Investigating Mobility-aware Strategies for IoT Services Placement in the Fog under Energy and QoS Constraints

Tanissia Djemai, Patricia Stolf, Thierry Monteil, and Jean-Marc Pierson

***Abstract*—Mobility of Internet of Things (IoT) objects is a key characteristic of IoT environments. It brings dynamicity, uncertainty and raises many challenges when it is associated with computation and network resources management for IoT applications. The resources management problem under objects mobility consideration is even more sensitive if we consider that various IoT applications have stringent Quality of Service (QoS) needs. Fog Computing is a distributed computation paradigm that increases data centers computation and storage abilities with nodes between end-users and the Cloud. Fog computing offers a large distributed infrastructure to support IoT applications needs by bringing services closer to end users. However, Fog infrastructures inherit the energy greediness characteristics of both data centers and network infrastructures. This work investigates the IoT services placement problem in the Fog as an optimization problem to minimize energy consumption and enhance QoS while considering mobility of IoT objects. We model the placement problem as a multi-objective optimization problem and we propose a location history based mobility model (HTM) to estimate future locations of IoT mobile nodes. We propose a framework composed of online strategies for IoT services placement and a Mobility-aware Genetic Algorithm (MGA) for services migrations. We evaluate our strategies through iFogSim simulator and compare the proposed framework to migrations and placement strategies from the literature based on Shortest Access Point migration strategy (SAP) and with Penguins Search Optimization Algorithm (PeSOA). Experiments show that the proposed framework outperforms literature approaches for the considered objectives and for various configurations of the mobile environment.**

***Index Terms*—IoT, Fog Computing, Mobility, Energy, QoS, Optimization.**

I. Introduction

THE Internet of Things (IoT) can be defined as a self-adaptive and intelligent network of real-world smart objects connected via the Internet to provide various numerical applications [1]. Those applications have mostly stringent Quality of Service (QoS) requirements and their virtual components need to be deployed and managed efficiently through computing infrastructures. The mobility of the IoT objects is a critical factor that may impact negatively management decisions quality over time.

It has been shown that the exclusive usage of Cloud data centers can not support efficiently the gigantic number of IoT applications and their Quality of Service needs. Study in [2] estimates that the number of connected objects will reach 75 billions by the end of 2025. Authors in [3] present studies that highlight the limits of actual Cloud paradigm to support the needs of IoT applications response time. The Round Trip Time measures (RTT) between a data center located in Washington DC and a data center located in Salt Lake City via a fiber link is around 41 ms. This value can goes up to 120 ms with traffic contention. The RRTs from Salt lake City data center to popular public Clouds such as Amazon EC2, Google Cloud Engine and Microsoft Azure Cloud were respectively in average around 55 ms, 13 ms and 25 ms. Those latencies can't ensure IoT applications response time requirements where applications such as vehicles control or augmented reality streams require a response time less than 10 ms. Indoor geolocation and high speed trading applications require a response time around 100 μs.

Fog/Edge computing is an emergent computation paradigm aiming to support partial Cloud computation and storage workload at the edge of the network. It increases cloud data centers abilities in order to reduce excessive usage of the network, enhance privacy of users data and reduce applications response time. However, taking management decisions in such heterogeneous and large scale infrastructures is a complex task owing to the mobility of users, the high number and the heterogeneity of IoT applications and computation nodes [4]. Furthermore, the energetic impact of Fog and IoT infrastructures is huge. A report of the International Energy Agency (IEA) [5] estimates that the energy consumption of IoT applications in the fields of intelligent lights, traffic control, and smart homes will be around 46 TWh in 2025. Authors in [6] have found that, if unchecked, Information and Communications Technology (ICT) Greenhouse Gaz emission (GHGE) relative contribution could grow from roughly 1–1.6% in 2007 to exceed 14% of the 2016-level worldwide GHGE by 2040, accounting for more than half of the current relative contribution of the whole transportation sector. To ensure the sustainability of

Manuscript received January 8, 2020; revised March 12, 2021. Date of publication April 16, 2021. Date of current version April 16, 2021.

T. Djemai is with the University of Toulouse III and IRIT and LAAS-CNRS laboratories. P. Stolf is with the University of Toulouse II and IRIT laboratory. T. Monteil is with the INSA of Toulouse and LAAS-CNRS laboratory. J.M. Pierson is with the University of Toulouse III and IRIT laboratory.

E-mails: tdjemai@laas.fr, patricia.stolf@irit.fr, monteil@laas.fr, jean-marc.pierson@irit.fr.

Part of this work was presented at the International Conference on Software, Telecommunications and Computer Networks, SoftCOM '20, Hvar, Croatia, September 17-19, 2020.

Digital Object Identifier (DOI): 10.24138/jcomss-2020-0024

1845-6421/06/2020-0024 © 2021 CCIS

those environments, the energy parameter should be taken into account in every ICT management strategy.

We can split the life cycle management of any application in three phases: (1) Design, (2) Placement and (3) Maintenance. In this work, we investigate the placement phase of applications by considering the placement problem of IoT services in a Fog infrastructure as a multi-objectives problem to minimize both energy consumption of infrastructures and reduce applications Quality of service (QoS) violations by considering a dynamic environment driven by the mobility of IoT objects.

This work comes as an extension of a previous presented work [7]. The new contributions of this work can be summarized as follows:

- proposing a taxonomy of mobility models in the literature and a mobility model based on nodes locations history,
- proposing IoT services placement strategies that use the mobility information of nodes and can trigger migrations of services over time in order to improve the quality of the placement,
- experimental validation of the proposed mobility model and placement methods through simulations using MyIFogSim simulator.

The remainder of this paper is structured as follows: Section II presents the state of the art of IoT services placement in the Fog and mobility models. Section III gives a formalization of the considered Fog-IoT system and the placement problem. In section IV, we present the proposed mobility model used to estimate IoT objects locations. Section V details the proposed placement and migration framework and section VI reports experimental results using MyiFogSim. Finally, section VII concludes this work and gives some perspectives.

II. Related Work

The following related work is divided into two parts. The first part presents a brief overview and taxonomy about mobility models characteristics to position and understand the model proposed in section IV. The second part is about works from the literature that have tried to tackle IoT services placement in the Fog by considering jointly or separately one of those parameters: Energy, Quality of service (QoS), and Mobility.

A. Mobility Models Characteristics

A mobility model is used to describe the characteristics of a moving object or a group of moving objects. Generally, the model proposes descriptive formulas based on parameters such as the previous positions of the object, the instants of its positions, the time it spends at each location (pause time), its speed, its direction, and the impact of other objects on its movement. Mobility models can also be built from statistical inferences and use probability laws to describe and estimate the movement of objects. A mobility model can in some cases anticipate the paths that the object will take and plays a key role in the development and evaluation of decision models in dynamic environments such as IoT and Fog.

In recent years, there have been very few new models or proposed improvements to existing mobility models. Mainly, new models are proposed exclusively for connected vehicle systems. Work in [8] has shown that the mobility model can impact the evaluation of network and resource management strategies. Unrealistic mobility models can overestimate the performances of the proposed strategies.

A mobility model can be categorized according to several aspects (figure 1):

(1) The nature of motion data: several literature models can be decomposed according to their source data into synthetic, trace, or hybrid models. Works [8] and [9] have shown that trace-based models are more realistic, relevant, and less expensive than purely synthetic analytical models. Work in [10] presents a mobility model based on the social behavior of nodes, which is an important dimension regarding IoT environments. The disadvantage of trace-based models is that they can be applied and studied in limited scenarios and well-defined environments. This can easily render them unusable in other scenarios or other areas under study. Besides, statistical studies on traces carried out at a given point in time may become obsolete over time. To compensate for the above-mentioned drawbacks, the literature attempts to identify periodic behaviors that can be generalized from traces by varying environments and increasing the duration of data collection. The CRAWDAD project [11] proposes a wide variety of hybrid mobility models: synthetic models improved by real environments' data.

The mobility model proposed in this work is based on trace characterization to extract generic information. Other works have also carried out the same approach using traces. [12] focus on the mobility of cabs in the city of Beijing.

The majority of trace-based works are done on vehicle data such as cabs or buses because it is less complex than identifying patterns in passenger vehicles and IoT (for pedestrians) objects' movement. Authors in [13] focus on mobility behavior in highways by collecting information such as arrival and departure flows and vehicle density. Unfortunately, this study can not be used in urban areas. [14] has worked on urban area traces in the city of Cologne in Germany. It focuses on vehicle streets' density. [9] proposes a vehicle model in an urban environment by considering a macroscopic origin-destination approach. This method builds the model only with the departure and arrival points of the mobile nodes. [15] proposes a methodology to infer characteristics from a statistical study on a sample of data.

(2) The granularity of the model's parameters: Generally, there are two levels of granularity for mobility metrics: (1) microscopic and (2) macroscopic. Microscopic models give a detailed view of the movement of an object at any given time. These models, with fine granularity, take into account the speed and direction of each object. They allow to represent realistic and detailed mobile systems. This type of model is very present in Ad-hoc vehicle networks (VANET). Macroscopic models focus on more generic data, for example, the departure and arrival rates of objects in a given time interval or the probability distribution of flows from point A to point B. The macroscopic approach identifies points of interest that influence the mobility behavior of nodes. Both microscopic and macroscopic aspects are necessary to establish a realistic

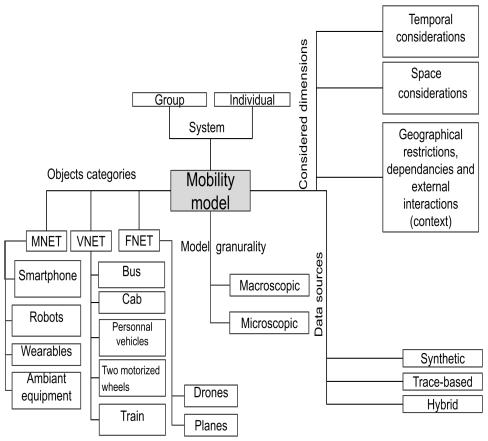


Fig. 1. Mobility models characteristics taxonomy

and comprehensive mobility model.

(3) Space and time dimensions: Mobility models can be classified according to temporal dependence, spatial dependence, or geographical restrictions. The work of [16] proposes a classification of individual and group mobility patterns for Ad-hoc networks. The latter groups them into random models, temporal correlation models, spatial correlation models, and geographic restriction models. However, today and particularly in Internet of Things environments, several models combine all the above characteristics. For IoT nodes' mobility, it is also important to consider the characteristics of IoT objects. Moreover, if these objects are carried by pedestrians, social behavior must also be considered.

(4) Mobile object classes: For the time being, we distinguish three main families: Vehicular Networks (VANET), Flying Ad-hoc Networks (FANET), and Mobile Ad-hoc Networks (MANET). [17] has proposed a selection of existing mobility models to best represent the systems of flying objects in a network. It is necessary to exploit the specificities of IoT objects and existing models to establish more precise models which can be generic and adaptive over time.

(5) The moving system under consideration: A mobility model can describe an individual movement or a group movement.

B. IoT Services Placement int the Fog with Mobility, Energy and QoS Considerations

Service placement problem in the Fog has been mainly investigated as an optimization problem in static environments (without nodes' mobility) to ensure various QoS criteria such as response time, resource usage efficiency, network consumption minimization [18], [19], [20], [21], [22]. The energetic problem has also been investigated in few works. In [23], authors try to minimize mobile nodes' energy consumption, time, and execution costs by transferring their tasks to complementary computation devices. This approach can induce excessive usage of bandwidth and increases the energy consumption of other nodes. Work in [24] address the IoT to Fog nodes assignment problem with Genetic Algorithm (GA) and Broadcast Incremental Power (BIP) algorithm to reduce mobile nodes energy consumption under delay constraints. The approach does not consider the energy consumption of Fog, cloud nodes, and the network. In [25], authors use a Discrete Particle Swarm Optimization approach to minimize the infrastructure energy consumption and applications delay violations in a static Fog environment.

Although mobility is a key point to consider in the Fog, only few works investigate it. Authors in [26] consider the influence of nodes mobility for applications' scheduling problem; however, mobility information is not used by the algorithms and the mobility scenario is only based on increasing and decreasing the number of connected nodes and does not consider any mobility model. Authors in [27] propose an Integer Linear Program to minimize the makespan while considering a random waypoint mobility model. This model neglects the temporal and spatial data. In [28], authors propose to dynamically re-schedule the placement of Virtual Network functions (VNF) to minimize end-to-end latency of users and the number of migrations. Compared to our work, this approach is only based on latency violation metric. None of these works introduces the mobility information in the placement modeling and algorithm.

The mobility aspect has been widely addressed in the field of Mobile Cloud Computation (MCC) through migration strategies, where most of the work tries to find a compromise between reducing the cost of the migration process and guaranteeing the quality of service requirements of applications/users often represented by latency. Most of the work identified is based on 1-D or 2-D Markov decision process models (CDM). The objective is to produce a set of migration decisions, taking into account the position of mobile nodes: [29], [30], [31].

Other works such as [32] use a theoretical approach to address the problem of long-term optimization at any given time with Lyapunov's optimization approach. These approaches are time and money consuming methods and are not appropriate to deployment on real systems.

The other category of approaches that could be used to address mobility is based on forecasting and learning methods.

Authors in [33] propose the SEGUE framework for migration decisions and use the ARIMA method to predict system QoS. In [34], authors propose a MAPE framework and a Genetic Algorithm for delay-triggered services migration in the Fog. This work focuses only on the delay metric and does not consider the energy cost of migrations. The main drawback of prediction methods is the high consumption of computing resources and the prediction time, which is difficult to apply online. In addition, the accuracy of the predictions is directly impacted by the quality of the data used. In [7], we proposed a placement strategy that minimizes the average energy consumption of infrastructures and QoS violations of applications through a certain time window in order to avoid the complexity induced by migration process. However, in some scenarios where services are state-full, migration is necessary. Moreover, static placement is not convenient in a long term as undeniably mobility through time will impact negatively applications response time and infrastructure's energy consumption. This work comes to extend the previous cited work by proposing a framework with fast placement

strategies and a replacement that will migrate services using a mobility-aware genetic algorithm. We also propose a mobility model in order to estimate future locations of IoT objects.

III. System Model and Problem Statement

In this part, we present the mathematical formulation of Fog infrastructures and IoT applications. Then, we state the multi-objectives placement problem, we present the formulas to estimate energy consumption, QoS violation metric, migration time and energy costs.

A. IoT Applications Model

We consider a set of IoT applications $\mathbb{A} = \{a_0, .., a_i, .., a_{A-1}\}$ designed in a distributed manner. Each application $a_i \in \mathbb{A}$ has a class type Υ_i which can be: Mission Critical (MC), Real Time (RT), Streaming (ST) and Best Effort (BE), having respectively priority level ψ_i of 0, 1, 2, and 3 and response delay requirement ϕ_i of 20 ms, 50 ms, 15 ms and 3000 ms.

An application a_i has a set of size n_i of communicating data-dependent services represented with a directed graph $\mathbb{G}_i = (\mathbb{S}_i, \mathbb{E}_i)$. Each service $s^i_j \in \mathbb{S}_i$ is defined by: (1) its requested CPU $inst^i_j$ in (MIPS), (2) its RAM ram^i_j in (MB), (3) its deployment technology tec^i_j (either a virtual machine, a container, OSGi plugin, Java Virtual Machine or a combination of them). We consider two specific nodes in the graph $\mathbb{G}_i$ that represent respectively sensor sending service (source node) and actuator receiving service (sink node).

Each directed edge $e^i_{jj'} \in \mathbb{E}_i$, from service s^i_j to service $s^i_{j'}$ represents a data dependency between s^i_j and $s^i_{j'}$ and carries $data^i_{jj'}$, the volume of data sent from s^i_j to $s^i_{j'}$ in (KB).

B. Fog Infrastructure Model

We consider a hierarchical Fog infrastructure described by a non directed graph $\mathbb{G} = (\mathbb{M}, \mathbb{L})$ and shown in Figure 2, where $\mathbb{M}$ is the set of nodes, and $\mathbb{L}$ the set of direct links between them.

Each physical node $m_k \in \mathbb{M}$ is defined by: (1) a category $\eta_k \in \{IoT, Fog, Cloud\}$, (2) a maximum and available capacities at instant t vectors $\Omega_k =< cpu^{max}_k, mem^{max}_k, disk^{max}_k >$, $\Omega^t_k =< cpu^t_k, mem^t_k, disk^t_k >$. Parameters units of elements are respectively in MIPS, MB and MB, (3) the electrical computing power at rest pc^{min}_k and its maximum dynamic power pc^{max}_k, which is the power consumed by the machine at the maximum of its CPU workload, (4) the electrical power of communication of its interfaces at rest pn^{min}_k and the maximum dynamic power pn^{max}_k, (5) a 2D coordinates and a coverage zone vector at time t, $\Delta^t_k =< x^t_k, y^t_k, r^t_k >$.

Nodes have additionally characteristics depending on their type: (1) IoT nodes have computation and storage capacities and a set of sensors $\mathbb{H}^0_k$ and actuators $\mathbb{H}^1_k$ components. IoT nodes move according to a probability mobility model ϑ_k (e.g. Manhattan model [35]), and have a set of requested applications $\mathbb{A}^t_k$ at time t. The coverage zone is fixed to $r^t_k = 0$.

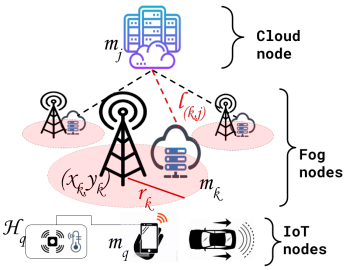


Fig. 2. Physical Topology representation highlighting the system's variables.

Let $\mathbb{U}$ be the subset of $\mathbb{M}$ with only IoT nodes, (2) Fog nodes have the ability to connect IoT nodes to external network through access point equipment and have computation and storage capacity through a server cloudlet device. (3) Cloud nodes are considered as a limitless source of computation with data centers nodes. The coverage zone is fixed to $r^t_k = 0$.

Each network link $ln_{kk'} \in \mathbb{L}$ between two nodes m_k and $m_{k'}$ has the following characteristics: (1) $nt_{kk'}$: represents link technology, (2) $bw^t_{kk'}$ representing the bandwidth at instant t which is defined by the number of bytes per second that a physical link is able to send from its source to its destination. It is measured in MB/s, (3) $lc^t_{kk'}$ representing the latency at instant t. It is measured in milliseconds (ms) and is the time elapsed between the sending and arrival of a data packet from the m_k machine to the $m_{k'}$ machine. The latency depends on the state of the network at the time the packet is sent.

C. Problem Statement

We consider the time window $W_{\mathbb{T}} = [0, T-1]$ of T homogeneous time slots and a set of mobile IoT nodes $\mathbb{U}$ requesting applications $\mathbb{A}^t$ at instant time t. Each IoT node $u \in \mathbb{U}$ has a set $\mathbb{A}^t_u$ of requested applications and $\mathbb{N}^t_u$ is the set of all services in $\mathbb{A}^t_u$.

The objective is to find a placement for IoT services that minimize energy consumption and delay violations over a certain time interval $[t_i, t_f] \subset [0, T-1]$.

We consider the binary decision variable x^{ut}_{ijk} defined by Eq (1) as follows:

$$x^{ut}_{ijk} = \begin{cases} 1, & \text{if the service j of application } i \text{ requested by node } u \\ & \text{is placed on machine } k \text{ at time } t. \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

For sake of simplicity but without loss of generality, we made the following assumptions on our system:

(1) The motion of a node is negligible during a time step i.e. the node can change position only at the beginning or at the end of a time step. (2) The duration of a service is longer than the duration of the study time window. (3) If two IoT objects m_u and $m_{u'}$ request an application a_i, each object will have its own instances of the services of a_i. (4) According to the specifications established for 5th generation networks, a user will always be covered by the network and will therefore always have access to a base station. From there, it is assumed that each IoT m_u object is connected by being attached to only one base station at a time and for each time step $t \in [0, T-1]$.

(5) Communications between IoT objects and their services are uniformly distributed over the $W_{\mathbb{T}}$ time window which is divided into T time slots.

In the following we present the formulas for energy consumption estimation, delay violations estimation, migration time of a service and energy consumed by the migration of a service.

1) Energy Consumption: The first metric we consider is the sum of computation and network energy consumption during time interval of $[t_i, t_f]$. As the energy model does not impact our resolution approach, we have chosen, similarly to work [36] a linear model but other models could also be used. The total energy consumption of the network and the computation is defined by Eq (2) as follows :

$$f_1(t_i, t_f) = f_C(t_i, t_f) + f_N(t_i, t_f) \tag{2}$$

1.1) The computation part is estimated according to the following equation:

$$f_C(t_i, t_f) = \sum_{t=t_i}^{t_f} \sum_{m_u \in \mathbb{U}} \sum_{a_i \in \mathbb{A}_u^t} \sum_{s_j^i \in \mathbb{S}_i} \sum_{m_k \in \mathbb{M}} x_{ijk}^{ut} \alpha_{ijk}^{ut} \gamma_k^c \tag{3}$$

where α_{ijk}^{ut} is the computation time of service instance s_j^i associated to the IoT user node m_u and placed on node m_k at instant time t. As the placement is made at the beginning and will remain unchanged, the computation cost is time independent. Placing service s_j^i on node m_k depends on the maximum processing capacity of the physical node cpu_k^{max} in MIPS, the amount of MIPS allocated to other services placed on the machine m_k and the amount of requested MIPS of service s_j^i. The computation of α_{ijk}^{ut} is defined by Eq (4) as follows:

$$\alpha_{ijk}^{ut} = \frac{inst_j^i + \sum_{u \in \mathbb{U}} \sum_{a_i \in \mathbb{A}_u^{t_i}} \sum_{s_l^i \in \mathbb{S}_i - \{s_j^i\}} inst_l^i x_{ilk}^{ut}}{cpu_k^{max}} \tag{4}$$

We define γ_k^c from Eq (3) as the computation power of the node m_k. It represents the difference between node maximum and minimum power consumption $\gamma_k^c = Pc_k^{max} - Pc_k^{min}$.

1.2) The communication part is estimated according to the following equation:

$$f_N(t_i, t_f) = \sum_{t=t_i}^{t_f} \sum_{m_u \in \mathbb{U}} \sum_{m_{k''} \in \mathbb{M} - \{m_u\}} P_{uk''}^t + \sum_{a_i \in \mathbb{A}_u^t} \sum_{s_j^i \in \mathbb{S}_i} \sum_{m_k \in \mathbb{M}} x_{ijk}^u x_{ij'k'}^u \beta_{ijj',kk'}^{ut} \gamma_{kk'}^n \tag{5}$$

where $\beta_{ijj',kk'}^{ut}$ defines the communication time between service instances s_j^i and $s_{j'}^i$ placed respectively on nodes m_k and $m_{k'}$ at time t for user node m_u. The communication time $\beta_{ijj',kk'}^{ut}$ varies over time, because it is impacted by the fluctuation of latency between the mobile node and its processing services placed on the infrastructure. There are two possible causes for this fluctuation:

1) The IoT node does not host services; then the latency variation is due to the communication between sensors and their destination services and actuators and their source services.
2) The IoT node hosts services; then the communication latency between those services and other services will vary over time depending on the localization of the IoT node's.

The computation of $\beta_{ijj',kk'}^{ut}$ is presented by Eq (6) as follows:

$$\beta_{ijj',kk'}^{ut} = \frac{1}{t_f - t_i} \frac{data_{jj'}^i}{bw_{kk'}^t} + lc_{kk'}^t \tag{6}$$

We define $\gamma_{kk'}^n$ from Eq (5) as the communication power constant between node k and node k' and is computed as follows: $\gamma_{kk'}^n = Pn_k^{max} + Pn_{k'}^{max} - Pn_k^{min} - Pn_{k'}^{min}$, where Pn^{max} and Pn^{min} are respectively the maximum and minimum power consumption of nodes network interfaces.

Considering a set of locations (point of interest or zones) and Access points nodes in a 2D area, a mobility model associated to an object u, P_{ur}^t is the probability that the object u will be in point of interest r at time t and it is deduced from its mobility model ϑ_u.

This work considers macroscopic mobility models with the Weighted Waypoints Model (WWP) [37] from the literature and a proposed History transition matrix model (HTM). However, even with microscopic models, it is possible to deduce macroscopic characteristics and then use the proposed formulas and placement approaches.

P_{uz}^t is the probability that an IoT node u is under the coverage zone of a Fog node z at time t and it is deduced by calculating the Euclidean distance between each point of interest and Access point. Then, under the assumption that each point of interest is attached to one and only one Access point, we associate the point of interest to their closest Access point.

2) Delay Violations : The second considered metric is the QoS violation metric defined in Eq (7) by the average number of applications delay violations in time interval $[t_i, t_f]$:

$$f_2(t_i, t_f) = \frac{1}{t_f - t_i} \sum_{t=t_i}^{t_f} \sum_{u \in \mathbb{U}} \sum_{a_i \in \mathbb{A}} w_i^u(t) \tag{7}$$

where $w_i^u(t)$ is the average sum of counted delay violations for each pair of object/application. It is defined by Eq (8) as follows:

$$w_i^u(t) = \begin{cases} 1, & \text{if } d_i^u(t) > d_i^{max} \\ 0, & \text{else} \end{cases} \tag{8}$$

For each application a_i and object m_u at time t, the estimated response time of application a_i for the object m_u corresponds to the time between the sending of one data packet from the sensor of node u and its corresponding action on the actuator.

The delay depends on the position of the IoT nodes and the placement of the used services. For each instance of the application $a_i \in \mathbb{A}$ requested by an IoT user node $u \in \mathbb{U}$, we compute the $d_i^u(t)$ according to the possible position of the user at time t as defined in Eq (9):

$$d_i^u(t) = \sum_{j\in\mathbb{S}_i}\sum_{k\in\mathbb{M}} x_{ijk}^{ut}\alpha_{ijk}^{ut} + \sum_{m_{k''}\in\mathbb{M}-\{m_u\}} P_{uk''}^{t} \sum_{j,j'\in\mathbb{S}_i}\sum_{k,k'\in\mathbb{M}} x_{ijk}^{ut}x_{ij'k'}^{ut}\beta_{ijj',kk'}^{ut} \quad (9)$$

The maximum response delay d_i^{max} of the application a_i varies according to its class type Υ_i.

3) Migration of a Service: In this section, we consider that the Fog infrastructure allows service migrations between nodes. We therefore propose formulas for estimating the time and energy consumed by the migration process. The formulas that we define thereafter are generic and independent of any technology, algorithm or migration strategy. We consider that all migrations start at the same time and run on a dedicated network.

a) Service Migration Time Estimation : We can intuitively deduce that the time due to the migration of a service depends mainly on its memory size and the bandwidth of the link connecting the source node to the destination node.

Several works in the literature [38], [39], [40] have proposed a linear model for estimating the migration time, considering the memory size of the service and the bandwidth of the transmission channel. Establishing a model for the power consumption of the migration is not part of the objective of our work. We use a high level generic model independent of the type of migration used and the nature of the services to be migrated.

Considering that a service s_j^i, placed on the physical node m_k, is going to be migrated to the node $m_{k'}$ at time t_r called revaluation time, the migration time $t_{mig}^{ij}(t_r)$ of this service is estimated according to Eq (10). We approximate the migration time with the transfer time, which according to several works in the literature represents most of the migration time. The migration time of a service is computed as follows:

$$\forall s_j^i, t_{mig}^{ij}(t_r) = \frac{ram_j^i}{bw_{kk'}^{t_r}} + lc_{kk'} \quad (10)$$

The migration time of an application a_i defined by t_{mig}^i is estimated by the maximum transfer time of its services. The maximum migration time of the services of a given application is the time when the whole application remains inaccessible and is defined by Eq (11). From this time, we can deduce the degradation of the quality of service. We recall that $lc_{kk'}$ and $bw_{kk'}^{t_r}$ respectively represent the latency and bandwidth of the link between nodes m_k and $m_{k'}$. The RAM requirement of the s_j^i service is defined by ram_j^i.

The total migration time t_{mig} defined by Eq (12) is the maximum value of application migration times and is computed as follows:

$$\forall a_i, t_{mig}^i(t_r) = \max_{s_j^i\in\mathbb{S}_i}\{t_{mig}^{ij}(t_r)\} \quad (11)$$

$$t_{mig}(t_r) = \max_{a_i\in\mathbb{A}}\{t_{mig}^i(t_r)\} \quad (12)$$

b) Service Migration Energy Consumption Estimation: Let $\mathbb{M}_{mig}$ be the set of source and destination machine pairs for the deduced service migrations between the placement solution at the moment $t_{r-\tau}$ and the new solution given at t_r.

The τ depends on the chosen revaluation strategy (it can be periodic or estimated by learning strategies or according to a certain management policy). It is assumed that the active power of the machines remains constant during the migration time.

Similar to works in the literature [41], we estimate the power consumed during the migration as the power consumed by the data transfer between the source and destination machines. We assume that the transfers between the source and destination machines of the different migrations are done in parallel. We define the total migration time t_{mig}^{max}, as the longest time taken between two machines.

Let $y_{kk'}^{uij}$ be a binary variable equal to 1 if the service s_j^i of the application a_i used by the object m_u migrates from the machine m_k to the machine $m_{k'}$ at time t_r and is equal to 0 otherwise.

The formula in Eq (13) presents the estimated energy consumption of the service migration triggered at time t_r and is computed as follows:

$$E_{mig}(t_r) = \sum_{(k,k')\in\mathbb{M}_{mig}} \gamma_{kk'}^n \frac{1}{bw_{kk'}^{t_r}} \sum_{m_u\in\mathbb{U}}\sum_{i\in\mathbb{A}_u}\sum_{j\in\mathbb{S}_i} y_{kk'}^{uij} ram_j^i \quad (13)$$

We recall that $\gamma_{kk'}^n$ is the power consumed for the exchange of one byte between machines m_k and $m_{k'}$.

The formula in Eq (14) defines the migration cost function at time t_r according to the consumed energy and the number of delay violations.

$$G(t_r) = f_3(t_r) + f_4(t_r) \quad (14)$$

We define respectively $f_3(t_r)$ and $f_4(t_r)$ as the normalized functions of $E_{mig}(t_r)$ and $t_{mig}(t_r)$. The functions $f_3(t_r)$ and $f_4(t_r)$ are computed in Eq (15) and Eq (16) as follows:

$$f_3(t_r) = \frac{E_{mig}(t_r) - E_{min}}{E_{max} - E_{min}} \quad (15)$$

$$f_4(t_r) = \frac{t_{mig}(t_r) - t_{min}}{t_{max} - t_{min}} \quad (16)$$

where E_{max} is the sum of the energy of all the machines at their maximum use during T time steps. E_{min} is the sum of the energy of the machines during a time slot when machines are without any workload. $t_{min} = 0$ is the time if no migration is triggered. t_{max} is set to the duration of the time window.

4) Objective Function: Considering that the placement remains the same between $[t_r, t_{r+\tau}]$ where $t_{r+\tau}$ is the known next re-evaluation time, the first function we consider is the combined sum of energy consumption and delay violation of the placement on the interval $[t_r, t_{r+\tau}]$ as defined by Eq (17) as follows:

$$F(t_r, t_{r+\tau}) = f_1(t_r, t_{r+\tau})[1 + f_2(t_r, t_{r+\tau})] \quad (17)$$

The objective function H is the sum of the function F in time interval $[t_r, t_{r+\tau}]$ and the migration cost defined by G at time slot t_r and is defined by Eq (18) as follows:

$$H(t_r, t_{r+\tau}) = F(t_r, t_{r+\tau}) + G(t_r) \tag{18}$$

IV. History-Base Transition Matrix Mobility Model (HTM)

With the advancement of location acquisition technologies such as the Global Position Systems (GPS), there is a growing need to analyze the huge amount of GPS data. Application management systems should be able to extract meaningful information from the huge amount of location data collected and stored. In this part, we propose a mobility model that can be applied to any type of mobile objects that can be geolocated on a 2D surface. This model uses the objects' position data. It is ideal in urban areas and smart city environments and is adapted for readjustment over time.

1) Model Principle: We propose a macroscopic mobility model. It is based on the assumption that dynamic human-driven environments such as smart cities are less subject to randomness and have recurrent mobility behaviors. Considering the power of connectivity and coverage provided by networks such as 4G-LTE and 5G, it is mostly superfluous to consider fine-grained mobility models that give precise geographical positions at any given time in order to provide the services demanded by mobile objects.

The proposed model determines the movements of classes of IoT objects according to time slots and their movements between well-defined regions within an observable mobility zone.

2) Methodology: We assume that we have an observable area of known size. We consider the smallest square that can contain the entire area. The main steps of the model are summarized as follows: (1) Define human-scale time intervals of a day divided into several time slots. (2) Subdivide the area into several small zones according to the specificity of the connection network. Each zone must be covered by at least one access point. (3) Create transition matrices for each moving object m_u and for each I_x interval of each observable d_x day of the data set. An element pz_{ij} of the transition matrix $MZ_{d_x}^{uI_x}$ thus gives the probability that an object moves to a zone z_j knowing that it is in the zone z_i. (4) Retrieve the macroscopic information of the zones: average density by object class and object flow. (5) Calculate and update the information of the objects and their user profile: the granularity and update of the mobility models of the objects is done according to their user profile. i.e. the type of applications they request and the frequency of requests per time interval during the day as well as the type or class of IoT objects. (6) Detect objects that have the same user profile and/or mobility behavior over the course of a day to reduce the number of matrices to be used for prediction.

3) Time Intervals and Mobility Zones: We consider a time window of 24 hours subdivided according to a typical day of activities into 4 time slots, where generally the mobility behavior of individuals doesn't know significant changes in each slot. $I_1 =]6am, 12pm], I_2 =]12pm, 6pm], I_3 =]6pm, 12am], I_4 =]12am, 6am]$. The times of greatest activity and mobility are considered to occur during the first 3 hours of each interval and behaviors are generally relatively different from one interval to another.

Considering the environment of a smart city, the last I_4 interval will generally present off-peak hours (with no application requests) that can be exploited to update the matrices and establish a new zone breakdown if necessary. The following classes of mobile IoT objects are considered: pedestrian objects (M1) (smartphone, wearable), vehicles (M2) and the rest of the objects (M3).

We consider at time t_0 the smallest square covering the geographical area under study z_0. In the initial state, the mobility area considered z_0 is subdivided into smaller areas $z_0 = \{z_{01}, ..., z_{0n(z_0)}\}$. The number of zones initially created $n(z_0)$ is calculated according to the smallest radio range (the smallest radius of coverage) $r^0_{min} = min_{k \in \mathbb{M}}\{r^0_k\}$ of the Fog nodes present in z_0 and the area of the zone z_0. The number of zones $n(z_0)$ is estimated by Eq (19) as follows:

$$n(z_0) = \frac{sup(z_0)}{(r^0_{min})^2} \tag{19}$$

A. Creation of Transition Matrices between Mobility Zones

Instead of having a central system that receives the history of node movements and zone information to compute the transition matrices of each object, it is proposed that the zone topology information is known by each zone and can be done through literature topology discovery methods such as [42]. Following this, each zone controller can communicate the topology information to the mobile nodes so that the mobile nodes can compute their transition matrix independently.

Similar to the computing infrastructure, a hierarchical control system is considered. If the mobile node does not have sufficient computing capacity, its position history is saved in its nearest Fog Gateway and its transition matrix is computed there. If the gateway does not have sufficient computing capacity, the process goes up to a higher level node (in our case the Cloud). It is assumed that the controller is chosen among the Fog nodes according to its computing power or its popularity (the greatest number of users connected during the day). We would like to draw the attention of the reader that consistency, message sequencing and synchronization in distributed systems does not falls in our work, so we consider an ideal situation where information arrives at the right time.

Assuming that the behavior of mobile nodes is relatively different between intervals, we need at least one transition matrix per time interval. To create the matrices, the history of the positions is browsed. Then, by considering each zone as an origin, we count the average number of times that every node was in the origin zone z_{0q} zone and moved to a zone z_{0p} within the corresponding time slot. Then, the values are averaged and the transition probabilities are deduced for each time interval. Figure 3 illustrates the matrices of an IoT object m_u for a day d_x.

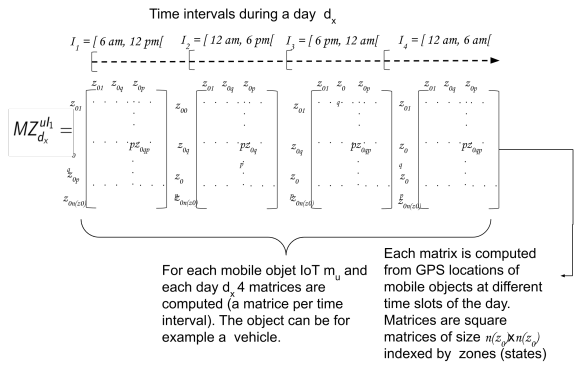


Fig. 3. Illustration of transition matrices for an object during one day.

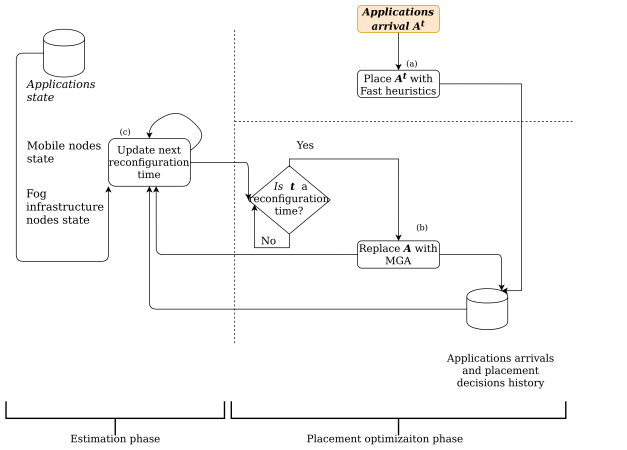


Fig. 4. IoT services placement Framework combing fast on-line placement heuristics and a Mobility-aware Genetic Algorithm for migrations.

B. Readjusting and Reducing Transition Matrix size

Starting from the principle that the environment is dynamic and large scale, in order to be able to use this method in a real system in a reasonable time, it is important to reduce the number of states (number of zones) of the transition matrices and/or the number of transition matrices, at the expense of the quality of the information and the accuracy of the displacement. This loss may not be important considering that with 5G networks the user will always have access to his services and the requested applications are not critical and do not have important QoS constraints (Best Effort applications).

We propose two approaches to reduce the number of matrices: (1) reduction by object class (M1, M2, M3)- the reduction of the number of matrices is done by averaging the matrices by object class, (2) reduction by user profiles (C1, C2, C3, C4)- this approach makes it possible to state whether it is important or not to lose precision for mobility information according to the category of application used. If the application is critical (MC), the matrices are not reduced (merged). If applications are Best Effort, the mobility information is less relevant then the number of matrices is reduced. This method is interesting when the usage of certain applications depends on the path that the mobile node will take.

The proposed mobility model has a number of advantages: (1) the model is easy to use and applicable to any geolocatable mobile node, (2) the model can be easily enhanced or customized by integrating additional matrix reduction policies, (3) if we have very specific models, we can always generate values from this model and switch back to this approach (with loss of information of course), (4) since this approach uses the notion of zones, it can be directly used by the service placement methods that we present in the section V.

The HTM mobility model also has certain limitations and drawbacks: (1) the method only proposes to reduce the number of matrices and not the number of states (zones), (2) reducing the number of matrices leads to a loss of information, which can degrade the quality of the model estimation, (3) the model considers only macroscopic mobility information, (4) the method generates additional communication costs due to the exchange of mobility information such as position history, (5) zones are created initially and statically according to the network characteristics of the Fog nodes.

V. Placement and Migration Strategies

In this section we present the proposed placement framework and the used placement strategies to solve the previous described problem.

A. General Placement Overview

Upon the arrival of an application set, the proposed framework uses a fast heuristic to place as fast as possible services. Then, at some specific moments, placements are reevaluated using a Mobility-aware Genetic Algorithm (MGA) and services can be moved from one machine to another via migration. The placement system is described in figure 4. Solutions are timestamped by the time step at which this solution was chosen. In addition, costs are calculated on subintervals that do not overlap in $[0, T-1]$.

B. Online Placement Heuristics

We propose three simple and fast heuristics for the online placement of services at the arrival of applications. Each heuristic considers one dimension of the problem to place services: applications classes, infrastructure computation state and users objects information: (1) Application Class Greedy Heuristic (AGH) considers the priority level of the application class for service placement as described in the Pseudo algorithm 1, (2) Infrastructure Greedy Heuristic (IGH), described in the Pseudo algorithm 2, places services according to the workload of physical nodes, (3) User based Greedy Heuristic (UGH) places services according to information about user nodes (IoT objects). The approach is presented by the Pseudo algorithm 3. The UGH heuristic uses the Round Robin principle with priority over all the IoT objects that have requested an application.

C. Placement Re-evaluation Strategies

1) The Mobility-aware Genetic Algorithm (MGA): Genetic Algorithm (GA) is an evolutionary based optimization approach, efficiently used for NP-hard problems resolution [43]. It has shown its efficiency in static Fog environments placement problems [19]. We propose to apply the GA method with

Algorithm 1 Applications Class based Greedy Heuristic

Sort the applications A^t by increasing order of their priorities.
Sort the services of each application by data dependency, then by increasing order of CPU demands.
while All applications $a_i \in \mathbb{A}$ are not placed **do**
  while All services s_j^i of the application a_i are not placed **do**
    Sort nodes by increasing order of their available computation capacities.
    If a_i is of priority 0 or priority 1, place all services in IoT nodes, then Fog then Cloud (first fit).
    If a_i is of priority 2 then place small services (according to CPU demand) in the Fog and bigger services in the cloud.
    If a_i is of priority 3 place all the services in the Cloud.
  end while
end while

Algorithm 2 Infrastructure State based Heuristic

Retrieve the state of the infrastructure, saved during the previous placement.
Select an application randomly from the inbound set and choose services randomly.
while All applications $a_i \in \mathbb{A}$ are not placed **do**
  while All services s_j^i of the application a_i are not placed **do**
    Sort the list of Fog nodes in ascending order of their available CPU capacity (Total CPU used / Maximum CPU of the machine)
    Place the service s_j^i in the first node with sufficient capacity.
  end while
end while

Algorithm 3 Users based Greedy Heuristic

Create application lists by application class.
Order IoT objects (users) in descending order of the number of applications in each list.
Let the current priority list be the highest priority list
while All non-empty applications lists **do**
  while All applications $a_i \in \mathbb{A}$ from the current priority list are not placed **do**
    Order the Fog nodes in ascending order of distance from the IoT object.
    while All services s_j^i from application a_i are not placed **do**
      Place s_j^i on the first node with sufficient computing capacity.
    end while
    Apply the RR selection on the current list to choose the next application to place.
  end while
  Let the current priority list be the next list.
end while

Algorithm 4 Mobility-aware Genetic Algorithm (MGA)

Retrieve the positioning probabilities for each mobile node $m_u \in \mathbb{U}$ at time $t_{r+\tau}$ according to its mobility model ϑ_u.
Uniformly generate the initial population size P_{size} and calculate the fitness of each chromosome defined by the sum of Eq (17) in time interval $[t_r, t_{r+\tau}]$ and Eq (14) at time t_r.
while The number of generations σ_3 is not reached **do**
  while The size of the next generation does not reach $\sigma_2 P_{size}$ individuals **do**
    On $(100 * \sigma_2)\%$ of the population apply 1-way tournament selection to select parents c_1, c_2.
    Apply Crossover on c_1, c_2 and get the chromosome c_3 by taking the first genes from c_1 and the rest from c_2.
    Apply the mutation on c_3 with the probability σ_1.
    Calculate the fitness of c_3 defined by Eq (18) and the mobility information part is detailed by Eq (5).
    Add c_3 to the next generation population.
  end while
  Keep $(100 * (1 - \sigma_2))\%$ of the best individuals in the population and inject them into the new generation.
  Increment the number of generations (μ).
end while

a probabilistic objective function enhanced by nodes mobility information. A chromosome c is a vector representing the placement solution.

Each vector element is indexed by the service Id s_j^i and gives the node Id m_k that will host the service. IoT users nodes are sorted by the increasing number of requested applications. The applications of each user node are sorted by the increasing priority values, and services of each application are sorted by increasing CPU demand. Through experiments we have chosen the following parameters values: mutation rate $\sigma_1 = 0.01$, crossover rate $\sigma_2 = 0.8$, population size $P_{size} = 20$, number of generations $\sigma_3 = 20$ and a 1-way tournament selection. The fitness of chromosomes is computed according to Eq (18) where the mobility model is used. For each user node u that requested an application at time slot t_0 and for each time step t of the time window, we get the mobility information of u. The mobility information P_{uk}^t is the probability that the user node u is under the coverage zone of Fog node k. MGA is presented in Pseudo Algorithm (4).

2) Re-evaluation Time Approaches: After the fast placement of applications by the heuristics, the placement is re-evaluated either periodically at each time step with the Periodic Mobility-aware Genetic Algorithm of periodicity equal to 1 (PMGA-1) or at times when the rate of application delay violations exceeds a certain threshold, variable according to their classes, with the QoS-Mobility Genetic Algorithm (QMGA). In our case, the violation rate of MC, RT, ST and BE applications has been set respectively at 10%, 30%, 40% and 100%.

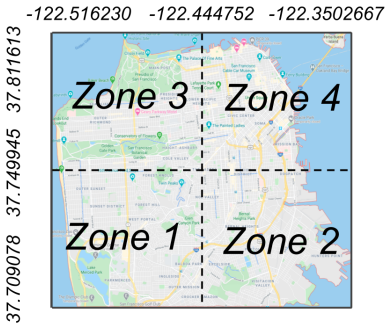


Fig. 5. City of San Francisco subdivided into four mobility zones used by the HTM mobility model.

VI. Results

In this section, we present validation experiments for the proposed mobility model and for the placement framework.

A. Validation of the Mobility Model

The model previously proposed in section IV is validated with a set of cab mobility data for the City of San Francisco provided by the CRAWDAD project [11]. Initially, we use these data alone. Then, we add synthetic pedestrian mobility data to test our matrix reduction strategies described in IV-B. As data is often vehicle-specific to validate our model, we use a synthetic model that groups three categories of IoT objects. The set of data collected in the city of San Francisco during 30 consecutive days [11] gathers the positions (longitude and latitude) of 500 cabs as well as their instantaneous speeds. The data are sent at different frequencies during the day. We add information on time slots, zones and user profiles to initial data. The z_0 zone of San Francisco represented by figure 5 has a $sup(z_0)$ area of $120km^2$ and is subdivided into 4 zones.

For these experiments, we considered the first 24 days of data to create the transition matrices between the zones and we estimated the positions of the mobile nodes over the last 6 days (convention of 80% of the data for the model and 20% for the tests [44]). The aim is to check whether the matrices produced represent the movements of the users over the remaining six days. Under the hypothesis that the movements are independent from one day j to one day $j+1$, the six remaining days are considered as 6 different instances of the 25th day.

Figure 6a shows the average success rate of estimating the positions of each vehicle over the last six days. The average success rate of the estimate over all 500 vehicles is 74%.

Figure 6b presents the average estimation success rate over the last 6 days by moving object and for the two matrix reduction strategies, object classes and user profiles. We notice that the reduction strategy by object classes gives better estimates.

B. Validation of the Placement Approaches

In this section, we present the performance of the placement strategies in a dynamic environment by considering the impact of (1) mobility, (2) the number of IoT objects and application classes.

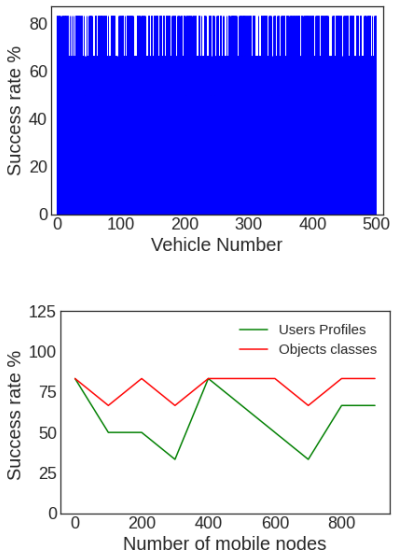


Fig. 6. Average success rates for positions estimations. a) Average success rate of vehicle position estimation per vehicle and for the last 6 days of the San Francisco data set. b) Position prediction success rate by object for the San Francisco dataset (with the addition of the synthetic pedestrian data) with matrix reduction.

The experiments were performed on an extended version of the MyiFogSim simulator [45]. We consider the scenario of the city of San Francisco where a fixed number of pedestrians and vehicles move within the z_0 zone. Pedestrian smartphones/wearables and vehicles are the two considered classes of mobile IoT objects in the system.

Pedestrians move between well-defined locations called "points of interest" according to the Weighted Waypoints mobility model (WWP) in the literature [37] and the model used to estimate vehicle movements is the previously presented HTM. The area is subdivided into 6 small zones, each containing a Fog node.

Each IoT object uses applications interacting with its sensors and actuators which can be of Mission Critical (MC), Real-Time (RT), Streaming (ST) or Best Effort (BE) class. The size of the studied area and the computing infrastructure gathering the Fog and Cloud nodes were determined. The time window is set to a 24-hour day with a time step of 15 min.

The proposed placement method is compared to: (1) Shortest Access Point migration Server Cloudlet (SAP) is a migration method whose initial placement strategy is to place object services on the closest Fog nodes if their capacity allows it. If capacity is insufficient, they are placed in the Cloud. Subsequently, at each time step and if there is a movement of objects, their services are migrated to the nearest Fog node if its capacity is sufficient, otherwise the services remain at their initial location. (2) K-Users Shortest Access Point migration Server (KSAP) consists of applying the same placement and migration approach as SAP on only K randomly selected mobile objects. The minimum value of the K parameter is 0 and gives a policy without migrations. The maximum value of K is the number of mobile objects and gives the SAP strategy.

After experiments with different values of K, we present the configuration that gives the most advantageous results for the method with K equals to 50% of the IoT objects. (3) Penguins Search Optimization Algorithm (PeSOA) [46] is a method inspired by the behavior of penguins searching for food in ice floes. The Penguin Search Aware Application Provisioning (PsAAP) method proposed in the literature for the placement of services in the Fog [47] aims to maximize the use of resources. We use this method with the same fitness as the MGA for placement system comparisons with reevaluation. The PeSOA method is known to be effective in dynamic environments [47], [46]. Our objective is to see if it would be more interesting to use another meta-heuristic for reevaluation knowing that until now this performance comparison has not been performed in the literature.

1) Impact of the Mobile Environment on Placement Methods: The objective of this part of the experiments is to see the impact of mobility on placement methods using the metrics defined above. The mobility environment here can be divided in two ways: (1) the "degree" of mobility of the environment, which determines in a global way between an environment with nodes that are often in movement against an environment where objects are much less active over time. (2) the mobility model used for pedestrians and vehicles.

a) Impact of Degree of Mobility on Placement Methods: In this sub-section, we will see the behavior of the methods according to the degree of mobility of the nodes.

For this, we consider three scenarios: the base scenario "normal mobility" where the IoT objects move according to a model at each time step, and two other scenarios, "fast mobility" and "slow mobility" where the objects move respectively twice as fast and twice as slow compared to the base scenario. The following parameters, which could impact the results and interpretations regarding the influence of mobility, were set as follows: (1) homogeneous mobility model by using only the WWP pedestrians model, (2) the number of IoT objects is fixed to 500 smartphones, (3) considering one application per IoT object where an application can have between 3 and 5 services, (4) applications arrival rate λ is following a Poisson law of parameter $\lambda = \frac{card(\mathbb{U})}{T}$ (T is the number of time steps and $card(\mathbb{U})$ is the number of IoT objects). This ratio implies that the average time between arrivals is $\frac{T}{card(\mathbb{U})}$, (5) the time window over a day is divided into 96 time steps.

Figure 7a shows the number of migrations for each degree of mobility and for each method. By observing the number of migrations of the methods from one scenario to another, we notice that the number of migrations of all methods is proportional to the degree of mobility.

In other words, the more frequently the nodes move, the more migrations occur. We notice that the number of migrations of SAP and KSAP methods is quite high whatever the scenario and that the difference from one scenario to another depends on the degree of mobility of the nodes.

The meta-heuristics Periodic Mobility-aware Genetic Algorithm of periodicity 1 (PMGA-1), QoS-Mobility aware Genetic Algorithm (QMGA) and Penguins Search Optimization Algorithm (PeSOA) remain quite stable. Even if the number of migrations increases from the slowest to the fastest scenario, this increase remains negligible compared to the KSAP and SAP methods. Considering mobility information, the methods place the services in such a way as to reduce the average cost of fitness Eq (18).

It can be noticed that the QMGA approach gives the lowest migration rate in the "slow" scenario compared to PeSOA and PMGA-1 approaches. However, the trend is reversed by increasing the degree of mobility. QMGA only starts if a certain threshold of time violation is exceeded. A faster environment leads to more trips and therefore more violations of sensitive applications.

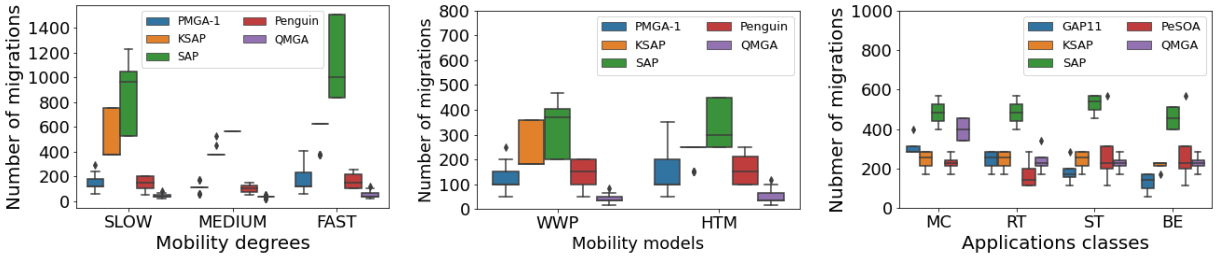
It is noticeable that even if the number of PMGA-1 and PeSOA migrations increases with the degree of mobility, this increase remains less important compared to QMGA. On average, the number of migrations of the PMGA-1 method is respectively 3% and 17% higher compared to the PeSOA and QMGA approaches. Finally, as a general remark, we can see that the average number of migrations triggered by meta-heuristics remains quite low compared to the total number placed in the infrastructure ($< 18\%$ of the total number of services).

Figure 7d presents the fitness values for each degree of mobility and for each method. It can be seen that the fitness values increase proportionally with the degree of mobility. It can also be seen that KSAP and SAP methods have the highest values. These methods have the only objective of bringing the services closer to the IoT objects and the large number of migrations they induce generates an additional energy cost. For meta-heuristics, we notice that the performances of QMGA and PMGA-1 are quite close. QMGA performs better in the "fast" scenario. For the other scenarios, PMGA-1 is more efficient. This can be justified by the fact that in a faster environment, QMGA is restarted more frequently (more violations) and at more relevant times than PMGA-1 (only if the threshold of time violations is exceeded). PeSOA performs worse than QMGA and PMGA-1 methods. PeSOA runs periodically like PMGA-1, so these results are the consequence of the exploratory behavior of the method, which is less efficient than the genetic algorithm. It is also noticed that PeSOA works better in slow environment compared to other methods that do not seem to be negatively impacted. On average over all scenarios, PMGA-1 reduces fitness by 61%, 48%, 15 % and 3 % compared to SAP, KSAP, PeSOA and QMGA methods respectively. The close results of the QMGA and PMGA-1 methods can be explained by the fact that PMGA-1 periodic triggering generates an additional energy cost due to a higher number of migrations, but it compensates by finding placement that reduce the average cost of energy consumption of the services. QMGA is triggered more rarely and therefore reassesses previous placement solutions less frequently.

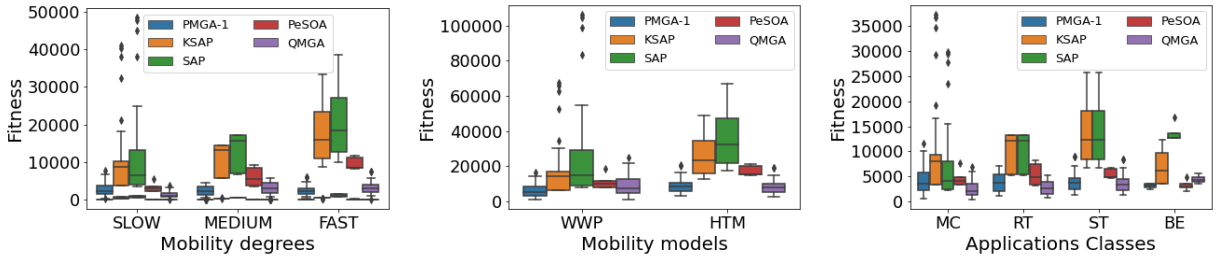
We can conclude from the previous results that the launching moments of the re-evaluation policy are crucial and must take into account the degree of mobility of the environment.

b) Impact of the Mobility Model on Placement Methods: In this sub-section we will see the behavior of the methods according to the node mobility model.

We place ourselves in the "normal mobility" scenario and



(a) Number of migrations per mobility degree (b) Number of migrations per mobility model (c) Number of migrations per application class



(d) Fitness values per mobility degree (10^3) (e) Fitness Values per mobility model (f) Fitness values per application class

Fig. 7. Migrations and fitness values for each placement/migration method considering: degrees and models of mobility and applications classes.

we consider the two models WWP (for pedestrians) and HTM (for vehicles) with 250 IoT objects and one application per object.

Figure 7b shows the number of migrations for each mobility model and for each method. It can be seen that the model does not influence the number of migrations regardless of the placement approach. We notice that for the two models used (WWP and HTM), the average number of migrations of the meta-heuristics is quite similar and that the trends of the methods remain the same from one model to another. It can be deduced that the model has little impact on the performance of the methods. The figure 7e presents the fitness value for each mobility model and for each method. We notice that the meta-heuristics QMGA and PMGA-1 are more performant with the HTM model. This can be justified by the advantageous breakdown of zones where we have one Fog node per zone and where in this scenario the HTM estimates were more accurate than the WWP. The PeSOA, on the other hand, gives better results with the WWP model. Performance comparisons between methods remain the same regardless of the model. On average PMGA-1 minimizes fitness by 2%, 18%, 42% and 67% compared to QMGA, PeSOA, KSAP and SAP methods.

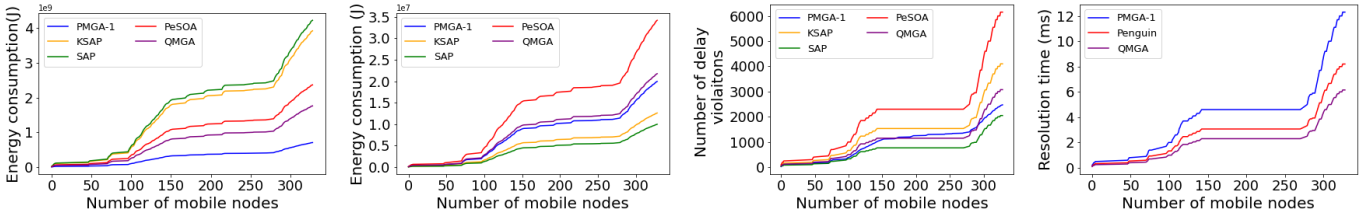
2) Impact of Application Classes on Policies: Figure 7c shows the number of migrations per application class for each method. We notice that the SAP and KSAP approaches have the same migration behaviors whatever the application class. For the rest of the methods, we see that the number of migrations of MC and RT classes is higher compared to ST and BE applications. We also see that it is more important for the PMGA-1 and QMGA methods compared to PeSOA method.

The figure 7f shows the fitness values according to the application classes. We notice that meta-heuristic methods are more efficient for the minimization of Best Effort (BE) classes which, having no strong QoS constraints, are more flexible for placement. We also notice that fitness values are more important for RT and ST applications which consume more computing resources and at the same time need to be close to the objects because of their response time constraint. This leads to a lot of migrations and service placements distributed between the Cloud (for computationally intensive applications) and between Fog/IoT nodes (to be as close as possible to the IoT nodes).

3) Impact of the Number of IoT Objects on Methods: Figures 8a, 8b and 8c show respectively the values of the computation and communication energy consumption and the number of delay violations by number of mobile objects for each method. We can observe that meta-heuristics are more efficient in minimizing computation energy consumption compared to KSAP and SAP migration strategies. PMGA-1 minimizes energy consumption by 83%, 71%, 37%, 14% compared to SAP, KSAP, PeSOA and QMGA approaches.

For the energy of communication, we notice that due to its larger number of migrations, PMGA-1 is the best placement approach compared to QMGA and PeSOA. PMGA-1 minimizes on average the communication energy by 3% and 29% compared respectively to QMGA and PeSOA methods while it increases the communication energy by 13% and 6% compared respectively to SAP and KSAP migration methods. It is noticeable that for the minimization of the number of time violations the PMGA-1 and QMGA methods have similar performance and are close to the performance results of the SAP method and are on average better than the KSAP method for this objective. PMGA-1 minimizes time violations by 11%, 0.5%, 41% compared respectively to KSAP, QMGA and PeSOA approaches while it increases the violations by 5% compared to SAP migration strategy.

Figure 8d shows the average method execution time as a function of the number of moving objects. It can be seen that the fastest approach is PeSOA, that is on average 3 times and



(a) Computation energy consumption (b) Network energy consumption (c) Number of delay violations (d) Resolution time

Fig. 8. Average values of energy consumption, delay violations and resolution times for each placement/migration method per increasing number of mobile nodes.

1.5 times faster than PMGA-1 and QMGA methods.

TABLE I
CHOICE OF STRATEGY ACCORDING TO EACH OBJECTIVE FOR VARIOUS MOBILITY DEGREES AND APPLICATIONS CLASSES.

	Mobility degrees			Applications classes			
Objectives	Slow	Medium	Fast	MC	RT	ST	BE
Energy	PMGA-1 > $QMGA$ > $PeSOA$	PMGA-1 > $QMGA$ > $PeSOA$	PMGA-1 ≈ $QMGA$ > $PeSOA$	PMGA-1 > $QMGA$ > $PeSOA$	$QMGA$ ≈ PMGA-1 > $PeSOA$	$QMGA$ ≈ PMGA-1 > $PeSOA$	$PeSOA$ > PMGA-1 > $QMGA$
QoS	$QMGA$ > $PeSOA$ > PMGA-1	$QMGA$ > PMGA-1 > $PeSOA$	$QMGA$ > PMGA-1 > $PeSOA$	$QMGA$ > PMGA-1 > $PeSOA$	$QMGA$ > PMGA-1 > $PeSOA$	$QMGA$ > PMGA-1 > $PeSOA$	-
Time	$QMGA$ > $PeSOA$ > PMGA-1	$QMGA$ > $PeSOA$ > PMGA-1	$QMGA$ ≈ $PeSOA$ > PMGA-1	$QMGA$ > $PeSOA$ > PMGA-1	$QMGA$ > $PeSOA$ > PMGA-1	$QMGA$ > $PeSOA$ > PMGA-1	$QMGA$ > $PeSOA$ > PMGA-1

VII. CONCLUSION AND FUTURE WORKS

In this work, we presented an extension of the model and the problem of placing IoT applications in the Fog by integrating the mobility of IoT objects. Then, we proposed a mobility model to estimate IoT objects future locations and a placement system integrating migration and mobility information. Finally, we presented experimental results that validate the proposed mobility model and the performance of the placement methods. Table I summarizes the experiments for the evaluation of placement strategies by comparing performances of PMGA-1, QMGA and PeSOA methods for the objectives: Energy, QoS and execution time and with different degrees of mobility and applications classes. The notation $x > y$ implies that on average the x method performs better than the y method and the notation $x \approx y$ implies that the methods are on average of equivalent performance. We notice that for fast environments and critical application classes the QMGA method is better for power and QoS. For slower environments, PMGA-1 is better for energy minimization. For non-critical applications the methods are more or less equivalent. Concerning the resolution time, the QMGA method is the least expensive, followed by PeSOA and PMGA-1. As future work, we plan to study other estimation and prediction approaches such as the integrated auto-regressive moving average methods (ARIMA), which are adequate for time series and which would improve the presented mobility model. We also plan to use learning approaches in order to reduce the costs of using placement methods and to estimate the times when migration or replacement of services is necessary.

ACKNOWLEDGMENT

This work is supported by neOCampus project, Toulouse university, France.

REFERENCES

[1] M. Rahmani Amir, L. Pasi, P. Jurgo-Soren and J. Axel. (2017). Fog Computing in the Internet of Things - Intelligence at the Edge. 10.1007/978-3-319-57639-8.

[2] L. Horwitz, "The future of IoT miniguide: The burgeoning IoT market continues", Cisco, 2019. [Online]. Available: https://www.cisco.com/c/en/us/solutions/internet-of-things/future-of-iot.html. [Accessed: 10- Oct- 2020].

[3] G. Ricard, "Application Latency", USignite, 2016. [Online]. Available: http://inlab.lab.asu.edu/nsf/files/NSF-workshop-2-Ricart.pdf. [Accessed: 10- Oct- 2020].

[4] F. Bonomi, R. Milito, Z. Jiang, A. Sateesh, "Fog Computing and Its Role in the Internet of Things," Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing, Association for Computing Machinery, Finland, 2012, pp. 13–16. doi: 10.1145/2342509.2342513.

[5] IEA, "Energy Efficiency of the Internet of Things, 4E EDNA Technology and Energy Assessment Report", IEA 4E EDNA, 2016. [Online]. Available: https://nachhaltigwirtschaften.at/resources/iea_pdf/reports/iea_4e_edna_energy_efficiency_of_the_internet_of_things_technical_report.pdf. [Accessed: 10- Oct- 2020].

[6] L. Belkhir, A. Elmeligi, "Assessing ICT global emissions footprint: Trends to 2040 & recommendations", Journal of Cleaner Production. 177. 10.1016/j.jclepro.2017.12.239, 2018.

[7] T. Djemai, P. Stolf, T. Monteil and J. -M. Pierson, "Mobility Support for Energy and QoS aware IoT Services Placement in the Fog," 2020 International Conference on Software, Telecommunications and Computer Networks (SoftCOM), Split, Hvar, Croatia, 2020. doi: 10.23919/SoftCOM50211.2020.9238236.

[8] R. Baumann, S. Heimlicher, M. May, "Towards Realistic Mobility Models for Vehicular Ad-hoc Networks", 2009. doi: 10.1109/MOVE.2007.4300807.

[9] A. Silva, A. Boukerche, R.M.B. Silva, L.B. Ruiz, A.F. Loureiro, "A novel macroscopic mobility model for vehicular networks", Computer Networks, Volume 79, 2015, Pages 188-202, ISSN 1389-1286, https://doi.org/10.1016/j.comnet.2015.01.003.

[10] M. Musolesi, C. Mascolo, "Mobility Models for Systems Evaluation", Middleware for Network Eccentric and Mobile Applications, 2009.

[11] D. Kotz, T. Henderson, I. Abyzov, J. Yeo, CRAWDAD dataset dartmouth campus, [Online]. Available: https://crawdad.org/dartmouth/campus/20090909. [Accessed: 10- Oct- 2020].

[12] C. Xia and D. Liang and H. Wang and M. Luo and W. Lv, "Characterization and modeling in large-scale urban DTNs", 37th Annual IEEE Conference on Local Computer Networks, 2012 , pp. 352-359, doi: 10.1109/LCN.2012.6423647.

[13] G. Füßler, M. Torrent-Moreno, M. Transier, R. Krüger, H. Hartenstein, W. Effelsberg, "Studying vehicle movements on highways and their impact on Ad Hoc connectivity", ACM SIGMOBILE Mobile Computing and Communications Review, 2006.

[14] S. Uppoor, O. Trullols-Cruces, M. Fiore and J. M. Barcelo-Ordinas, "Generation and Analysis of a Large-Scale Urban Vehicular Mobility Dataset," in IEEE Transactions on Mobile Computing, vol. 13, no. 5, pp. 1061-1075, May 2014. doi: 10.1109/TMC.2013.27.

[15] A. Law, D. Kelton, "Simulation Modeling and Analysis", 2000.

[16] F. Bai, A. Helmy, "Chapter 1: A Survey of Mobility Models in Wireless Adhoc Networks", 2011.

[17] K. Kumari, S. Maakar, "A Survey: Different Mobility Model for FANET", 2015.

[18] Q. T. Minh, D. T. Nguyen, A. Van Le, H. D. Nguyen and A. Truong, "Toward service placement on Fog computing landscape," 2017 4th

NAFOSTED Conference on Information and Computer Science, Hanoi, Vietnam, 2017, pp. 291-296, doi: 10.1109/NAFOSTED.2017.8108080.

[19] C. Canali, R. Lancellotti, "GASP: Genetic Algorithms for Service Placement in Fog Computing Systems" Journal of Algorithms, 2019.

[20] A. Brogi, S. Forti, C. Guerrero and I. Lera, "Meet Genetic Algorithms in Monte Carlo: Optimised Placement of Multi-Service Applications in the Fog," 2019 IEEE International Conference on Edge Computing (EDGE), Milan, Italy, 2019, pp. 13-17, doi: 10.1109/EDGE.2019.00016.

[21] M. Nardelli, V. Cardellini, V. Grassi and F. L. Presti, "Efficient Operator Placement for Distributed Data Stream Processing Applications," in IEEE Transactions on Parallel and Distributed Systems, vol. 30, no. 8, pp. 1753-1767, 1 Aug. 2019, doi: 10.1109/TPDS.2019.2896115.

[22] C. Guerrero, I. Lera, C. Juiz, "A lightweight decentralized service placement policy for performance optimization in fog computing ", Journal of Ambient Intelligence and Humanized Computing, 2019.

[23] L. Liu, Z. Chang, X. Guo, S. Mao, T. Ristaniemi, L. Liu, Z. Chang, X. Guo, S. Mao and T. Ristaniemi, "Multiobjective Optimization for Computation Offloading in Fog Computing," in IEEE Internet of Things Journal, vol. 5, no. 1, pp. 283-294, Feb. 2018, doi: 10.1109/JIOT.2017.2780236.

[24] A. Mebrek, L. Merghem-Boulahia and M. Esseghir, "Efficient green solution for a balanced energy consumption and delay in the IoT-Fog-Cloud computing," 2017 IEEE 16th International Symposium on Network Computing and Applications (NCA), Cambridge, MA, USA, 2017, pp. 1-4, doi: 10.1109/NCA.2017.8171359.

[25] T. Djemai, P. Stolf, T. Monteil and J. Pierson, "A Discrete Particle Swarm Optimization Approach for Energy-Efficient IoT Services Placement Over Fog Infrastructures," 2019 18th International Symposium on Parallel and Distributed Computing (ISPDC), Amsterdam, Netherlands, 2019, pp. 32-40, doi: 10.1109/ISPDC.2019.00020.

[26] L. F. Bittencourt, J. Diaz-Montes, R. Buyya, O. F. Rana and M. Parashar, "Mobility-Aware Application Scheduling in Fog Computing," in IEEE Cloud Computing, vol. 4, no. 2, pp. 26-35, March-April 2017, doi: 10.1109/MCC.2017.27.

[27] C. Mouradian, S. Kianpisheh, M. Abu-Lebdeh, F. Ebrahimnezhad, N. T. Jahromi and R. H. Glitho, "Application Component Placement in NFV-Based Hybrid Cloud/Fog Systems With Mobile Fog Nodes," in IEEE Journal on Selected Areas in Communications, vol. 37, no. 5, pp. 1130-1143, May 2019, doi: 10.1109/JSAC.2019.2906790.

[28] R. Cziva, C. Anagnostopoulos and D. P. Pezaros, "Dynamic, Latency-Optimal vNF Placement at the Network Edge," IEEE INFOCOM 2018 - IEEE Conference on Computer Communications, Honolulu, HI, USA, 2018, pp. 693-701, doi: 10.1109/INFOCOM.2018.8486021.

[29] A. Ksentini, T. Taleb and M. Chen, "A Markov Decision Process-based service migration procedure for follow me cloud," 2014 IEEE International Conference on Communications (ICC), Sydney, NSW, Australia, 2014, pp. 1350-1354, doi: 10.1109/ICC.2014.6883509.

[30] S. Wang, R. Urgaonkar, M. Zafer, T. He, K. Chan and K. K. Leung, "Dynamic Service Migration in Mobile Edge Computing Based on Markov Decision Process," in IEEE/ACM Transactions on Networking, vol. 27, no. 3, pp. 1272-1288, June 2019, doi: 10.1109/TNET.2019.2916577.

[31] A. Aissioui, A. Ksentini, A. M. Gueroui and T. Taleb, "On Enabling 5G Automotive Systems Using Follow Me Edge-Cloud Concept," in IEEE Transactions on Vehicular Technology, vol. 67, no. 6, pp. 5302-5316, June 2018, doi: 10.1109/TVT.2018.2805369.

[32] T. Ouyang, Z. Zhou and X. Chen, "Follow Me at the Edge: Mobility-Aware Dynamic Service Placement for Mobile Edge Computing," in IEEE Journal on Selected Areas in Communications, vol. 36, no. 10, pp. 2333-2345, Oct. 2018, doi: 10.1109/JSAC.2018.2869954.

[33] W. Zhang, Y. Hu, Y. Zhang and D. Raychaudhuri, "SEGUE: Quality of Service Aware Edge Cloud Service Migration," 2016 IEEE International Conference on Cloud Computing Technology and Science (CloudCom), Luxembourg, Luxembourg, 2016, pp. 344-351, doi: 10.1109/CloudCom.2016.0061.

[34] J.P. Martin, A. Kandasamy, K. Chandrasekaran, "Mobility aware autonomic approach for the migration of application modules in fog computing environment". Journal of Ambient Intelligence and Humanized Computing volume 11, 5259–5278, 2020.

[35] M. Alam, M. Sher and S. A. Husain, "Integrated Mobility Model (IMM) for VANETs simulation and its impact," 2009 International Conference on Emerging Technologies, Islamabad, Pakistan, 2009, pp. 452-456, doi: 10.1109/ICET.2009.5353127.

[36] A. Beloglazov, R. Buyya, "Optimal Online Deterministic Algorithms and Adaptive Heuristics for Energy and Performance Efficient Dynamic Consolidation of Virtual Machines in Cloud Data Centers", Concurrency and Computation: Practice and Experience, 2012.

[37] W.J. Hsu, K. Merchant, H.W. Shu, C.H. Hsu, A. Helmy, "Weighted waypoint mobility model and its impact on ad hoc networks," SIGMOBILE Mob. Comput. Commun,2002.

[38] A. Strunk, "Costs of Virtual Machine Live Migration: A Survey," 2012 IEEE Eighth World Congress on Services, Honolulu, HI, USA, 2012, pp. 323-329, doi: 10.1109/SERVICES.2012.23.

[39] Y. Kuno, K. Nii and S. Yamaguchi, "A Study on Performance of Processes in Migrating Virtual Machines," 2011 Tenth International Symposium on Autonomous Decentralized Systems, Tokyo, Japan, 2011, pp. 567-572, doi: 10.1109/ISADS.2011.79.

[40] F. Salfner, P. Tr, A. Polze, "Downtime Analysis of Virtual Machine Live Migration", The Fourth International Conference on Dependability, 2011.

[41] B. Yu, Y. Han, X. Wen and Z. Xu, "SMPA: An Energy-Aware Service Migration Strategy in Cloud Networks," 2016 IEEE 9th International Conference on Cloud Computing (CLOUD), San Francisco, CA, USA, 2016, pp. 984-989, doi: 10.1109/CLOUD.2016.0150.

[42] Cisco, Cisco Discovry protocol, 1994. [Online]. Available: https://www.cisco.com/en/US/technologies/tk652/tk701/technologies_white_paper0900aecd804cd46d.html. [Accessed: 10- Oct- 2020].

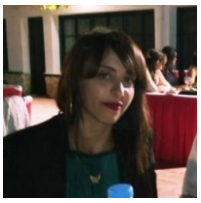
[43] X. Dai, J. M. Wang and B. Bensaou, "Energy-efficient virtual machine placement in data centers with heterogeneous requirements," 2014 IEEE 3rd International Conference on Cloud Networking (CloudNet), Luxembourg, Luxembourg, 2014, pp. 161-166, doi: 10.1109/CloudNet.2014.6968986.

[44] T. Nisonger, "The "80/20 Rule" and Core Journals", Serials Librarian - SERIALS LIBR, pp. 62-84 (55), 2008.

[45] M.M. Lopes, W.A. Higashino, M. Capretz, L.F. Bittencourt, "MyiFogSim: A Simulator for Virtual Machine Migration in Fog Computing," Companion Proceedings of the 10th International Conference on Utility and Cloud Computing, pp. 47–52, 2017.

[46] Y. Gheraibia, A. Moussaoui, "Penguins Search Optimization Algorithm (PeSOA)",Recent Trends in Applied Artificial Intelligence, Springer Berlin Heidelberg, pp. 222–231, 2013.

[47] A.R. Benamer, H. Teyeb, H-A. Nejib Ben, "Penguin Search Aware Proactive Application Placement",Algorithms and Architectures for Parallel Processing, Springer International Publishing, pp. 229–244, 2020.



Tanissia Djemai is a Ph.D. student from the University of Toulouse III (France). She is a team member of SEPIA team, in the laboratory of IRIT and SARA team, in the laboratory of LAAS-CNRS. She works on Internet of Things, Fog/Edge computing and, mobility issues.

Patricia Stolf is a researcher in the SEPIA team (Operating System, Distributed Systems, from Middleware to Architecture) of the IRIT Laboratory in Toulouse (France). She works in the field of grid computing, Cloud Computing, distributed autonomous administration and energy saving in large-scale distributed systems. She is the scientific leader of the SEPIA team since February 2020 and is an associate professor at the University of Toulouse II.

Thierry Monteil is a full Professor in the Electrical and Computer Engineering Department of INSA Toulouse (France), and a researcher at LAAS-CNRS in the Services and Architectures for Advanced Networks team (SARA). He is the moderator of the Computer Science and Networking Specialty Council.

Jean-Marc Pierson is a Full Professor in Computer Science at the University of Toulouse III (France) since 2006. He is the chair of the IRIT Laboratory and a researcher in the SEPIA team. His main interests are related to large-scale distributed systems. He serves on several PCs and editorial boards in the Cloud, Grid, Pervasive, and Energy-aware computing area.