Future Generation Computer Systems 124 (2021) 21–32

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Stable federated fog formation: An evolutionary game theoretical approach

Ahmad Hammoud [a], Hadi Otrok [b], Azzam Mourad [c,*], Zbigniew Dziong [a]

[a] Department of Electrical Engineering, Ecole de Technologie Superieure (ETS), Montreal, Canada
[b] Department of EECS, Khalifa University, Abu Dhabi, United Arab Emirates
[c] Department of Computer science and Mathematics, Lebanese American University, Beirut, Lebanon

ARTICLE INFO

Article history:
Received 15 January 2021
Received in revised form 29 March 2021
Accepted 16 May 2021
Available online 24 May 2021

Keywords:
Evolutionary game theory
Federated fog
Fog computing
Stability
IoT

ABSTRACT

Instability within fog federations is considered as a serious problem that degrades the performance of the provided services. The latter may affect the service availability due to fog providers withdrawing their resources. It may either lead to failures for some users invocations, or to an increase in the number of tasks inside the servers' processing-queue. Such a critical problem strips the fog paradigm from its main characteristic, the low latency factor. As far as we are aware, no work in the literature has addressed the problem of encountering unstable fog federations. Their main concerns were increasing the providers' payoff regardless of their behavior. To address the aforementioned limitation, this work studies the stability of the federations through modeling the problem as an evolutionary game-theoretical model. Moreover, it devises a decentralized algorithm that implants the Replicator Dynamics model within which expresses the evolutionary dynamics. Experiments are conducted using EUA Datasets to simulate our algorithm and to show that it leads to an evolutionarily stable strategy over time, which stabilizes the federations and improves the Quality-of-Service for the users.

1. Introduction

The current revolutionary period we are living in was science fiction a few decades ago. Technology redefined the way people interact with their surroundings. In particular, Internet-of-Things (IoT) applications have become a necessary part of our lifestyle towards smart cities [1]. From a smart light bulb, to a smart home and a driverless vehicle, IoT devices are everywhere to improve our quality of life [2,3]. A statistical study that was recently published by Statista[1] depicts that the number of IoT devices will increase from 15.41 billion in the year 2015 to 75.44 billion devices in 2025. Such an increase encourages investors and stakeholders for investing more in the computational resources to satisfy the huge demand required by IoT devices.

In parallel, cloud providers cannot meet the Quality-of-Service (QoS) requested by the IoT applications due to the high latency between the devices and the cloud servers. Such network delays constitute a barrier for some applications such as health-care and autonomous driving where even small delays are costly [4,5]. To address this issue, Cisco[2] proposed a new concept called fog computing, which extends clouds to the edge of the network in order to massively reduce the network delays [6]. IoT devices can now request resources from available nearby fog nodes instead of communicating with the relatively far-away cloud servers. Nevertheless, fog servers entail high deployment costs leading to limitations in available resources compared to the clouds [7,8]. Hence, alternative solutions must be explored to satisfy the huge demand for resources by the Application Service Providers (ASPs). Many scholars recently addressed the resource limitation problem by trying to optimally schedule the tasks invoked by the IoT devices [9,10], whereas others considered overcoming such an issue through placing on-demand fog [11]. However, such alternatives are not feasible nor efficient when the fog provider, i.e. the party providing fog nodes, runs out of available resources in the geographical location having high demands for computing resources. Thus, federating fog providers would be considered as a convenient solution to overcoming all the aforementioned limitations.

Simultaneously, the concept of fog federations refers to various participants making use of their unallocated resources, instead of keeping them idle [12]. By reaching an agreement, the collaborators will be able to handle more tasks than anyone can handle on its own [13,14]. The advantages of such collaboration are twofold. On one hand, fog federations allow offloading tasks among servers (i.e., fog nodes) belonging to different fog providers for the sake of processing the user request as quickly as possible, thus improving the QoS for the

* Corresponding author.
E-mail address: azzam.mourad@lau.edu.lb (A. Mourad).

[1] https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/

[2] https://www.cisco.com/

https://doi.org/10.1016/j.future.2021.05.021
0167-739X/

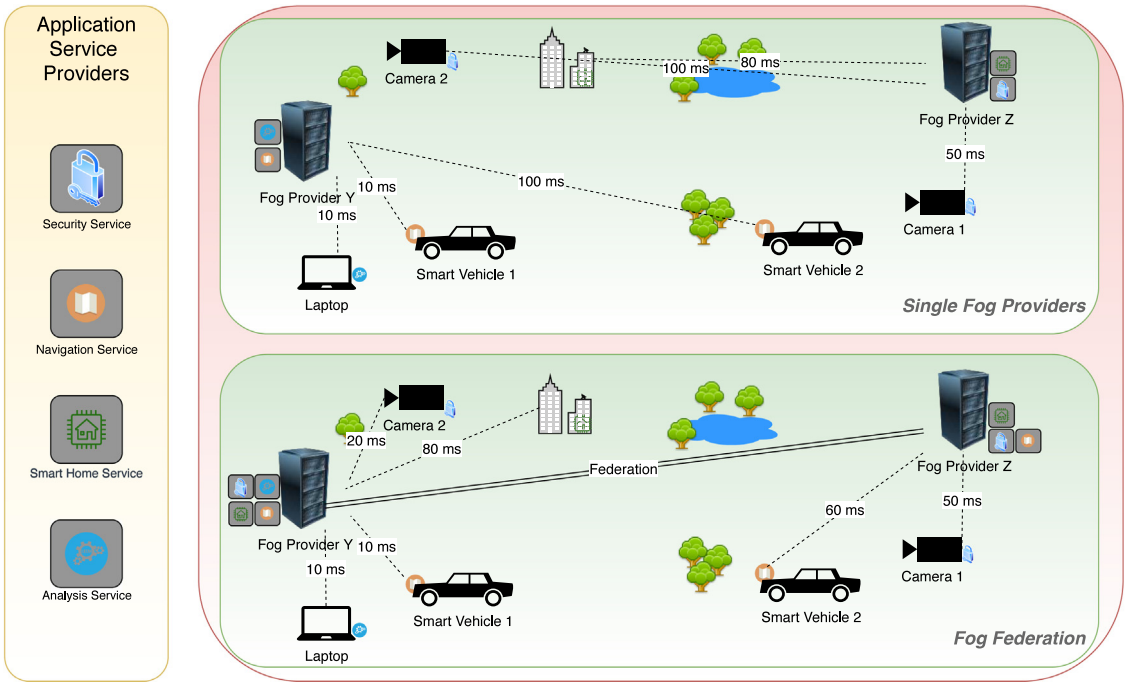


Fig. 1. Fog federations vs. single fog providers.

requests if efficient offloading/resource allocation mechanisms were applied [15,16]. On the other hand, it enables increasing the payoff of fog providers by renting out their unused resources and through expanding their geographical footprints without the need of being there physically. Hence, through fog federations, services can be deployed on more geographically distributed servers whenever there are spikes in the demands for serving such requests with acceptable QoS. In our experiments, we show that the federations can boost the service quality by improving many factors, such as the response time. In order to figuratively demonstrate the effectiveness of the fog federations, Fig. 1 illustrates how the ASPs are renting resources from fog providers to deploy their services. From the other side, users are trying to access these services by sending requests to the servers running the desired applications. Internally, federation members may offload requests to other members within the same federation in order to shorten the waiting delay for the requests. Thus, as illustrated, the latency values for the users are reduced, leading to a faster processing of the requests when the providers are federating compared to the typical single fog providers. It should be emphasized that the presented latency values in Fig. 1 are estimated after considering some cases from the dataset used in our simulation.

1.1. Problem statement

To motivate the concept of federating fog providers, we show in Fig. 2 the response time of serving the requests, with and without federations, using models presented in this paper. Precisely, using Matlab, we apply the initial formation mechanism by employing the K-means technique (presented in Algorithm 1) merged with the greedy service deployment mechanism (presented in Algorithm 2) in order to evaluate how a cluster of providers can cooperate for serving invocations compared to having each provider relying on its resources separately. The X-axis represents the timeline (in terms of hours), whereas the Y-axis is the response time (in milliseconds). The purple and the orange lines are the response time the servers need to process the requests issued by the IoT device with and without federating. We notice that at any specific time, the federation was able to guarantee a satisfactory response time on average due to the cooperation among fog providers. The response time is reduced by almost 26% on average when the services are being handled by fog federations. On the other hand, if the providers show no cooperation, then in some situations the QoS requested by the services could not be reached, leading to penalties. For instance, the fog providers, as rational decision-makers, might feel urged to renege on their commitments and deviate from their federations for seeking better ones that can satisfy them. Such an act reflects negatively on the federations that are suffering from members loss, due to the decrease of the shared resource pool in terms of computational capabilities and points-of-presence. These federations are referred to as *unstable fog federations*.

Hence, how can such fog federations be efficiently formed? It is a dilemma that encounters every fog provider due to the fact that the members of the federation directly affect the QoS [17]. It becomes challenging for fog providers to remain stable, i.e. choosing a federation and remaining committed to it instead of changing to another one. To the best of our knowledge, none of the proposed solutions have yet tackled the aforementioned problem.

1.2. Contributions

In this paper, we address the raised problems by proposing a novel fog formation scheme embedding evolutionary game theoretical model. Our approach offers to form stable fog federations in which no member has incentives to reallocate his resources somewhere else. We form the initial set of federations using the k-means clustering technique. It is an unsupervised learning model that forms clusters based on the similarities among nodes. Afterwards, we extend the formation with a learning-based evolutionary model. Such a game studies the conflict and cooperation among fog providers in the existence of dynamic strategies. It encompasses a state (strategy) where no fog provider has incentives to change its current federation, i.e. evolutionary stable strategy. We also propose a greedy service placement algorithm

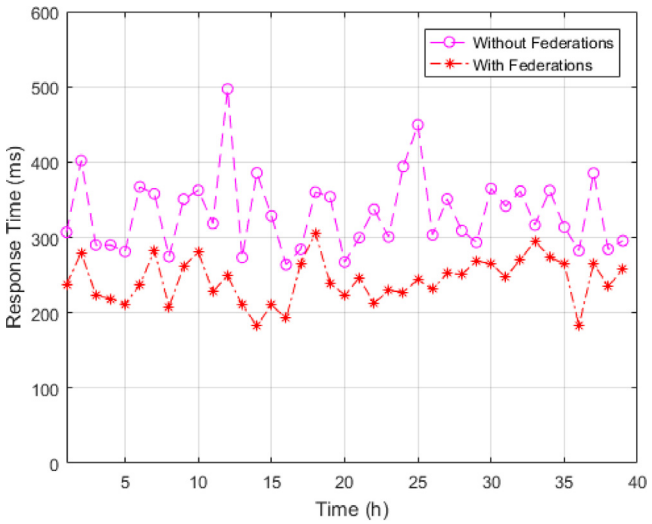


Fig. 2. Response time of requests with vs. without federations.

to cope with placing the services on the evolved generation in order to maintain relatively short network delays. The evolvement from a generation to another is presented using a decentralized algorithm that can be executed by the providers separately for reaching stability. We use EUA Datasets [18], containing data collected from real IoT devices, to simulate and evaluate our approach while comparing it with Genetic, Greedy, and Hedonic approaches. Experimental results explore that our proposed approach increases the total payoff for the federations and improves the QoS in terms of stability, response time, and availability. It is worth to mention that we have previously published a work addressing the cloud federation formation using Genetic and Evolutionary mechanisms [19]. Nonetheless, this work differs from the latter in terms of novelty and formalism due to the nature of the fog computing paradigm. Federated fog, different from federated cloud, is considered as a critical paradigm that supports real-time IoT applications due to the existence of fog servers near the IoT devices, hence the objective varies. In addition, applications are strongly dependent on reaching their desired QoS to keep providing a delay-free and smooth service which is not a crucial issue in the cloud federation paradigm. Furthermore, our proposed solution is boosted by a K-means clustering technique, followed by a latency-aware service placement mechanism and then by a decentralized/parallel decision making technique that is executed by the fog providers to imitate a further realistic environment. Thus, the main contributions of this work are summarized as follows:

- Adopting an evolutionary game mechanism that simulates the dynamicity of the fog providers, in terms of rational and irrational decision making. To the best of our knowledge, no previous work has ever addressed the dynamic strategies that encounter such a paradigm.
- Forming the initial set of fog federations using the k-means clustering technique. Such a technique allows forming federations based on providers' similarities. In our algorithm, we use the location of the providers to join neighboring fog providers altogether.
- Advancing a latency-aware greedy approach for placing services on the available fog nodes within the federation.
- Devising a decentralized algorithm for fog providers that leads to stabilizing the federations through reaching the evolutionarily stable strategy.

Outline of the paper. The rest of the paper is organized as follows. In Section 2, we overview the literature and compare the solutions in the literature with respect to the proposed approach. In Section 3, we formulate the federation formation problem. In Section 4, we propose our algorithm for solving the problem by employing an initial k-means clustering to form the initial set of federations, and then, studying the dynamicity of the fog providers through advancing an evolutionary game theoretical approach. We provide a numerical example in Section 5. After that, we discuss the results of running our algorithm to form the stable fog federations in Section 6. Finally, we give a conclusion in Section 7.

2. Related work

In this section, we give an overview of the literature and highlight on what is needed for advancing a quality fog federation formation mechanism.

2.1. Cloud federation formation approaches

Due to the wide range of techniques used for forming cloud federations, we select and discuss the most recent ones in this subsection. In [19], the authors advanced an approach based on a genetic and an evolutionary models to reach a cloud federations formation that is highly profitable, while reinforcing the stability among cloud providers. In [20], the authors addressed the problem of encountering passively malicious providers allocating their resources within cloud federations. They proposed a Maximin game-theoretical model that assists the broker to maximize the detection of the malicious providers. They were able to maximize the detection of malicious providers and improve the profit and QoS of the federations. In [21], the authors focused on increasing the profit of cloud service providers. They assisted the providers by making optimal decisions on where and when to allocate their computing resources. A linear optimization program was derived in [22] for helping the providers in a certain federation to tune their hosting and cooperation decisions according to the encountered workload and the available pool of resources. In [23] the authors advanced a technique for integrating resource and reputation management in the context of collaborative cloud computing. Their approach ameliorated the QoS offered due to the existence of a price-assisted resource/reputation control component. In [24], a formation mechanism was proposed to build a near-optimal federation. Mainly, their algorithm consists of merging and splitting clusters of providers together until reaching the best solution possible. Authors in [25] addressed the formation problem by using trust as a measurement among providers. They claimed to reach stability, profit maximization, and fairness through their formation mechanism. In [26], the authors proposed a genetic approach for cloud brokering. Their approach consists of forming the federations according to the QoS requested by the applications. However, none of the aforementioned works have considered the latency factor in its mechanism, where it is an essential component in forming fog federations for real-time applications. Thus, they cannot be applied to the fog level. Hence, a need for dedicated fog federation formation techniques rises in order to maintain an adequate QoS.

2.2. Fog service deployment and task-scheduling approaches

Initially, some works considered increasing the QoS by decreasing the latency between the fog and IoT devices. In [29], the authors presented a latency-aware application module management policy that increases the QoS and optimizes resource usage. Their policy can identify which applications should be deployed

Table 1
Comparison among related work.

	Latency-aware	Dynamic providers behaviors	Decentralized mechanism	Stable solution
[27]	✓	x	✓	✓
[10,28]	✓	x	✓	x
[19]	x	✓	x	✓
[21]	x	x	✓	x
[24,25]	x	x	x	✓
[20,22,26]	x	x	x	x
[8,9,11,23,29–34]	✓	x	x	x
Our solution	✓	✓	✓	✓

on the lower fog nodes (near the devices), and which shall be shifted to the upper fog nodes. In [9], the authors addressed the problem of forming fog clusters to locally process the set of offloaded requests by multiple users. The proposed approach covers both the task scheduling problem and cluster formation. The authors of [10] covered the same problem, however, they modeled the formation process as a coalitional game, where each player (fog node) joins its preferred cluster. In [35], a game theoretical model was proposed to increase the performance of the services by forming and joining communities to stabilize them and increase the QoS. In [8], the authors highlighted the problem of deploying fog servers, and how costly it can be. They proposed a dynamic mobile cloudlet cluster policy for fog computing by using cloudlets as a supplement for the fog server for offloading. The problem of allocating a set of docker containers to a set of volunteering devices to provide services on the fly was studied in [11]. Their main aim was to provide efficiently enough resources for real-time IoT applications requiring computation processing. They used a Multi-Objective Memetic algorithm to solve that problem. In [34], the authors addressed the need for a fitting service placement strategy in the Fog-to-Cloud infrastructure. Their aim was to design offloading strategies among cloud and fog resources. In [32], the authors advanced a model based on Reinforcement R-learning where they study the behavior of services' users and produce a suitable fog placement schedule based on the concept of average reward. Some scholars tackled the latency problem by developing an approach based on Matching game theory to derive intelligent scheduling decisions [28]. However, all of these efforts are not convenient in case of the absence of available resources for scheduling the tasks. In addition, most of these works lack a business-driven model that motivates the participants to show cooperation when deploying services.

2.3. Fog federation-based solutions

The concept of federating fog providers is still in its early phases. To the best of our knowledge, there are few published works tackling specifically federations in fog that can directly, or indirectly, enhance the QoS. In [27], the authors tackled the concept of federating fog providers for the sake of improving the latter's payoff. They modeled the problem as a Hedonic game with transferable utility where players are the fog providers seeking to maximize their own payoff. The authors in [30] provided a solution to reduce the latency of streaming video through federations. In particular, their approach was based on evaluating whether it was more convenient to fetch cached video data from neighboring nodes or to process them independently. In [31], authors proposed a micro-level resource management mechanism for fog federations, where they implemented a price-based workload balancing technique to limit offloading among units relative to other consortium members. In [33], the authors devised a novel federated fog architecture and modeled the federated fog formation problem using genetic and machine learning approaches to optimize the QoS. However, to the best of our knowledge, the main limitation of the fog federated-based solutions in the literature is not taking into account the fog providers' dynamicity. The abandonment of a critical provider on a particular federation may lead to reducing the offered QoS, resulting in a state of dissatisfaction among the ASPs.

2.4. Analysis

Table 1 highlights the main features of the related work compared to our proposed mechanism. Clearly, none of the aforementioned works in the literature has considered all of the latency factor, dynamicity, independence in decision making, and stability when forming the federations. Such four factors altogether can enhance the quality of the formed fog federation.

3. System model and problem formulation

Let us consider a set of fog providers $P = p_1, p_2, \ldots, p_n$, each of which has a number of servers $S_{p_i} = s_1, s_2, \ldots, s_m$ located at a particular geographical locations. Such servers are characterized by their processing power, measured in million instructions per second (MIPS). $F = f_1, f_2, \ldots, f_h$ is the set of federations under which the fog providers unite to form coalitions. We refer to the providers allocated in federation f_i at time t by P_{f_i}. At the same time, ASPs need to offer their services to the users in such a way that the offered QoS should meet the required minimum, otherwise the applications function poorly and ASPs lose some of their users. The federations handle sets of services by deploying them on the providers' servers (fog nodes). The set of applications allocated to federation f_i is represented by set $A_{f_i} = a_{1,f_i}, a_{2,f_i}, \ldots, a_{o,f_i}$. Likewise, each user is located at a specific location and is enrolled in a set of applications that sends out requests to the servers hosting these applications in order to process at a certain rate. Let the set $Usr_{a_j} = u_{1,a_j}, u_{2,a_j}, \ldots, u_{q,a_j}$ represent the users requesting service a_j (see Table 2).

The accrued cost C_{p_i} for a certain fog provider p_i is represented by the sum of the operational cost which is a function of CPU usage cost, storage and memory allocated, and energy usage of all of its servers $OC(s_j)$, in addition to their traffic cost in terms of allocated bandwidth $TC(s_j)$ as in the following equation:

$$C_{p_i} = \sum_{s_j \epsilon S_{p_i}} \left(OC(s_j) + TC(s_j) \right) \tag{1}$$

For a fair monetary distribution to the federation members, every fog provider p_i receives a percentage of federation f_i's total payoff. We consider the utility to be the cost of the servers subtracted from the payoff, divided by the computation power of the servers (i.e. total MIPS within the federation). Such utility can be expressed as follows:

$$U(f_i) = \frac{\left(\sum_{a_{k,f_i} \epsilon A_{f_i}} Payment(a_{k,f_i}) \times \sigma_{a_{k,f_i}} - \sum_{p_k \epsilon f_i} C_{p_k} \right)}{\sum_{p_k \epsilon P_{f_i}} \sum_{s_l \epsilon S_{p_k}} Pow(s_l)} \tag{2}$$

Table 2
Definitions.

Symbol	Description
P	set of all fog providers
p_i	fog provider i
S_{p_i}	set of all fog nodes (servers) belonging to p_i
s_i	fog node i
F	set of all fog federations
f_i	fog federation i
P_{f_i}	set of all fog providers allocated within f_i
A	set of all applications (services)
A_{f_i}	set of all applications (services) that belong to f_i
a_i	application i
Usr	set of all users
Usr_{a_i}	set of all users of application i
$\sigma_{a_{k,f_i}}$	discount factor of application a_k for federation f_i
α_{a_k}	deduction rate for not meeting a_k's requested QoS
$OC(s_i)$	operational cost of s_i
$TC(s_i)$	traffic cost of s_i
$C(p_i)$	total cost of p_i
$Payment(a_{k,f_i})$	the payment from a_k's ASP to f_i
$Pow(s_i)$	s_i's computation value in terms of MIPS
$U(f_i)$	utility of f_i
$\bar{U}$	average utility of F
$R_{p_{j,f_i}}$	payoff of provider p_j from federation f_i
ρ	the number of fog federations
x	the vector of distribution of available strategies
x_i	the percentage of the population adopting strategy i
$f_i(x)$	fitness function for strategy i
$\nu(x)$	the average fitness by the population

where $Pow(s_i)$ represents the value of server s_i in terms of computing power (i.e. MIPS) and $\sigma_{a_{k,f_i}}$ is considered to be the discount factor that alters the regular payment issued by the application a_k content provider if the federation f_i is not able to meet the minimum requirements and it can be expressed as:

$$\sigma_{a_{k,f_i}} = \begin{cases} 1 & \text{if QoS is met,} \\ \alpha_{a_k} & \text{otherwise.} \end{cases} \tag{3}$$

where α_{a_k} is the deduction rate due to not meeting the QoS, e.g. the average response time is above the agreed threshold.

Hence, the payoff of a provider p_{j,f_i} can be calculated as the following:

$$R_{p_{j,f_i}} = U(f_i) \times \left(\sum_{s_k \in S_{p_{j,f_i}}} Pow(s_k) \right) \tag{4}$$

To stabilize the set of federations, we need to reduce the variability of the payments per share. The least the difference among the latter, the more satisfied the fog providers would be, leading to fewer deviations from the federations. Such stability can be represented by the equation below:

$$minimize \sum_{f_i \in F} (\bar{U} - U_{f_i})^2 \tag{5}$$

where $\bar{U}$ is the average utility which is calculated using the following:

$$\bar{U} = \sum_{f_i \in F} U_{f_i} \times \frac{1}{\rho} \tag{6}$$

where ρ is the number of fog federations.

In the next section, we will discuss the formation and stability mechanism used to overcome the problem of unstable federations.

4. Evolutionary federated fog formation

Our proposed scheme is based on defining the formation process as an evolutionary game where each player has a preference function that leads the whole set of federations into its stable state.

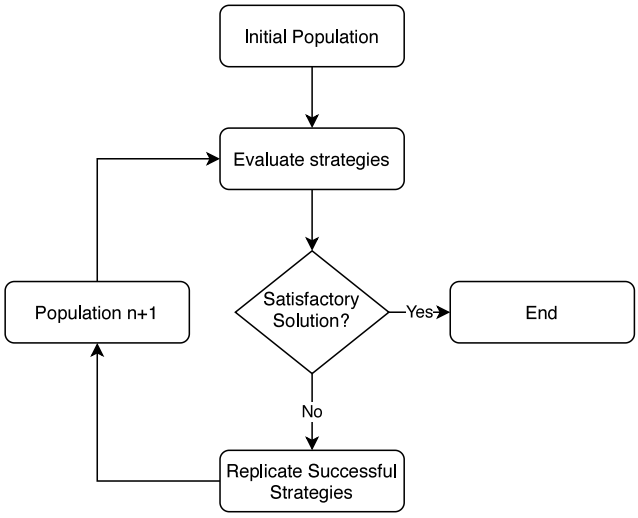


Fig. 3. Evolutionary game theory flowchart.

4.1. Background

Game theory is the science of the optimal decision-making of independent and competing players in a strategic environment [36]. Evolutionary game theory is used in settings where players are not obliged to be reasonable in their decisions [37]. It has been widely applied in many fields including biology [38] and economy [39]. Recently, it was adopted in cloud and fog computing environments for various purposes [40,41]. Such a game focuses on the dynamics of strategy change and on which among these strategies can persist in these settings. The success of a strategy is directly related to the other players' selected strategies. Hence, a strategy is evaluated by comparing it with the other strategies within the same population. A strategy that shows success will be replicated by other players as well. Once the evolutionarily stable strategy is adopted by a certain population, no player has intentions to deviate from it. Such a strategy can survive invasions of relatively small invaders trying to sabotage it. Hence, it leads to stabilizing the population. In other words, let *Pop* denote the population adopting an evolutionarily stable strategy X. *Pop* will not deviate from X if a small number of invaders, adopting strategy Y, joined the population. In contrast, the invaders will be forced to switch to X. Suppose that $O(X, Y)$ represents the outcome of an individual choosing strategy X facing another one with strategy Y. X is stable if it represents a strict Nash-Equilibrium ($[O(X, X)] > [O(X, Y)]$), or if $O(X, X) = O(Y, X)$ and $O(X, Y) > O(Y, Y)$. If any of these two applies, then no player has the incentive to deviate from their current strategy, even if the population gets invaded by a few mutants. Fig. 3 depicts the evolutionary mechanism. It shows that the population will keep on changing until reaching the state where all players are inheriting satisfactory strategy. The characteristics of such an evolution are similar to the ones inherited by the Genetic Algorithm [42].

4.2. Game characteristics

We present in this subsection the characteristics of the evolutionary game model to reach the evolutionarily stable strategy. The objective is to proceed from the initialization step to reach the 'End' state. The main components of this game are (1) the

players, (2) strategy, and (3) utility. Below, we break down each of these components and map them to our settings.

- **Players:** The players are the fog providers. Clearly, they are the decision-makers.
- **Strategy:** A strategy is represented by a particular fog federation. In particular, a player adopting strategy i can be interpreted as the player allocating its resources in federation i.
- **Utility:** The utility is the player's monetary payoff per 1 unit of allocated resources in a certain federation.

Hence, the problem becomes finding the best formation of fog federation that keeps all the fog providers satisfied with their coalition. To reach such a formation, the fog providers will deviate from their federations if they find a better payoff by joining others. Successful federations are most likely to be joined at time $t+1$ by unfortunate fog providers who are not satisfied with their selected federation at time t. The term evolution refers to this specific stage, i.e. the change that occurs from a state to another, making the population n evolve into $n+1$ where some of the players change their strategy. To represent such an evolution, we employ the replicator dynamics model that expresses the evolutionary dynamics [43]. In particular, we assume that the set $x = x_1, x_2, \ldots, x_m$ serves as the vector of distribution of available strategies within the targeted population. Intuitively, since all strategies are included in the set x, we can conclude the equation below:

$$\sum_{i=1}^{m} x_i = 1 \tag{7}$$

The replicator dynamic's general form is represented by $\dot{x}_i$ and is calculated as:

$$\dot{x}_i = x_i\,[f_i(x) - \nu(x)] \tag{8}$$

where $f_i(x)$ represents strategy i fitness function and $\nu(x)$ is the population's average fitness, which can be calculated from:

$$\nu(x) = \sum_{j=1}^{m} x_j f_j(x) \tag{9}$$

Mapped to our problem, the fitness function is the payoff of the provider per unit of resources obtained from Eq. (2). The replicator dynamics' defined by Eq. (8) shows the percentage of payoff increase for the individuals adopting a successful strategy. Once $\dot{x}_i = 0$ is obtained, the evolutionarily stable strategy is reached.

4.3. Stable fog federation formation

To demonstrate our approach, we divide the process into 3 different stages: Initialization, Player Strategy, and Stability.

4.3.1. Initialization

To initialize the population, we employ the K-means clustering technique. Such a technique follows the Expectation-Maximization approach [44]. It consists of assigning data points to their nearest cluster (i.e. Expectation). After that, the process of recomputing the centroid for each cluster takes place (i.e. Maximization). Algorithm 1 shows how federations are initialized. The algorithm takes the set of fog providers, represented by P, and the desired number of federations $\mathcal{K}$, and outputs the federations with their members. In Lines 1–3, we define and initialize the variables. In Lines 4–7, we set K initial centroids with random values. Then, we iterate on the fog providers and assign the providers according to their nearest centroid. Afterward, we recalculate the centroids. The recalculation function takes into consideration all of the clustered providers to calculate the new midpoint. Such steps are repeated until no more providers change centroids (Lines 8–17). Finally, we initialize a federation for each centroid and allocate the providers inside of them with respect to their centroid (Lines 18–23).

Algorithm 1: Initial Clustering

```
   Input: 𝒦, Pop
   Output: F
1  F ← ∅;
2  Centroids ← ∅;
3  terminate ← 0;
4  while 𝒦 > 0 do
5      C ← RandomPoint;
6      Centroids ← Centroids ∪ C;
7      𝒦 ← 𝒦 − 1;
8  while terminate ≠ 1 do
9      terminate ← 1;
10     forall p ∈ P do
11         C ← nearest_Centroid(p);
12         if p.centroid ≠ C then
13             p.centroid ← C;
14             terminate ← 0;
15     if terminate ≠ 1 then
16         forall C ∈ Centroids do
17             C ← recalculate(C);
18 forall C ∈ Centroids do
19     f ← ∅;
20     forall p ∈ P | p.centroid = C do
21         f ← f ∪ p;
22     F ← F ∪ f;
23 return F;
```

Once clustering is done, we employ Algorithm 2 to assign services to the provider's servers, with respect to the profit obtained, using a greedy allocation approach. The algorithm takes the set of services and the set of the providers allocated within federation f_i (A_{f_i} and P_{f_i} respectively) and outputs an allocation list $\mathcal{M}$ which has references to which services shall be deployed on which servers. After initializing the variables (Lines 1 and 2), we evaluate the performance of each server assigned to each service and store them inside the list (Lines 3–7). We order the list by the profit of each assignment in descending order (Line 8). Then, using a greedy technique, we pick the best available server for each service by keeping the best fit in terms of value (Lines 9 and 10).

Algorithm 2: Services Deployment

```
   Input: A_fi, P_fi
   Output: 𝓜
1  𝓜 ← ∅;
2  value ← 0;
3  forall a_l ∈ A_fi do
4      forall p_j ∈ P_fi do
5          forall s_k ∈ s_{k,p_j} do
6              value ← P(a_l) × σ_{a_l,f_i} − C_{s_k};
7              𝓜 ← 𝓜 ∪ [a : a_l, s : s_k, v : value];
8      Order 𝓜 by v descending;
9      forall m_k ∈ 𝓜 do
10         removeAll m_l from 𝓜|m_l.s = m_k.s, m_l.a ≠ m_k.a;
11 return 𝓜;
```

4.3.2. Player strategy

A player may reflect on its current strategy (i.e. federation) and decide that it might be better for him to switch to another. To imitate such an act, we devise a decentralized algorithm, that can be executed by the fog provider, to decide on which federation to join according to their preferences. Algorithm 3 shows how a fog provider may interact according to the evaluation of the available federations. The algorithm takes as arguments the fog provider's current federation f_i, the set of federations F, and the current vector of distribution of available strategies x. The output of the algorithm is the provider's preferred federation at the current time. Lines 1–6 consists of initializing the parameters. At Lines 7 and 8, the provider calculates the average utility by summing up all federations utilities with regards to the percentage of the population adopting them. At Line 9, if the player notices that they are not getting at least the average utility in terms of value, compared to all other strategies being played, then they consider switching into a more profitable federation. At Line 10, the player filters the federations, such that only the more profitable are being kept. After that, these federations get stored inside F' after being sorted in descending order according to the player's preferences. Lines 11–17 presents how the player sets his next strategy. Finally, the player selects his preferred strategy, according to how preferable a strategy with respect to the others is. It is worth mentioning that we imitate the player's behavior in terms of preferences and with the presence of slight randomness.

Algorithm 3: Player Preference

Input: f_i, F, x
Output: f
1 $\alpha \leftarrow 0$;
2 $r \leftarrow random(0, 1)$;
3 $f \leftarrow f_i$;
4 $\nu(x) \leftarrow 0$;
5 $F' \leftarrow \emptyset$;
6 $x' \leftarrow \emptyset$;
7 **forall** $f_j \in F$ **do**
8 $\quad \nu(x) \leftarrow \nu(x) + x_j \times utility(f_j|F)$;
9 **if** $\nu(x) > utility(f_i|F)$ **then**
10 $\quad F' \leftarrow sort(F|f_j \in F, utility(f_j|F) > \nu(x)$;
11 $\quad$ **forall** $f_j \in F'$ **do**
12 $\quad\quad x' \leftarrow x' \cup x_j(\nu - utility(f_j|F))$;
13 $\quad$ **forall** $f_j \in F'$ **do**
14 $\quad\quad \alpha \leftarrow \frac{x_j}{\sum x_k \in x'(x_k)}$;
15 $\quad\quad$ **if** $r < \alpha$ **then**
16 $\quad\quad\quad f \leftarrow f_j$;
17 $\quad\quad\quad$ break;
18 return f;

4.3.3. Discussion

Evolutionary games are time-aware in the sense that the population is studied and evaluated over time. After setting the initial formation at time t, providers will start acting as rational beings for seeking better federations. To further imitate the dynamicity of such a non-cooperative game and the irrationality of the providers, players are allowed to change strategies at any particular time repetitively until they are satisfied, i.e. they do not have incentives anymore to break from their current federation. Having all players executing the decentralized algorithm over time will result in solving $\dot{x}_i = 0$ for all $x_i \in x$. In other words, it will lead to a state where all the utilities are equal or similar to the average. Thus, any deviation attempt from that state will lead back to it again, as it represents the evolutionarily stable strategy.

Table 3
Available fog provider.

Fog provider	Number of fog nodes	Latitude	Longitude
A	3	10	10
B	4	12	11
C	2	13	9
D	4	11	13
E	1	1	3
F	2	3	3
G	4	7	13
H	6	6	12
I	2	2	3
J	2	2	4

Table 4
Fog federations using K-means.

Federation	Fog provider
f_1	A B C D
f_2	E F I J
f_3	G H

Table 5
Application service providers.

ASP #	Agreed price	Chosen federation
1	15 $/h	f_1
2	5 $/h	f_1
3	20 $/h	f_2
4	30 $/h	f_2
5	10 $/h	f_3
6	20 $/h	f_3

The main advantage of an evolutionary non-cooperative game over classical cooperative and non-cooperative games is that the former survives mutant strategies invasions by shifting the population towards equilibrium [38]. There is indeed an assumption of players cooperating together to increase the coalitions' payoff in cooperative games (e.g. [45,46]). Such kind of games impose sequential strategies where steps and events happen one at a time to reach equilibrium. In contrast, players engage individually in order to increase their own payoff against the strategies of other players in non-cooperative games. Nonetheless, both types do not consider mutant strategies invasions and neglect outside circumstances that may force some players to deviate from their current strategies. In our problem, we consider a vast number of players (i.e. fog service providers) and plenty of strategies (i.e. fog federations) that are joinable at any time by any of these players. Thus, in order to avoid drifts from the desired stable state and to maintain equilibrium in the QoS, we adopt an evolutionary game theoretical model for replicating successful strategies among players.

5. Numerical example

In this section, we evaluate the proposed scheme in terms of forming stable fog federations. We consider a set of 10 fog providers with different locations and different numbers of participating nodes as presented in Table 3. We assume that all fog nodes are equal in terms of computing power and total cost (5000 MIPS and 0.5$/h respectively). By applying the initial clustering technique, defined via Algorithm 1, we group up neighboring fog providers together. By setting K to 3, we get the federations given in Table 4.

Our solution grouped up providers A, B, C, and D into the first federation, providers E, F, I, and J into the second federation, and the remaining providers (G and H) are grouped into the third federation. Table 5 represents the ASPs and the federations they have chosen to request computing resources from.

Table 6
Fog providers' utility and payoff.

Fog provider	Federation	Utility	Payoff ($/h)
A	f_1	0.0002	8.5
B	f_1	0.0002	11.33
C	f_1	0.0002	5.67
D	f_1	0.0002	11.33
E	f_2	0.0013	2.83
F	f_2	0.0013	5.67
G	f_3	0.0005	11.33
H	f_3	0.0005	17
I	f_2	0.0013	5.67
J	f_2	0.0013	5.67

Then the payoff is distributed based on to Eqs. (2) and (4). We set σ is equal to 1. Hence, the utility of federation f_1 is computed as follows:

$$U(f_1) = \frac{1}{13 \times 5000} \times ((15+5) \times 1) - (3 \times 0.5 + 4 \times 0.5 + 2 \times 0.5 + 4 \times 0.5) = 0.0002$$

whereas the payoff of the fog providers in exchange to their fog nodes allocated in federation f_1 would be given as:

$$R(A) = 3 \times 5000 \times U(f_1) = 3.12\$$$
$$R(B) = 4 \times 5000 \times U(f_1) = 4.155\$$$
$$R(C) = 2 \times 5000 \times U(f_1) = 2.07\$$$
$$R(D) = 4 \times 5000 \times U(f_1) = 4.155\$$$

The utility of federation f_2 is computed the same way:

$$U(f_2) = \frac{1}{7 \times 5000} \times ((20+30) \times 1) - (1 \times 0.5 + 2 \times 0.5 + 2 \times 0.5 + 2 \times 0.5) = 0.0013$$

and the payoff of federation f_2's members are:

$$R(E) = 1 \times 5000 \times U(f_2) = 6.642\$$$
$$R(F) = 2 \times 5000 \times U(f_2) = 13.286\$$$
$$R(I) = 2 \times 5000 \times U(f_2) = 13.286\$$$
$$R(J) = 2 \times 5000 \times U(f_2) = 13.286\$$$

Likewise, federation f_3's utility and its members' payoff are calculated as:

$$U(f_3) = \frac{1}{10 \times 5000} \times ((10+20) \times 1) - (4 \times 0.5 + 6 \times 0.5) = 0.0005$$

$$R(G) = 4 \times 5000 \times U(f_3) = 10\$$$
$$R(H) = 6 \times 5000 \times U(f_3) = 15\$$$

Table 6 summarizes the aforementioned calculations. We notice that some of the providers would not be satisfied, thus starting to deviate from their current federations. For instance, providers B and G are both having the same number of fog nodes and specs. However, due to G's allocation in f_3, it is getting an hourly payoff of more than 240% of what B is acquiring from f_1. Hence, provider B might get tempted to break from f_1 and join another federation for the sake of improving its payoff. Algorithm 3 reflects such behavior by solving the replicator dynamic's $\dot{x}_i = 0$ in order to obtain a satisfactory solution (i.e. fog federations formation) for all fog providers. Since the algorithm is time aware and executed in a decentralized manner, provider B may execute the algorithm to select the preferred federation at time t by B calculating first $\nu(x)$:

$$\nu(x) = 0.0002 \times \frac{13}{30} + 0.0013 \times \frac{7}{30} + 0.0005 \times \frac{10}{30} = 0.00055$$

Afterwards, it compares its utility with $\nu(x)$. If it does not meet the average utility, then it starts seeking other federations having a utility higher than $\nu(x)$. In this example, the only available federation that meets such a condition is f_2. So provider B should consider f_2 as the next strategy to adapt at time $t+1$. After that, all the utilities for the federations affected by such a move (i.e. having B switching from f_1 to f_2) are recalculated as the formation becomes different from what it was at time t. The same process repeats until the algorithm returns the same federation which is represented in Table 7. This distribution of resources among federations would remain stable and cannot be sabotaged by invaders, since the algorithm will lead back to the same (or to a similar) distribution. Thus, the QoS will remain stable for the clients.

Table 7
Fog providers' utility and payoff after convergence.

Fog Provider	Federation	Utility	Payoff ($/h)
A	f_2	0.00056	3.12
B	f_1	0.00056	4.155
C	f_1	0.00056	2.07
D	f_2	0.00056	4.155
E	f_3	0.00056	6.642
F	f_2	0.00056	13.286
G	f_3	0.00056	10
H	f_2	0.00056	15
I	f_3	0.00056	13.286
J	f_3	0.00056	13.286

6. Experimental evaluation

6.1. Experimental setup

The simulation has been conducted using *Matlab R2019a* on a *Windows 10* equipped with Intel Core i7-9700F and 32 GB of RAMs. We used EUA Datasets,[3] which have data collected from IoT and Edge devices. We assigned random transmission delays on the links and generated 40 services. The minimum demanded response time by the services varies from 250 to 350 ms. Each IoT device u_i has a set of various services as mentioned in Section 3, and a request rate per second ($0 \leq rr_{d_i}^{s_j} \leq 1$) for them. Each request needs processing of [800–1200] million instructions to acquire a result. We limit the number of fog provider to 100 and IoT devices to 600. Each provider has [1–3] available servers, each with a processing power of [4000–6000] MIPS. Finally, we consider $\mathcal{K} = 10$ after applying the Elbow method on evaluating the fittest number of federations, according to the provider's distribution.

6.2. Results and discussion

In this section, we evaluate our evolutionary approach and the proposed Algorithms in terms of stability, response time, availability, player utility and federation payoff. We compare our evolutionary game to three different approaches: (1) the Genetic approach, presented in [33], (2) the Hedonic approach, presented in [45], due to their resemblance to our approach, and (3) a greedy approach presented in Algorithm 4. This algorithm is similar to Algorithm 3 in a way that it is also executed by each player independently. However, it further pushes the fog providers to

[3] https://github.com/swinedge/eua-dataset

Algorithm 4: Greedy Player Preference

Input: f_i, F
Output: f
$\alpha \leftarrow 0$;
$r \leftarrow random(0, 1)$;
$f \leftarrow f_i$;
$f' \leftarrow f_i$;
$\nu \leftarrow -\infty$;
forall $f_j \in F$ **do**
  if $\nu < utility(f_j|F)$ **then**
    $\nu \leftarrow utility(f_j|F)$;
    $f' \leftarrow f_j$
if $r < \alpha$ **then**
  $f \leftarrow f'$;
  break;
return f;

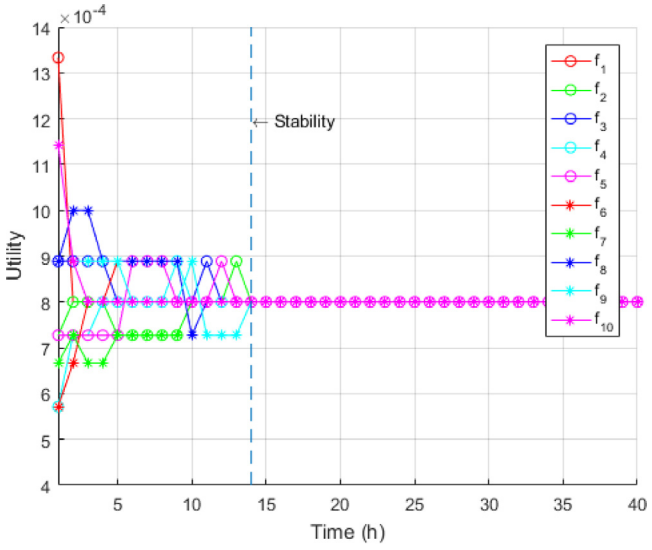


Fig. 4. Stability: evolutionary game approach.

always reallocate their resources to the best available federation, i.e., the federation with the highest utility. For a fair comparison, we assign the same initial solution to all approaches and each one of them updates that solution using its own mechanism.

Figs. 4–7 depict the utilities of federations when the formation is maintained by the Evolutionary, Greedy, Genetic, and Hedonic approaches, respectively. The X-axis represents the time-line and the Y-axis represents the utility of the strategy. We notice that when $x < 14$, the utilities of the federations were not stable at all. However, the federations converge at $x = 14$ when the Evolutionary approach is used. This is due to the stability mechanism implanted in Algorithm 3 where the population realizes the evolutionarily stable strategy. On the other hand, the population could not stabilize at all when relying on the other reallocation approaches and the variance of the utilities remained high for the first 40 h.

The total payoff of all the federations is presented in Fig. 8 where the Y-axis represents the payoff in terms of USD. According to the simulation, the evolutionary approach is able to always outperform the benchmarked models and maintain a higher payoff that stabilizes at $x = 14$, whereas the Greedy, Genetic, and Hedonic approaches are suffering from a lack of resources in some federations, which leads to having some non-deployed services and reduction in the payoff.

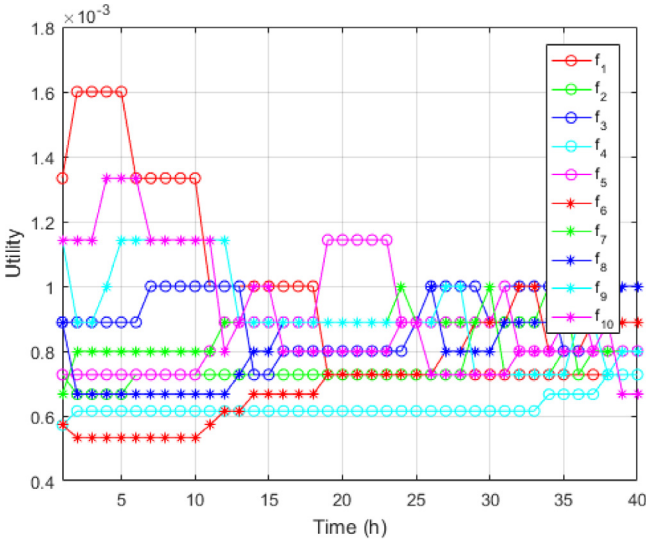


Fig. 5. Stability: greedy approach.

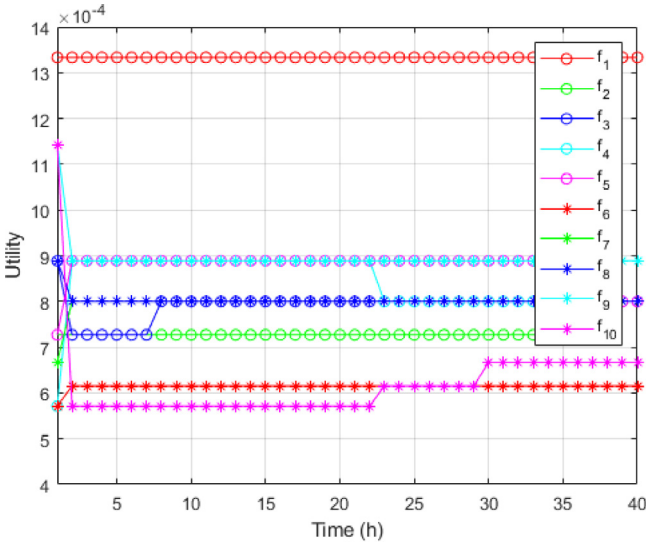


Fig. 6. Stability: genetic approach.

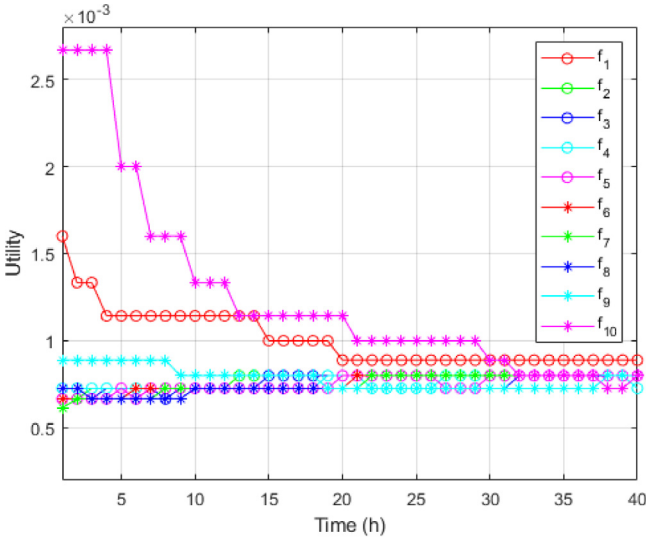


Fig. 7. Stability: hedonic approach.

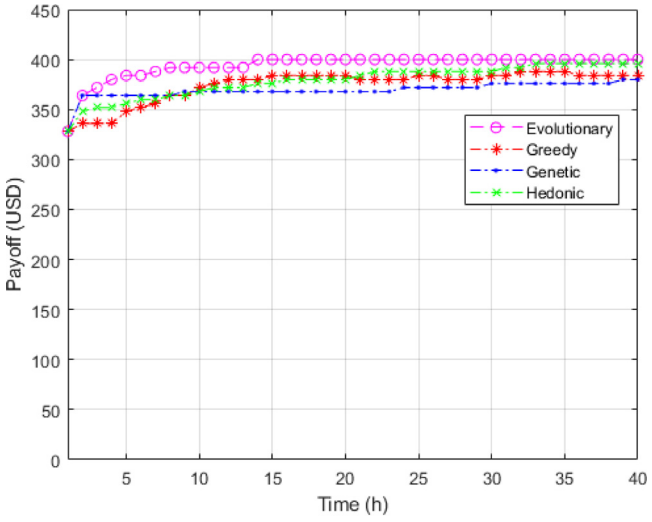


Fig. 8. Total federations' payoff.

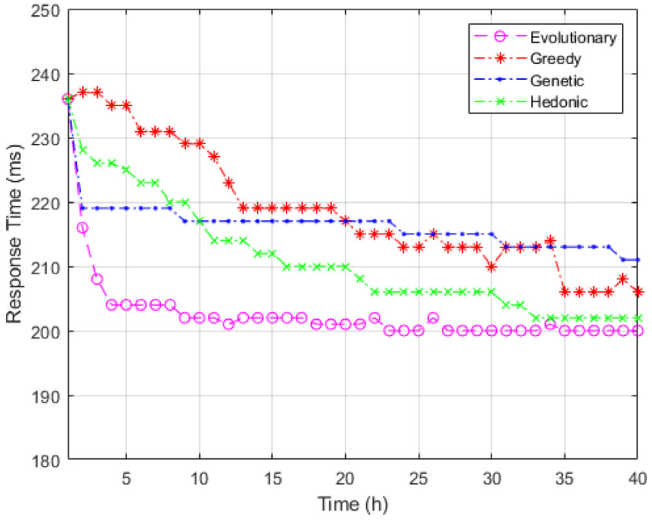


Fig. 9. Response time of requests.

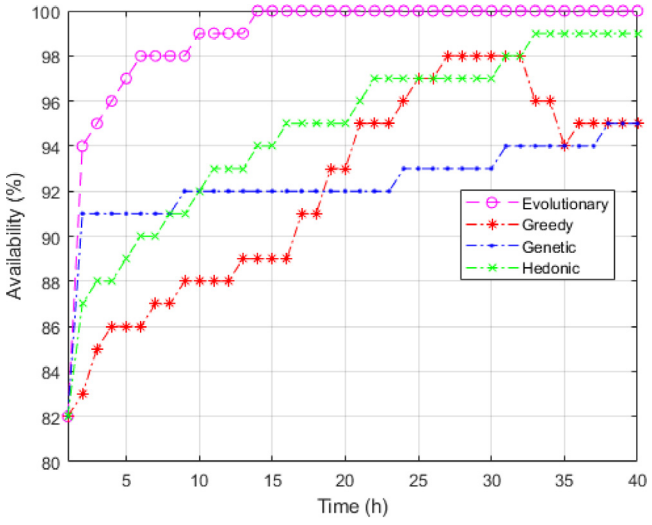


Fig. 10. Availability of requests.

Finally, we compare the performance of forming the fog federations using the Evolutionary, Greedy, Genetic, and Hedonic approaches in terms of services' response time and availability in Figs. 9 and 10, respectively. While the X-axis remains the timeline for both figures, the Y-axis represents the response time in milliseconds for Fig. 9, and the percentage of availability for Fig. 10. All approaches are able to decrease the response time and increase the availability from the initial formation. However, the evolutionary game outperforms the other approaches and stabilizes the services at full availability and a lower response time due to the stability mechanism reached at $x = 14$, whereas the other approaches still suffer from the lack of a satisfactory strategy that pleases the participants and reduces their obligations to deviate from their federations.

7. Conclusion

Fog federation is a concept worth exploring since it helps to increase the computational capabilities of the fog providers and can provide improved QoS for real-time applications. On the other hand, federations may suffer from instabilities due to the providers' dynamicity that may lead some providers to leave their coalitions and join others that are more profitable. In this paper, we devised an evolutionary model to stabilize the federations. We modeled the non-cooperative scheme as an evolutionary game and advanced a decentralized model that inherits the settings of the replicator dynamics in order to reach an evolutionary stable strategy. The numerical results show how the formation process converges to a stable state which improves the payoff and QoS in terms of services' availability and response-time.

CRediT authorship contribution statement

Ahmad Hammoud: Software, Conceptualization, Methodology, Validation, Formal analysis, Investigation, Resources, Writing - original draft, Writing - review & editing, Visualization. **Hadi Otrok:** Supervision, Conceptualization, Methodology, Validation, Formal analysis, Investigation, Resources, Writing - review & editing, Visualization. **Azzam Mourad:** Supervision, Conceptualization, Methodology, Validation, Formal analysis, Investigation, Resources, Writing - review & editing, Visualization. **Zbigniew Dziong:** Supervision, Conceptualization, Methodology, Validation, Formal analysis, Investigation, Resources, Writing - review & editing, Visualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

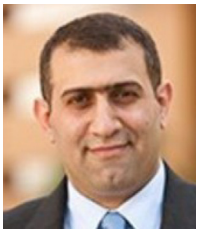
This work has been supported by Ecole de Technologie Superieure, Khalifa University, and Lebanese American University.

References

[1] A. Gharaibeh, A. Khreishah, M. Mohammadi, A. Al-Fuqaha, I. Khalil, A. Rayes, Online auction of cloud resources in support of the Internet of Things, IEEE Internet Things J. 4 (5) (2017) 1583–1596.
[2] M. Khabbaz, C. Assi, S. Sharafeddine, Multi-hop V2u path availability analysis in UAV-assisted vehicular networks, IEEE Internet Things J. (2021).
[3] S.A. Rahman, A. Mourad, M. El Barachi, W. Al Orabi, A novel on-demand vehicular sensing framework for traffic condition monitoring, Veh. Commun. 12 (2018) 165–178.
[4] Y.-Y. Shih, W.-H. Chung, A.-C. Pang, T.-C. Chiu, H.-Y. Wei, Enabling low-latency applications in fog-radio access networks, IEEE Netw. 31 (1) (2016) 52–58.

[5] H. Otrok, A. Mourad, J.-M. Robert, N. Moati, H. Sanadiki, A cluster-based model for QoS-OLSR protocol, in: 2011 7th International Wireless Communications and Mobile Computing Conference, IEEE, 2011, pp. 1099–1104.
[6] M. Chiang, T. Zhang, Fog and IoT: An overview of research opportunities, IEEE Internet Things J. 3 (6) (2016) 854–864.
[7] V. Stantchev, A. Barnawi, S. Ghulam, J. Schubert, G. Tamm, Smart items, fog and cloud computing as enablers of servitization in healthcare, Sens. Transducers 185 (2) (2015) 121.
[8] Y. Li, N.T. Anh, A.S. Nooh, K. Ra, M. Jo, Dynamic mobile cloudlet clustering for fog computing, in: Proceedings of the International Conference on Electronics, Information and Communication, ICEIC, IEEE, 2018, pp. 1–4.
[9] J. Oueis, E.C. Strinati, S. Barbarossa, The fog balancing: Load distribution for small cell cloud computing, in: Proceedings of the 81st Vehicular Technology Conference, VTC Spring, IEEE, 2015, pp. 1–6.
[10] Y. Sun, T. Dang, J. Zhou, User scheduling and cluster formation in fog computing based radio access networks, in: Proceedings of the IEEE International Conference on Ubiquitous Wireless Broadband, ICUWB, IEEE, 2016, pp. 1–4.
[11] H. Sami, A. Mourad, Dynamic on-demand fog formation offering on-the-fly IoT service deployment, IEEE Trans. Netw. Serv. Manag. (2020).
[12] A. Hammoud, H. Sami, A. Mourad, H. Otrok, R. Mizouni, J. Bentahar, AI, blockchain, and vehicular edge computing for smart and secure IoV: Challenges and directions, IEEE Internet Things Mag. 3 (2) (2020) 68–73.
[13] W. Fawaz, C. Abou-Rjeily, C. Assi, UAV-Aided cooperation for FSO communication systems, IEEE Commun. Mag. 56 (1) (2018) 70–75.
[14] A. Al-Hilo, M. Samir, C. Assi, S. Sharafeddine, D. Ebrahimi, Cooperative content delivery in UAV-RSU assisted vehicular networks, in: Proceedings of the 2nd ACM MobiCom Workshop on Drone Assisted Wireless Communications for 5G and Beyond, 2020, pp. 73–78.
[15] H. Tout, C. Talhi, N. Kara, A. Mourad, Selective mobile cloud offloading to augment multi-persona performance and viability, IEEE Trans. Cloud Comput. 7 (2) (2016) 314–328.
[16] H. Marshoud, H. Otrok, H. Barada, R. Estrada, A. Jarray, Z. Dziong, Resource allocation in macrocell-femtocell network using genetic algorithm, in: 2012 IEEE 8th International Conference on Wireless and Mobile Computing, Networking and Communications, WiMob, IEEE, 2012, pp. 474–479.
[17] B. Rochwerger, D. Breitgand, E. Levy, A. Galis, K. Nagin, I.M. Llorente, R. Montero, Y. Wolfsthal, E. Elmroth, J. Caceres, et al., The reservoir model and architecture for open federated cloud computing, IBM J. Res. Dev. 53 (4) (2009) 4–1.
[18] P. Lai, Q. He, M. Abdelrazek, F. Chen, J. Hosking, J. Grundy, Y. Yang, Optimal edge user allocation in edge computing with variable sized vector bin packing, in: Proceedings of the International Conference on Service-Oriented Computing, Springer, 2018, pp. 230–245.
[19] A. Hammoud, A. Mourad, H. Otrok, O.A. Wahab, H. Harmanani, Cloud federation formation using genetic and evolutionary game theoretical models, Future Gener. Comput. Syst. 104 (2020) 92–104.
[20] A. Hammoud, H. Otrok, A. Mourad, O.A. Wahab, J. Bentahar, On the detection of passive malicious providers in cloud federations, IEEE Commun. Lett. 23 (1) (2018) 64–67.
[21] I. Goiri, J. Guitart, J. Torres, Characterizing cloud federation for enhancing providers' profit, in: 2010 IEEE 3rd International Conference on Cloud Computing, IEEE, 2010, pp. 123–130.
[22] S. Rebai, M. Hadji, D. Zeghlache, Improving profit through cloud federation, in: Proceedings of the 12th Annual IEEE Consumer Communications and Networking Conference, CCNC, IEEE, 2015, pp. 732–739.
[23] H. Shen, G. Liu, An efficient and trustworthy resource sharing platform for collaborative cloud computing, IEEE Trans. Parallel Distrib. Syst. 25 (4) (2013) 862–875.
[24] L. Mashayekhy, M.M. Nejad, D. Grosu, Cloud federations in the sky: Formation game and mechanism, IEEE Trans. Cloud Comput. 3 (1) (2015) 14–27.
[25] A. Dhole, M.V. Thomas, K. Chandrasekaran, An efficient trust-based game-theoretic approach for cloud federation formation, in: Proceedings of the 3rd International Conference on Advanced Computing and Communication Systems, ICACCS, 1, IEEE, 2016, pp. 1–6.
[26] G.F. Anastasi, E. Carlini, M. Coppola, P. Dazzi, QoS-Aware genetic cloud brokering, Future Gener. Comput. Syst. 75 (2017) 1–13.
[27] C. Anglano, M. Canonico, P. Castagno, M. Guazzone, M. Sereno, A game-theoretic approach to coalition formation in fog provider federations, in: Proceedings of the Third International Conference on Fog and Mobile Edge Computing, FMEC, IEEE, 2018, pp. 123–130.
[28] S. Arisdakessian, O.A. Wahab, A. Mourad, H. Otrok, N. Kara, Fogmatch: An intelligent multi-criteria IoT-fog scheduling approach using game theory, IEEE/ACM Trans. Netw. (2020).
[29] R. Mahmud, K. Ramamohanarao, R. Buyya, Latency-aware application module management for fog computing environments, ACM Trans. Internet Technol. 19 (1) (2018) 1–21.
[30] V. Veillon, C. Denninnart, M.A. Salehi, F-FDN: Federation of fog computing systems for low latency video streaming, in: Proceedings of the 3rd International Conference on Fog and Edge Computing, ICFEC, IEEE, 2019, pp. 1–9.
[31] Z. Sharmin, A.W. Malik, A.U. Rahman, R.M. Noor, Towards sustainable micro-level fog federated load-sharing in internet of vehicles, IEEE Internet Things J. (2020).
[32] P. Farhat, H. Sami, A. Mourad, Reinforcement R-learning model for time scheduling of on-demand fog placement, J. Supercomput. 76 (1) (2020) 388–410.
[33] H. Shamseddine, J. Nizam, A. Hammoud, A. Mourad, H. Otrok, H. Harmanani, Z. Dziong, A novel federated fog architecture embedding intelligent formation, IEEE Netw. (2020).
[34] V.B. Souza, X. Masip-Bruin, E. Marín-Tordera, S. Sànchez-López, J. Garcia, G.-J. Ren, A. Jukan, A.J. Ferrer, Towards a proper service placement in combined Fog-to-Cloud (F2C) architectures, Future Gener. Comput. Syst. 87 (2018) 1–15.
[35] B. Khosravifar, J. Bentahar, R. Mizouni, H. Otrok, M. Alishahi, P. Thiran, Agent-based game-theoretic model for collaborative web services: Decision making analysis, Expert Syst. Appl. 40 (8) (2013) 3207–3219.
[36] R.B. Myerson, Game Theory, Harvard university press, 2013.
[37] J.M. Smith, G.R. Price, The logic of animal conflict, Nature 246 (5427) (1973) 15–18.
[38] P. Hammerstein, R. Selten, Game theory and evolutionary biology, in: Handbook of Game Theory with Economic Applications, vol. 2, 1994, pp. 929–993.
[39] D. Friedman, On economic applications of evolutionary game theory, J. Evol. Econ. 8 (1) (1998) 15–43.
[40] Y. Sun, F. Lin, N. Zhang, A security mechanism based on evolutionary game in fog computing, Saudi J. Biol. Sci. 25 (2) (2018) 237–241.
[41] S. Yan, M. Peng, M.A. Abana, W. Wang, An evolutionary game for user access mode selection in fog radio access networks, IEEE Access 5 (2017) 2200–2210.
[42] A. Nader, D. Azar, Searching for activation functions using a self-adaptive evolutionary algorithm, in: Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion, 2020, pp. 145–146.
[43] P. Schuster, K. Sigmund, Replicator dynamics, J. Theoret. Biol. 100 (3) (1983) 533–538.
[44] R.A. Haraty, M. Dimishkieh, M. Masud, An enhanced k-means clustering algorithm for pattern discovery in healthcare data, Int. J. Distributed Sens. Netw. 11 (6) (2015) 615740.
[45] C. Anglano, M. Canonico, P. Castagno, M. Guazzone, M. Sereno, Profit-aware coalition formation in fog computing providers: A game-theoretic approach, Concurr. Comput.: Pract. Exper. 32 (21) (2020) e5220.
[46] N. Moati, H. Otrok, A. Mourad, J.-M. Robert, Reputation-based cooperative detection model of selfish nodes in cluster-based QoS-OLSR protocol, Wirel. Pers. Commun. Int. J. 75 (3) (2014) 1747–1768.

Ahmad Hammoud is currently a Ph.D. candidate in Electrical Engineering Department at Ecole de Technologie Superieure (ETS). He received his BS in Business Computing and MS in Computer Science from the Lebanese University in 2016 and the Lebanese American University in 2019, respectively. His current research interests include cloud federation, fog federation, game theory, Blockchain, Artificial Intelligence, and security.



Hadi Otrok holds an associate professor position in the department of EECS at Khalifa University, Abu Dhabi, UAE. He received his Ph.D. in ECE from Concordia University. He is a senior member at IEEE, and associate editor at: Ad-hoc networks (Elsevier) and IEEE Network. He co-chaired several committees at various IEEE conferences. His research interests include the domain of computer and network security, crowd sensing and sourcing, ad hoc networks, and cloud security.

Azzam Mourad received his M.Sc. in CS from Laval University, Canada (2003) and Ph.D. in ECE from Concordia University, Canada (2008). He is currently an associate professor of computer science with the Lebanese American University and an affiliate associate professor with the Software Engineering and IT Department, Ecole de Technologie Superieure (ETS), Montreal, Canada. He published more than 100 papers in international journal and conferences on Security, Network and Service Optimization and Management targeting IoT, Cloud/Fog/Edge Computing, Vehicular and Mobile Networks, and Federated Learning. He has served/serves as an associate editor for IEEE Transaction on Network and Service Management, IEEE Network, IEEE Open Journal of the Communications Society, IET Quantum Communication, and IEEE Communications Letters, the General Chair of IWCMC2020, the General Co-Chair of WiMob2016, and the Track Chair, a TPC member, and a reviewer for several prestigious journals and conferences. He is an IEEE senior member.

Zbigniew Dziong received the Ph.D. degree from the Warsaw University of Technology, Poland where he also worked as an Assistant Professor. From 1987 to 1997, he was with INRS-Telecommunications, Montreal, QC, Canada. From 1997 to 2003, he worked at Bell Labs, Holmdel, NJ, USA. Since 2003, he has been with the École de Technologie Supérieure (University of Quebec), Montreal, as a Full Professor. He is an expert in the domain of performance, control, protocol, architecture and resource management for data, wireless and optical networks. He has participated in research projects for many leading telecommunication companies including Bell Labs, Nortel, Ericsson, and France Telecom. He won the prestigious STENTOR Research Award (1993, Canada) for collaborative research. His monograph *ATM Network Resource Management* (McGraw Hill, 1997) has been used in several universities for graduate courses.