization model is shown in Figure 2, where we use the same notation defined in Section 2 and where we denote as $p(\cdot)$ the function $p : \mathcal{F} \to \mathcal{C}$ which maps, in the given area of interest h, a fog node to its FIP, and as $s(\cdot)$ the function $s : \mathcal{V} \to \mathcal{A}$ which maps a VM to the (instance of) application it runs. Also, to ease readability, we simplify the model by dropping from it the dependence from the discretization interval t (e.g., we denote as $N_{j,h}$, instead of $N_{j,h}(t)$, the required number of VMs to allocate in order to meet the QoS of application j).

In the optimization model, we define the following decision variables:

- x_i: a binary decision variable which is equal to 1 if fog node i is powered on, and 0 otherwise;
- $y_{i,j}$: a binary decision variable which is equal to 1 if VM j is allocated on fog node i, and 0 otherwise;
- u_i: a non-negative real decision variable which represents the total fraction of CPU capacity of fog node i that has been allocated to the VMs it hosts;
- n_k: a non-negative integer decision variable which denotes the number of VMs allocated to run instances of application k.

The objective function of the optimization model represents the overall net profit rate earned by the coalition $\mathcal{C}$ of FIPs, which is defined as the difference between the revenues obtained by the allocation of VMs, and the costs due both to the electricity power absorbed by the powered-on fog nodes and to QoS violations (if any).

The maximization of this objective function is bound to the following constraints:

- Eq. (11b) assures that no VM is allocated on a fog node that will be powered off;
- Eq. (11c) states that each VM is hosted by no more than one fog node;

$$
\begin{aligned}
\text{maximize } u(\mathcal{C}) = & \sum_{k \in \mathcal{A}} R_{Ref(k),k} n_{k,h} \\
& - \Bigg[\sum_{i \in F} \big(x_i W_i^{\min} + (W_i^{\max} - W_i^{\min}) u_i \big) E_{p(i),h} \\
& + \sum_{k \in \mathcal{A}} \big(n_{k,h} < N_{k,h} \big) L_{Ref(k),k} \Bigg] \qquad \text{(11a)}
\end{aligned}
$$

subject to

$$
\begin{aligned}
& \sum_{j \in \mathcal{V}} y_{i,j} \le |\mathcal{V}| x_i, \quad \forall i \in F, && \text{(11b)} \\
& \sum_{i \in \mathcal{F}} y_{i,j} \le 1, \quad \forall j \in \mathcal{V}, && \text{(11c)} \\
& u_i = \sum_{j \in \mathcal{V}} y_{i,j} U_{i,j}, \quad \forall i \in \mathcal{F}, && \text{(11d)} \\
& u_i \le x_i, \quad \forall i \in \mathcal{F}, && \text{(11e)} \\
& n_k = \sum_{i \in \mathcal{F}} \sum_{\substack{j \in \mathcal{V}, \\ s(j)=k}} y_{i,j}, \quad \forall k \in \mathcal{S}, && \text{(11f)} \\
& n_k \le N_{k,h}, \quad \forall k \in \mathcal{A}, && \text{(11g)} \\
& x_i \in \{0,1\}, \quad \forall i \in \mathcal{F}, && \text{(11h)} \\
& y_{i,j} \in \{0,1\}, \quad \forall i \in \mathcal{F}, j \in \mathcal{V}, && \text{(11i)} \\
& u_i \in \mathbb{R}^*, \quad \forall i \in \mathcal{F}, && \text{(11j)} \\
& n_k \in \mathbb{N}, \quad \forall k \in \mathcal{A}. && \text{(11k)}
\end{aligned}
$$

Figure 2. The optimization model for the maximization of the coalition net profit rate.

- Eq. (11d) defines the value of the variable u_i as the sum of the CPU capacity requirements of the VMs allocated on fog node i;
- Eq. (11e) ensures that the allocated CPU capacity of a powered-on fog node is not exceeded;
- Eq. (11f) defines the value of the variable n_k as the number of allocated VMs where to run instances of application k;
- Eq. (11g) ensures that for each application no more VMs are allocated than needed;
- Eqs. 11h–11k define the domain of decision variables x_i, $y_{i,j}$, u_i and n_k, , respectively.

4. Experimental Evaluation

To assess the effectiveness of the proposed coalition formation algorithm in increasing the net profits for a population of FIPs, we perform an experimental evaluation in which we run our algorithm for various scenarios. In these scenarios, we vary the workload of each application and we assess the impact of it on the performance of the proposed algorithm.

The results we obtain from these experiments show the ability of our algorithm to form coalitions of FIPs that are stable and profitable in all the scenarios we consider.

In the rest of this section, we first provide the settings we use in our experimental scenarios (Section 4.1), and then we show the results we obtain by applying our proposed algorithm to these scenarios (Section 4.2).

TABLE 2. PARAMETERS USED IN THE EXPERIMENTAL SCENARIOS. SUBSCRIPTS i AND j TAKE VALUES ON THE SET $\{1, 2, 3\}$.

	Parameter	Value
$\lvert App(i)\rvert$	Number of applications associated to FIP i	1
$E_{i,j}$	Electricity price for FIP i in area j	0.0001 \$/Wh
$\lvert FN(i,j)\rvert$	Number of fog nodes for FIP i in area j	3
$L_{i,j}$	Penalty rate for FIP i and application j	0.022 \$/h
m	Number of FIPs	3
n	Number of applications	3
Q_i	Max request processing time for application i	0.7 sec
$R_{i,j}$	Revenue rate for FIP i and application j	0.0022 \$/h
$U_{i,j}$	CPU demand for any VM j and fog node i	0.05
$W_j^{\max}$	Max power consumption of fog node j	200 W
$W_j^{\min}$	Idle power consumption of fog node j	100 W
τ_j	Request processing time of any VM j	0.5 sec

4.1. Experimental Setup

In this section, we present the settings of the scenarios used in our experimental evaluation, which are summarized in Table 2 and Table 3.

In a given geographic area h, we consider $m = 3$ identical FIPs, each of which is in charge of running instances of a different application (i.e., $App(i) = \{S_i\}$, for $i = 1, \dots, 3$). The physical infrastructure of each FIP consists of 3 identical fog nodes whose idle and maximum power consumptions $W_j^{\min}$ and $W_j^{\max}$ are set to 100 W and 200 W, respectively (for $j = 1, \dots, 3$). This number of fog nodes ensures that, in the scenarios we consider, each FIP, when working in cooperation, is able to accept on her/his fog nodes all the workload of the other FIPs that are member of the same coalition. For each application j, its master VM VM_j is characterized by a request processing time τ_j of 0.5 sec and, to achieve this value, it requires a physical CPU capacity $U_{i,j}$ of 0.05 for every fog node i (i.e., each VM of application j consumes 5% of the physical CPU capacity of each fog node i).

We assume that the electricity price $E_{i,h}$ charged hourly to each FIP i is the same for all FIPs and we set it to 0.0001 \$/Wh [30]. Also, we set the revenue rate $R_{i,j}$ that each FIP i earns for hosting a VM j to 0.0022 \$/hour. We derived this value by assuming that each FIP reaches the break-even point when the load of one of her/his fog node (i.e., the total CPU capacity of the fog node allocated to VMs) is 30%, which is a value – that depends on the specific parameters we set for our experimental evaluation – that we have suitably chosen 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 to 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 (s)he cannot amortize).

Finally, for each application j, we set its QoS parameter Q_j to 0.7 sec and the related monetary penalty rate $L_{i,j}$ to 0.022 \$/hour for each FIP i, which is 10 times the revenue rate $R_{i,j}$. We choose this value for the penalty rate so that an FIP always prefer allocating VMs for the applications (s)he hosts than refuse them for reducing energy consumption costs.

TABLE 3. THE EXPERIMENTAL SCENARIOS CORRESPONDING TO DIFFERENT LOAD LEVELS. SUBSCRIPT k TAKES VALUES ON THE SET $\{1, 2, 3\}$.

Scenario	$\lambda_{k,h}$	$N_{k,h}$	α_k
Scenario #1	2.1	2	0.1
Scenario #2	5.7	4	0.2
Scenario #3	9.4	6	0.3
Scenario #4	13.3	8	0.4
Scenario #5	17.2	10	0.5
Scenario #6	21.1	12	0.6
Scenario #7	25.0	14	0.7
Scenario #8	28.9	16	0.8
Scenario #9	32.8	18	0.9

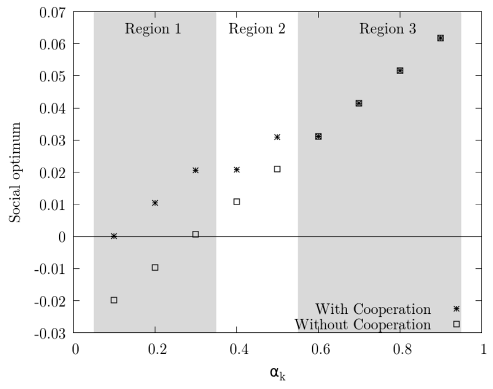

Figure 3. Comparison between the case where the providers work independently with the coalition formation algorithm.

Given the above settings, 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 so that the induced load α_i, experienced by each FIP i (when (s)he works alone) on the fog node where the $N_{h,i}$ 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 reported in Table 3, where the first column gives the scenario name, the second column is the load level of each application k stated in terms of its workload intensity $\lambda_{k,h}$ in the area of interest h, the third column contains the minimum number $N_{k,h}$ of VMs required to satisfy the QoS parameter Q_k of application S_k in face of the workload $\lambda_{k,h}$, and the last column reports the load α_i induced on a fog node of FIP i by $\lambda_{k,h}$.

To run our experiments, we develop an ad hoc simulator in C++ where we use the IBM ILOG CPLEX solver 12.7.1 [31] for solving the optimization problem discussed in Section 3.2.

4.2. Experimental Results

Figure 3 summarizes the behaviour of the coalition formation algorithm for different values of the load (or of the request arrival rates). In particular, we compare the social welfare values (i.e., the sum of the provider's utility) in case that the providers work independently (symbol

$*$ in the figure) vs the case where the providers set up coalition among them if this increases their profits (symbol $\square$ in the figure). Despite the discrete granularity we can identify three regions in the figure that correspond to different coalition formation algorithm outcomes. Region 1 characterizes the low load values. In these cases the coalition formation algorithm yields the grand coalition, and the coalition always increases social welfare.

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 behavior is due to the reduction of waste of resources that occurs in case of under utilized FIPs.

Despite of its simplicity, the set of experiments summarized by Figure 3, illustrates how to use the distributed coalition formation algorithm. The fog providers, 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.

5. Related Works

The game theoretical approach to coalition formation has been used in many other scenarios. In the field of cloud computing there is a large body of research, see, for instance [20], [32], [33], [34]. Similar approaches have been used to study cooperative behavior in cellular networks [24], [35], [36]. Coalition formation frameworks for femtocell networks has been used for different purposes such as mitigating the interference, and resource and power allocation, examples of such proposals can be found in [37], [38], [39].

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). In these scenarios the emphasis is on optimizing resources [40], [41], optimization of latency and/or the energy consumption [42], [43], offloading strategies [44], [45], and approaches tailored for macro/micro cellular network scenarios [46], [47].

In this paper 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.

6. Conclusions

In this paper we have dealt with the problem of making a set of FIPs increase their profits when allocating their resources to process the data generated by IoT applications that need to meet specific QoS targets in face of time-varying workloads.

To this end, we proposed a cooperative game-theoretic framework to study the federation formation problem, and a mathematical optimization model to allocate IoT applications of the resources of a FIP in order to increase its net profit.

In the proposed scheme, we model the cooperation among FIPs as a coalition game with transferable utility and we devise a distributed algorithm for coalition formation. 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. Furthermore, we show that the proposed algorithm converges to a Nash-stable partition which determines the resulting coalition structure. Numerical results exhibit the effectiveness of our approach.

The future developments of this research is following several directions. In particular, we would like to enhance the coalition value function in order to account for possible request losses due to lack of physical resources. Furthermore, we want to improve the game-theoretic and optimization models in order to include costs in terms of loss of revenues as well as other aspects like the ones related to trustworthiness among FIPs. Finally, we want to implement and validate the proposed algorithm on a real testbed.

Acknowledgments

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

References

[1] M. Hassanalieragh, A. Page, T. Soyata, G. Sharma, M. Aktas, G. Mateos, B. Kantarci, and S. Andreescu, "Health monitoring and management using internet-of-things (iot) sensing with cloud-based processing: Opportunities and challenges," in *Services Computing (SCC), 2015 IEEE International Conference on*. IEEE, 2015, pp. 285–292.

[2] C. Perera, Y. Qin, J. C. Estrella, S. Reiff-Marganiec, and A. V. Vasilakos, "Fog Computing for Sustainable Smart Cities: A Survey," *ACM Comput. Surv.*, vol. 50, no. 3, pp. 32:1–32:43, October 2017.

[3] D. Vasisht, Z. Kapetanovic, J. Won, X. Jin, R. Chandra, S. N. Sinha, A. Kapoor, M. Sudarshan, and S. Stratman, "Farmbeats: An iot platform for data-driven agriculture." in *NSDI*, 2017, pp. 515–529.

[4] C. C. Byers, "Architectural imperatives for fog computing: Use cases, requirements, and architectural techniques for fog-enabled iot networks," *IEEE Communications Magazine*, vol. 55, no. 8, pp. 14–20, 2017.

[5] F. Bonomi, R. Milito, P. Natarajan, and J. Zhu, *Fog Computing: A Platform for Internet of Things and Analytics*. Cham: Springer International Publishing, 2014, pp. 169–186.

[6] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proceedings of the 1^{st} Edition of the MCC Workshop on Mobile Cloud Computing*, ser. MCC '12. Helsinki, Finland: ACM, 2012, pp. 13–16.

[7] P. Hu, S. Dhelim, H. Ning, and T. Qiu, "Survey on fog computing: architecture, key technologies, applications and open issues," *Journal of Network and Computer Applications*, vol. 98, no. Supplement C, pp. 27 – 42, 2017.

[8] Y. Liu, J. E. Fieldsend, and G. Min, "A framework of fog computing: Architecture, challenges, and optimization," *IEEE Access*, vol. 5, 2017.

[9] J. Weinman, "The Economics of the Hybrid Multicloud Fog," *IEEE Cloud Computing*, vol. 4, no. 1, pp. 16–21, Jan 2017.

[10] 451 Research, "Customer Insight: Future-proofing your colocation business."

[11] B. Peleg and P. Sudhölter, *Introduction to the Theory of Cooperative Games*, 2nd ed. Springer Berlin Heidelberg, 2007.

[12] J. Drèze and J. Greenberg, "Hedonic Coalitions: Optimality and Stability," *Econometrica*, vol. 48, no. 4, pp. 987–1003, 1980.

[13] A. Bogomolnaia and M. Jackson, "The Stability of Hedonic Coalition Structures," *Games Economic Behavior*, vol. 38, pp. 201–230, 2002.

[14] 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*, ser. Mobidata '15. New York, NY, USA: ACM, 2015, pp. 37–42.

[15] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The Case for VM-Based Cloudlets in Mobile Computing," *IEEE Pervasive Computing*, vol. 8, no. 4, pp. 14–23, Oct 2009.

[16] M. Aazam and E. N. Huh, "Fog Computing Micro Datacenter Based Dynamic Resource Estimation and Pricing Model for IoT," in *2015 IEEE 29th International Conference on Advanced Information Networking and Applications*, March 2015, pp. 687–694.

[17] Primate Labs, Inc., "GeekBench: Next-generation processor benchmark," 2018, accessed: Feb 2, 2018. [Online]. Available: http://www.geekbench.com

[18] S. Rivoire, P. Ranganathan, and C. Kozyrakis, "A comparison of high-level full-system power models," in *Proceedings of the 2008 Conference on Power Aware Computing and Systems*, ser. HotPower'08. Berkeley, CA, USA: USENIX Association, 2008, pp. 3–3.

[19] M. Aazam and E. N. Huh, "Fog computing and smart gateway based communication for cloud of things," in *2014 International Conference on Future Internet of Things and Cloud*, Aug 2014, pp. 464–470.

[20] C. Anglano, M. Guazzone, and M. Sereno, "A Game-Theoretic Approach to Coalition Formation in Green Cloud Federations," in *Proc. of the 2014 14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*, May 2014, pp. 618–625.

[21] C. Anglano, M. Canonico, and M. Guazzone, "FCMS: A fuzzy controller for CPU and memory consolidation under SLA constraints," *Concurrency and Computation: Practice and Experience*, vol. 29, no. 5, 2017.

[22] ——, "FC2Q: exploiting fuzzy control in server consolidation for cloud applications with SLA constraints," *Concurrency and Computation: Practice and Experience*, vol. 27, no. 17, pp. 4491–4514, 2015.

[23] ——, "Prometheus: A flexible toolkit for the experimentation with virtualized infrastructures," *Concurrency and Computation: Practice and Experience*, published online.

[24] C. Anglano, M. Guazzone, and M. Sereno, "Maximizing profit in green cellular networks through collaborative games," *Computer Networks*, vol. 75, Part A, pp. 260 – 275, 2014.

[25] G. Bolch, S. Greiner, H. de Meer, and K. S. Trivedi, *Queueing Networks and Markov Chains: Modeling and Performance Evaluation With Computer Science Applications, Second Edition*. Wiley, 2006.

[26] C. Hasan, E. Altman, and J. Gorce, "On the nash stability in the hedonic coalition formation games," *CoRR*, vol. abs/1405.3360, 2014.

[27] I. Milchtaich, "Congestion Games with Player-Specific Payoff Functions," *Games and Economic Behavior*, vol. 13, no. 1, pp. 111–124, 1996.

[28] K. R. Apt and S. Simon, "A classification of weakly acyclic games," in *Proc. of Algorithmic Game Theory: 5th International Symposium, SAGT 2012*. Springer, 2012, pp. 1–12.

[29] H. P. Young, "The evolution of conventions," *Econometrica: Journal of the Econometric Society*, pp. 57–84, 1993.

[30] U.S. Energy Information Administration (EIA), "Average price of electricity to ultimate customers," 2018, accessed: Feb 2, 2018. [Online]. Available: https://www.eia.gov/electricity

[31] IBM, "ILOG CPLEX Optimization Studio," 2018, accessed: Feb 2, 2018. [Online]. Available: https://www.ibm.com/products/ilog-cplex-optimization-studio

[32] R. Kaewpuang, D. Niyato, P. Wang, and E. Hossain, "A Framework for Cooperative Resource Management in Mobile Cloud Computing," *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 12, pp. 2685–2700, 2013.

[33] C. A. Lee, "Cloud federation management and beyond: Requirements, relevant standards, and gaps," *IEEE Cloud Computing*, vol. 3, no. 1, pp. 42–49, 2016.

[34] L. Mashayekhy, M. M. Nejad, and D. Grosu, "Cloud Federations in the Sky: Formation Game and Mechanism," *IEEE Transactions on Cloud Computing*, vol. 3, no. 1, pp. 14–27, 2015.

[35] M. Sereno, "Cooperative game theory framework for energy efficient policies in wireless networks," in *Third International Conference on Future Energy Systems (e-Energy 2012)*, 2012, pp. 1–9.

[36] C. Hasan, E. Altman, and J. M. Gorce, "The coalitional switch-off game of service providers," in *IEEE 9th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, 2013, pp. 223–230.

[37] F. Pantisano, M. Bennis, W. Saad, and M. Debbah, "Spectrum leasing as an incentive towards uplink macrocell and femtocell cooperation," *IEEE Journal on Selected Areas in Communications*, vol. 30, no. 3, pp. 617–630, 2012.

[38] Z. Zhang, L. Song, Z. Han, and W. Saad, "Coalitional games with overlapping coalitions for interference management in small cell networks," *IEEE Trans. Wireless Commun.*, vol. 13, no. 5, pp. 2659–2669, 2014.

[39] F. Pantisano, M. Bennis, W. Saad, M. Debbah, and M. Latvaaho, "Interference alignment for cooperative femtocell networks: a game-theoretic approach," *IEEE Trans. Mobile Comput.*, vol. 12, no. 11, pp. 2233–2246, 2013.

[40] S. Barbarossa, S. Sardellitti, and P. Di Lorenzo, "Joint allocation of computation and communication resources in multiuser mobile cloud computing," in *IEEE 14th Workshop on Signal Processing Advances in Wireless Communications*, 2013, pp. 26–30.

[41] S. Meng, Y. Wang, Z. Miao, and K. Sun, "Joint optimization of wireless bandwidth and computing resource in cloudlet-based mobile cloud computing environment," *Peer-to-Peer Networking and Applications*, Feb 2017.

[42] M.-H. Chen, M. Dong, and B. Liang, "Joint offloading decision and resource allocation for mobile cloud with computing access point," in *IEEE Int. Conf. Accoustic, Speech, and Signal Processing (ICASSP)*, 2016, pp. 3516–3520.

[43] T. Q. Dinh, J. Tang, Q. D. La, and T. Q. Quek, "Offloading in mobile edge computing: Task allocation and computational frequency scaling," *Commun. ACM*, no. 99, Apr 2017.

[44] S. Barbarossa, S. Sardellitti, and P. Di Lorenzo, "Computation offloading strategies based on energy minimization under computational rate constraints," in *23rd Europ. Conf. on Networks and Communications*, 2014, pp. 1–5.

[45] O. Munoz-Medina, A. Pascual-Iserte, and J. Vidal, "Optimization of radio and computational resources for energy efficiency in latency-constrained application offloading," *IEEE Trans. Veh. Technol.*, no. 99, 2014.

[46] J. Oueis, E. Calvanese-Strinati, and S. Barbarossa, "Small cell clustering for efficient distributed cloud computing," in *IEEE PIMRC*, 2014, pp. 1474–1479.

[47] S. Tanzil, O. Gharehshiran, and V. Krishnamurthy, "A distributed coalition game approach to femto-cloud formation," *IEEE Transactions on Cloud Computing*, vol. 99, 2016.