Computer Communications 113 (2017) 43–52

Contents lists available at ScienceDirect

Computer Communications

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

Evaluating the benefits of combined and continuous Fog-to-Cloud architectures

W. Ramirez[*,a], X. Masip-Bruin[a], E. Marin-Tordera[a], V.B.C. Souza[a], A. Jukan[b], G-J. Ren[c], O. Gonzalez de Dios[d]

[a] Advanced Network Architectures Lab (CRAAX), Technical University of Catalonia (UPC), Spain
[b] Technische Universitat Braunschweig, Germany
[c] IBM, Almaden Research Center, USA
[d] Telefonica I+D, Spain

A B S T R A C T

The need to extend the features of Cloud computing to the edge of the network has fueled the development of new computing architectures, such as Fog computing. When put together, the combined and continuous use of fog and cloud computing, lays the foundation for a new and highly heterogeneous computing ecosystem, making the most out of both, cloud and fog. Incipient research efforts are devoted to propose a management architecture to properly manage such combination of resources, such as the reference architecture proposed by the OpenFog Consortium or the recent Fog-to-Cloud (F2C). In this paper, we pay attention to such a combined ecosystem and particularly evaluate the potential benefits of F2C in dynamic scenarios, considering computing resources mobility and different traffic patterns. By means of extensive simulations we specifically study the aspects of service response time, network bandwidth occupancy, power consumption and service disruption probability. The results indicate that a combined fog-to-cloud architecture brings significant performance benefits in comparison with the traditional standalone Cloud, e.g., over 50% reduction in terms of power consumption.

1. Introduction

The Internet of Things (IoT) embraces a large set of heterogeneous devices, demanding anywhere and anytime connectivity to run value-added services falling into distinct emerging domains, such as Intelligent Transportation Systems, E-health, or Smart cities, to name a few. Predictions available in [1] point out that near 26 billion of devices are to be connected in 2020, collecting more than 1.6 zeta bytes of data.

In order to address the ever-increasing demand for computing and processing information, Cloud computing is the major and widely adopted commodity [2], conceptually supported by its massive storage and huge processing capabilities. However, Cloud computing has shown limitations to meet the specific demands of IoT services requiring strict low latency, which can neither be overlooked in near-future IoT deployments, nor easily addressed with current network transport technologies. This in turn, has led to further innovations in the area of Cloud computing under the umbrella of Fog computing.

Fog computing leverages the capabilities of end devices –i.e., devices located at the edge of the network–, such as smart vehicles, 5G mobile phones, or autonomous IoT devices –e.g., smart sensors or wearables–, to enable service execution closer to IoT users. In this way, it is possible to reduce the overall service response time –suitable for real-time services–, reduce network congestion and energy consumption while addressing some of the cloud security gaps [3]. Nevertheless, as a novel technology, some major issues surround Fog computing, which if unaddressed may hinder its real deployment and exploitation as well as limit its applicability. We consider two major challenges to overcome: 1) the fog storage and processing capabilities are limited in comparison with the cloud, and; 2) the resources volatility, inherent to the mobility and energy constraints of fog devices, might cause undesired service disruptions. These challenges may undoubtedly hinder the adoption of Fog computing by the potential users, be it either traditional data center operators, ISPs or new actors such as smart city managers or smart transportation clients. Authors in [4] survey main research challenges for Fog Computing.

In order to take advantage of the benefits brought by Cloud and Fog, recent studies positioned Fog computing as a complementary solution to the cloud. Two main works may be highlighted, the Reference

* Corresponding author.
E-mail addresses: wramirez@ac.upc.edu (W. Ramirez), xmasip@ac.upc.edu (X. Masip-Bruin), eva@ac.upc.edu (E. Marin-Tordera), vbarbosa@ac.upc.edu (V.B.C. Souza), a.jukan@tu-bs.de (A. Jukan), gren@us.ibm.com (G.-J. Ren), oscar.gonzalezdedios@telefonica.com (O. Gonzalez de Dios).

http://dx.doi.org/10.1016/j.comcom.2017.09.011
Received 12 February 2017; Received in revised form 6 July 2017; Accepted 23 September 2017
Available online 25 September 2017

Architecture delivered by the OpenFog Consortium [5] and the Fog-to-Cloud (F2C) proposal [6]. F2C envisions a hierarchical resources architecture, consisting in a layered structure of heterogeneous Cloud and Fog resources working in a collaborative model under a coordinated and orchestrated management. The main aim of a F2C architecture is to efficiently provide enough capacity to execute services while guaranteeing low service response time, reduced network load and better energy efficiency.

In this paper we study the Dynamic Service Execution (DSE) problem in F2C scenarios. The DSE problem is defined as follows: given an IoT service with random arrival and holding times and with particular computing requirements, the objective is to discover and allocate the computing devices capable of meeting these requirements with the lowest cost resources. Indeed, the combined and continuous use of fog and cloud resources tailored to support the particular service demands, makes service allocation on such a distributed, dynamic and highly heterogeneous scenario a key challenge. Thus, the main rationale for this paper is threefold. First, we describe the main components of a F2C envisioned architecture through an illustrative example. Second, we put the focus on highlighting the benefits brought by deploying a combined F2C architecture, in terms of service response time, network bandwidth, energy consumption and disruption probability. Third, we also introduce two basic, easy-to-deploy service allocation strategies. For the sake of realism, the evaluated scenario considers fog device mobility to illustrate the dynamics and volatility associated with the overall fog computing capacity, also showing the impact rapid fog capacity changes may have on service execution. To the best of our knowledge, this is the first paper dealing with service allocation in dynamic F2C scenarios. We consider a dynamic scenario where service requests arrive in a random manner, as well as Fog nodes have mobility capabilities.

The rest of this paper is organized as follows. Section 2 introduces the concept of combined fog-to-cloud. Section 3 revisits existing contributions related to services allocation with special focus on fog computing. Section 4 describes the model adopted for the dynamic execution of services in combined F2C scenarios. Section 5 presents simulation results related to the overall F2C architecture performance. Finally, Section 6 concludes the paper.

2. Introducing the Fog-to-Cloud architectural concepts

This section is devoted to introduce the F2C management architecture as a potential approach to manage the combined set of fog and cloud resources. We show its potential benefits on an illustrative example and also discuss its main deployment challenges.

2.1. The building blocks

The F2C envisioned architecture is illustrated in Fig. 1. The whole architecture is based on two main management domains, layers and areas. On one hand, a layer is a set of devices with similar characteristics and features, such as processing capacity or mobility pattern – three layers are considered in Fig. 1, Cloud, High Capacity Fog (HCF) and Local Fog (LF). On the other hand, areas are a set of nearby logically and physically connected resources within the same layer.

Indeed, as shown in Fig. 1, F2C is a hierarchical layered architecture where fog and cloud resources are located in three distinct layers, "vertically" distributed according to their computing capacity, vicinity to the edge of the network and the amount of resources conforming the layer. For example, LF exhibits the closest proximity to the user, but lower aggregate storage, network and processing capacity than HCF. Fig. 1 also depicts the so-called Access Network, including the connectivity and devices on the user side to allow users to access the required resources to run a service.

Each area in the F2C envisioned architecture features the so-called Service Controller (SC), responsible for both gathering the data from the different IoT resources and maintaining the state of the IoT resources within an area. The state information gathered from SC members of the same layer is forwarded to the so-called Layer Controller (LC) for its management. LCs are responsible for allocating and releasing computing resources within a layer as well as for communicating with other LCs located at other F2C layers. The state information collected by the LCs is used by the so-called Service Computation Element (SCE) to provide service scheduling features. It must be remarked that an SCE is located at all areas/layers (Fig. 1 shows only one SCE due to readability purposes). Finally, when an SCE receives a service request, the SCE communicates to the LCs to start the service allocation process.

The Cloud layer is formed by dedicated static servers with on-demand computing features, such as processing and storage capabilities, accessible through the Internet backbone, providing nearly unlimited resources. As previously mentioned, the higher capacity is often provided at cloud assuming the cost of high access latency, whereas the opposite is the fog. In the architecture illustrated in Fig. 1, the upper layers have comparably, higher resource capacities, a longer service response time, as well as the expected higher energy consumption. At the same time, due to the vicinity between the access network and LF, workloads allocated within LF have lower service response time in comparison with either HCF, or Cloud layers. A clear example of an HCF layer can be set by aggregating resources in neighborhood fog areas, thus enabling a collaborative sharing with medium capacity and latency –e.g., vehicles in a parking lot sharing their computing resources with the data center in the shopping mall.

2.2. The context

Two inherent characteristics in the analysis of a combined and continuous F2C architecture, specifically linked to the F2C context and particularly impacting on its performance, must be observed, namely mobility and power consumption. The first, mobility of end-devices is a key issue highly impacting on resources volatility, hence with relevant effects in the quality perceived by the user running the service. Since it seems rather logical to assume that mobility grows as moving down on the F2C architecture, lower layers are expected to show higher volatility than higher layers. Let us illustrate mobility effects through a simple example, by considering a smart city scenario running a particular service S. We assume that, according to the resource allocation policy in use, service S is to be executed at resources within fog layer X, consisting in one single area (Fog-X). As shown in Fig. 2, Fog-X is formed by different devices, including a static traffic light as well as distinct mobile nodes –such as cars and buses. The volatility of the mobile devices included in the depicted topology is undoubtedly assessing the fact that the total Fog-X capacity will dynamically change over time according to the amount of nodes setting Fog-X. For the sake of comprehension, control plane entities, such as SC and SCE are not depicted in Fig. 2. Let us now assume the following: i) service S requires 3 units of computing resources to be allocated depending on real-time resources availability; ii) each individual device may only provide one resource slot, and; iii) the traffic light is supposed to be the node providing the computing resources. In such a dynamic scenario, a successful execution of a service relying on Fog-X capacity, will strongly depend on the real-time resources availability. Fig. 2a depicts a scenario where 4 slots, provided by three different cars and one bus, are available for service execution. Instead, Fig. 2b only shows 2 available slots, –i.e., 2 cars have already left the fog area. Thus, assuming 3 slots are required to execute service S, Fig. 2a stands for a successful execution, while Fig. 2b would not offer sufficient resources, hence turning into service disruption. Indeed, the latter may also occur when the node executing a job of a service leaves the fog (such as due to battery limitation, SLA policy, or similar).

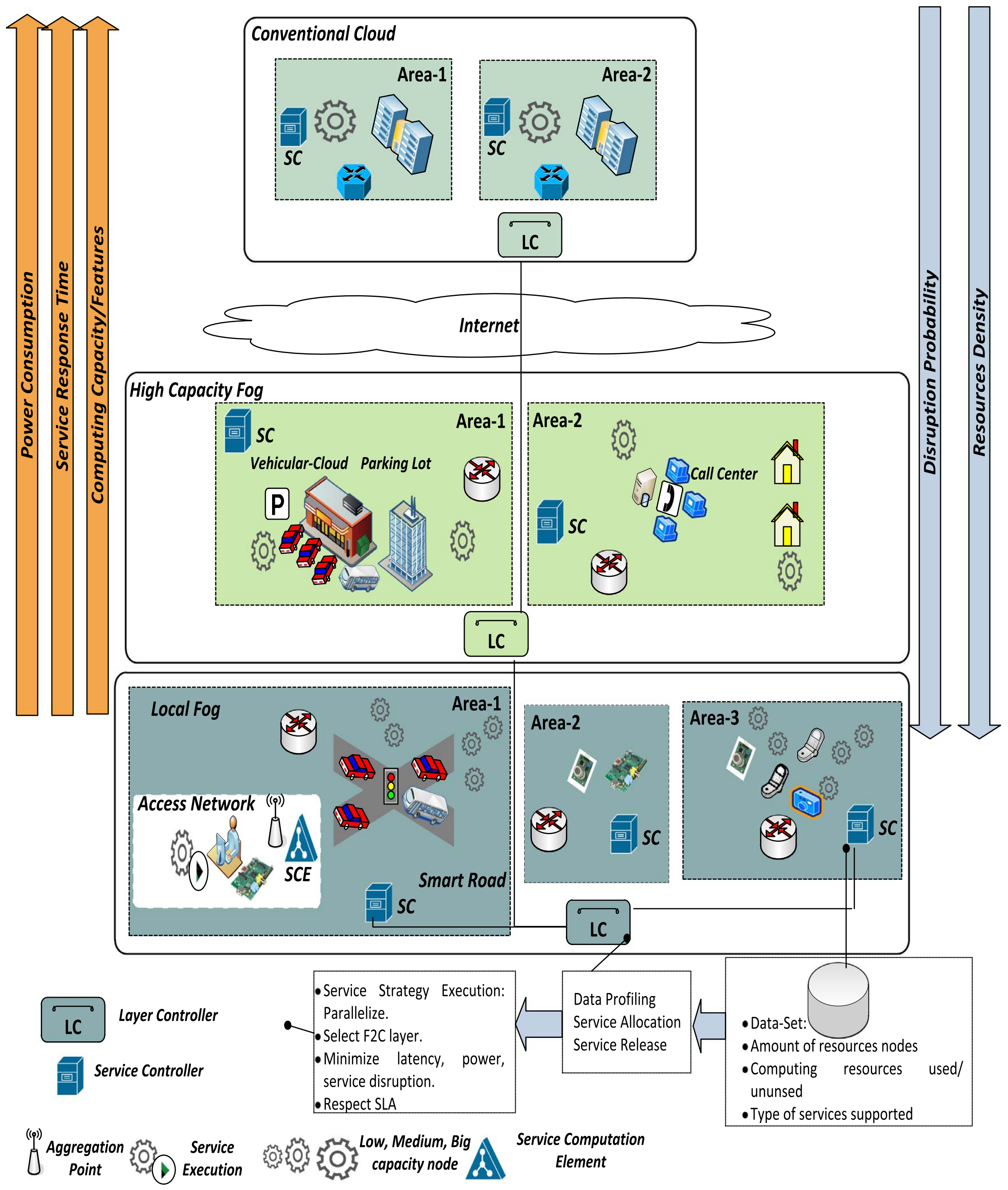

Fig. 1. Example of an F2C topology.

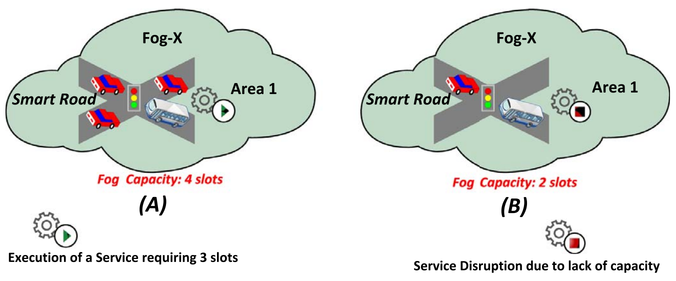
Fog-X
Smart Road
Area 1
Fog Capacity: 4 slots
(A)
Execution of a Service requiring 3 slots
Fog Capacity: 2 slots
(B)
Service Disruption due to lack of capacity

Fig. 2. Example of service disruption: a) successful execution; b) disruption.

3. Related work on service allocation in fog computing

As introduced in Section 2, F2C, as a potential strategy leveraging a coordinated management of fog and cloud resources, is expected to fuel the deployment of novel services through distributing the service execution through different layers and resources. Focusing on the service allocation arena, Section 2 also introduces the DSE problem, as a key issue to be fixed. While service allocation has been largely studied in cloud computing, in this section, we highlight the most recent contributions related to service allocation in Fog computing – which is currently more subject to research than deployment –, that may be certainly close to F2C.

As introduced above, unfortunately, to deliver on the promise of F2C architectures, it must be remarked that the design and evaluation of service allocation schemes is an issue not yet addressed. Among the research efforts in service allocation somehow related to combined F2C scenarios, we can mention the studies available in [7] and [8]. Both studies discuss the allocation of services in Fog and Cloud, with the emphasis on static service planning scenarios, that is the set of services to be allocated is known in advance. Authors in [9] describe the so-called FUSION framework and its architectural aspects regarding services orchestration and allocation in the fog (here fog is called “edge clouds”). This work is focused on data collection only, where the resources of fog devices are not shared with other fog nodes. This means that resources volatility or data quality are not considered. At this point the reader should notice that service orchestration refers to the actions required to determine both how the service is to be executed (for example be it in parallel or sequential) and the spectrum of computing resources required for the service execution. On the other hand, service allocation refers to the actions concerning the selection of computing devices matching the constraints imposed by the service orchestration and usually meeting policies in place, dealing with for example costs, etc. Recall that the latter is where this paper contributes to.

The work in [10] introduces a prediction strategy for the pre-allocation of resources in fog scenarios based on customer’s loyalty, measured by service relinquish probabilities. This study focuses on scenarios where customers have intermittent connectivity, rather than on mobility. The work available in [11] proposes a hierarchical 4-layer Fog computing architecture for big data analysis, specifically in smart cities scenarios. Finally, on the commercial side, just as recent solutions aimed at linking cloud and fog, we can mention IBM Bluemix Local, that extends the Bluemix product by considering not only IBM owned data-center premises, but also adding the data-center premises near the end-user side [12], or the AWS Greengrass by Amazon, also benefitting from local devices [13]. These notorious examples show the commercial appeal of combined cloud and fog strategies.

It is worth noticing that previous works focus on a set of issues related to the service allocation problem in either planning or single Fog scenarios. To the best of our knowledge, this is the first work considering mobility in combined Fog and Cloud architectures. This paper provides an architecture-level, though highly indicative, F2C performance evaluation, aimed at showing the benefits of a combined management of cloud and fog resources, in terms of service response time, power consumption, network bandwidth and disruption probability, all considered key for an IT/Network operator looking forward a real F2C-based architecture deployment.

It is important to remark that service disruption probability is an important performance metric for evaluating the impact devices mobility brings to the accuracy of the state information. Indeed, the vast and unstoppable population of potential Fog nodes along with their heterogeneity in terms of technologies, is adding more complexity to conduct a proper management of the state information. Therefore, novel updating policies must be considered to gather accurate information about computing (from the edge up to the cloud) node’s capabilities. These novel updating policies must target state information accuracy without neglecting the potential impact on bandwidth and memory space of the networking components. Thus, an updating mechanism, deploying the selected updates policies, will be responsible for disseminating the data and computing features offered by a resource. To that end a proper dissemination strategy must be designed to guarantee that the state information –available computing resources, including for example processing, disk storage, etc., or other resources characteristics, including for example energy constraints, switch on/off state, sleep mode, etc.– is suitably and accurately distributed on the F2C network. In this way, the service computation element can take accurate decisions concerning service allocation strategies. In this paper we do not particularly deal with update policies or dissemination strategies. Thus, in this paper neither the update policies nor the dissemination strategies are considered when analyzing the service disruption probability, instead service disruption probability is evaluated considering mobility patterns and fog nodes density.

Table 1
List of symbols and terminology.

Symbols and terminology	Meaning
S	Set of services or computing jobs.
X_i	State of a service i: 1 executed, 0 otherwise, $i \in S$.
u	Resource unit.
K_i	Number of tasks in service i that may run in parallel.
$W_{k,\ i,\ r}$	1 if the task k of service i is using a resource unit of layer r, 0 otherwise, where $r \in R$ and $k \in K$.
R	The set of F2C layers.
L_i	Service Response Time of service i.
b	Baseline power consumption.
a	Slope of load-dependent power consumption.

4. Dynamic service execution in combined Fog-to-Cloud scenarios

In this section, we focus on the Dynamic Service Execution (DSE) problem in the particular F2C scenario, and then propose two basic service allocation strategies, later used to evaluate the performance of an F2C architecture through four different parameters (see Section 5).

4.1. The DSE context

For the sake of global understanding, Table 1 lists the set of symbols used in this paper. The main rationale behind the DSE problem is to successfully execute a set of randomly arriving service requests, all demanding a certain amount of computing resources in the F2C system for a particular time. A service is defined as a computing job that can be split into K tasks, all running either sequentially or in parallel at fog or/and cloud premises. We assume a task requires a minimum of 1 resource (computing capacity or slot) unit, 1 u, be it CPU speed, memory or networking speed (see [14]). We also assume that a service will be successfully executed ($X_i = 1$) only if all its computing tasks (i.e., those the service is split into) are allocated either at cloud and/or fog layers, such as Cloud, HCF or LF layers, as illustrated in Fig. 1. Eq. (1) quantifies the *Success of Service Execution*, where $W_{k,i,R}$ defines whether a task k of service i is using 1 u of (e.g., Cloud, HCF or LF in Fig. 1) layer r, and K_i is the amount of tasks required by service i. It should be emphasized that tasks belonging to a same service do not need to be always allocated at the same layer, which is the salient feature of F2C systems.

$$X_i = \sum_{k \in K_i} \sum_{r \in R} \frac{W_{k,i,R}}{|K_i|}. \tag{1}$$

An F2C system increases the complexity of the DSE problem, since fog resources –located at the edge of the network–, may be moving constantly. This mobility effect, leads to computing capacity fluctuations, negatively impacting on the service execution.

4.2. The proposed DSE strategies

To analyze the performance based on the following four parameters, Service Response Time, Power Consumption, Network Bandwidth Occupancy and Service Disruption Probability, we propose two simple heuristics for job allocation: *First-Fit* and *Random-Fit*. In the First-Fit strategy, workloads are allocated with a Bottom-Up approach, hence when the bottom layer runs out of capacity, the next upper layer is considered for service allocation. A First-Fit approach prioritize the use of computing resources in the LF layer. In this way, the service response time can be substantially reduced since computing resources of the

```
Input: (service $i$)
Output: ($Allocated/NoAllocated$)
  for each task $j$ in $K_i$
  if LF has enough capacity then
    allocate task $j$ in LF
  else if LF does not have enough capacity, but HCF does then
    allocate task $j$ in HCF
  else
    allocate task $j$ in Cloud
```

Algorithm 1. Overall Procedure of First-Fit.

```
Input: (service $i$)
Output: ($Allocated/NoAllocated$)
  for each task $j$ in $K_i$
  Obtain $X$=Set of layers with capacity >0
  while task $j$ is not allocated and $|X| > 0$ do
    Randomly select a layer $r$
    if layer $r$ has enough capacity to support task $j$ then
      allocate task $j$ in $r$
```

Algorithm 2. Overall Procedure of Random-Fit.

Cloud layer are reserved for services requiring its high capacity resources.

On the other hand, in Random-Fit a layer is selected randomly for service allocation. Indeed, Random-Fit does not prioritize any layer for service computation, i.e., all layers have the same probability to be selected. In this way, Random-Fit is useful to find a balance between service response time and service disruption.

The pseudo-code description of both First-Fit and Random-Fit is described in Algorithms 1 and 2 respectively.

5. Evaluation performance

The main objective of this section is to clearly show the potential benefits brought by jointly managing cloud and fog resources, through the deployment of a novel Fog-to-Cloud management architecture enabling novel strategies, such as collaborative models based on resources sharing or parallel service execution, to name a few. To that end, in this paper, we propose to analyze the DSE performance in terms of four parameters, namely Service Response Time, Power Consumption, Network Bandwidth Occupancy and Service Disruption Probability. In the next paragraphs we describe each one of them.

Since all tasks of a service are executed in a parallel and independent manner, we define the *Service Response Time* (L_i) as the maximum latency among all its allocated workloads, c.f., Eq. (2).

$$L_i = max(L_r \times W_{k,i,R}). \tag{2}$$

Let L_i be the sum of the transmission latency and the processing latency. The transmission latency is defined as the round trip time required to obtain the information required for the service execution. It is worth stopping here to highlight the difference between the Service Response Time and the holding time. The Service Response Time refers to the elapse time to first, process a task and second, send the response back to the end user. Instead, the holding time is the time defining how long resources will be reserved for a particular service. In fact, a task might be completed, but still resources can be reserved for the exclusive use of a service. Therefore, a service will stop and its reserved resources will be released only when its holding time expires.

It is well known that the cloud layer is located farther from the user (access network) than fog layers, hence it typically exhibits higher transmission latency than any fog layer. The processing latency on the other hand, refers to the time needed to execute a service, which in turn depends on the processing capacities. While it is expected that a major difference between processing time at cloud and fog layers exists, that difference is not expected to be large when comparing different fog devices. Let L_i be expressed as shown in Eq. (2), where L_r is the transmission latency plus the processing latency of resource r.

The Power Consumption parameter can be generally determined as the addition of the energy consumed at fog and cloud premises. In this study, we only consider cloud Power Consumption, due to the fact that it is a dominant factor compared to mobile end-devices, and also because the power related consideration in the fog would require a slightly different approach – to consider for example the fact that end-devices may refuse the request for processing resources if their energy reserves are limited–, that is out of the scope of this paper. Thus, Eq. (3) shows the Power Consumption, where b is the baseline power consumption (assumed to be 325W as per [15]), a is the slope of the load-dependent Power Consumption (assumed to be 30.5 Joules per Gigabit), and x is the traffic rate.

$$Power = b + ax. \tag{3}$$

We also analyze the *Network Bandwidth Occupancy*, defined as the amount of traffic related to computing services processed in the cloud. Thus, the analysis of this parameter does not consider all traffic related to services executed at any fog layer. In fact, we pay attention to the traffic processed at Cloud, since this is the one that is conveyed along the backbone of ISPs –cloud's traffic consumes a high amount of Internet bandwidth which is a common concern among ISPs, since it can affect the Internet connectivity performance of their clients. On the contrary, the traffic processed on the Fog layer is usually conveyed on the access network. Hence, Internet bandwidth is not required. We consider the Network Bandwidth Occupancy a useful metric for IT providers, to measure the benefits brought by fog computing to reducing the network core traffic.

Finally, we define the *Service Disruption Probability* as the amount of disrupted services (when being executed), due to the lack of resources. This parameter considers the tasks that being allocated to a particular fog layer, the selected layer does not have sufficient resources to support the tasks demands. Therefore, a service is disrupted if at least one of the computing nodes assigned for service computation become unavailable (i.e., it runs out of capacity or it moves out of the layer coverage area), and no other computing node may be found in the same layer to execute the allocated tasks. For instance, if a service is being executed using two LF nodes, it will only be disrupted if during its execution the LF layer runs out of capacity. In addition, if a service is being executed using a computing node (A) belonging to the LF layer, and a computing node (B) belonging the HCF layer, this service will be disrupted if: i) node A becomes unavailable and there is no other node in the LF layer available, or; ii) if the node B becomes unavailable and there is no other node in the HCF layer available. In other words, we only consider protection between the same layers. Finally, it is also worth noticing the fact that we assume services running in the Cloud layer cannot be disrupted.

5.1. Evaluation settings and assumptions

In order to show the impact brought by considering the F2C envisioned architecture, we evaluate the effects of distinct DSE strategies on the Service Response Time, Power Consumption, Network Bandwidth Occupancy and Service Disruption Probability in three distinct and incremental F2C scenarios: i) a conventional Cloud scenario; ii) a Cloud and a LF architecture, hereinafter referred to as F2C-1, and; iii) Cloud, LF and HCF architecture, hereinafter referred to as F2C-2. The main objective of such a split is to show the effects of adding layers (i.e., resources) on the delivered performance.

The simulation model used to validate and obtain the presented simulation results can be found in [16]. The simulation model was built using the well known network simulation tool Omnetpp [17]. Moreover, all plotted values have a 95% confidence interval not larger than 0.5 of the plotted value.

The following assumptions apply to the adopted simulation model:

- A resource unit consumes 10 Mbits of networking bandwidth (independently of the layer where the resource unit is allocated). Moreover, a workload requires one unit (slot) of capacity (memory, storage, processing). It must be remarked that a slot is consumed by a service while its holding time does not expire. The reader should recall that the holding time refers to the time that a service will keep consuming computing and networking resources, independently if a service's task finishes its execution.
- 90% of services requests are mice services. A mice service stands for a service that consumes a minimum amount of slots and whose arrival and holding times are low – web search is a common example of a mice service. The remaining 10% of the services are elephant services. An elephant service requires a high amount of slots (ten times more than a mice), and its holding time is also larger in comparison with a mice service – video on demand and file transfer backup are common examples of elephant services [18].
- Mice service requests arrive randomly with a Poisson distribution, with a mean arrival time of 10 time slots. The holding time of a mice service follows a negative exponential distribution of 10 time slots on average. A mice service requires an average of two slot units, i.e., a mice service can be divided into 2 independent tasks thus easing

parallel execution.

- Elephant service requests arrive randomly following a Poisson distribution, with a mean arrival time of 100 time slots. The holding time of an elephant service follows a negative exponential distribution of 500 time slots on average. We assume that an elephant service requires an average of 10 units of capacity.
- The LF layer consists of 10 mobile nodes. Each LF node has 2 units of capacity. The policy for a mobile node to get in/get out the LF, is random according to a Poisson and negative exponential distribution respectively, with an average time of 50 time slots.
- The HCF layer consists of 2 mobile nodes. Each HCF node has 20 units of capacity. The policy for a mobile node to get in/get out the HCF is random according to a Poisson and negative exponential distribution respectively, with an average time of 500 time slots.
- L_r values per slot for Cloud, HCF and LF layers are 10, 2 and 1 ms respectively.

It is worth remarking that the main rational driving the adoption of 10 LF nodes and 2 HCF nodes, for the local fog and the high capacity fog layer respectively, is to consider edge devices heterogeneity. As well as other related research studies, see [10,19,20], we consider the fog layer closest to the edge of the network has the highest amount of computing nodes with limited capacity. This capacity limitation is because Local Fog devices are edge devices (e.g., mobile phone or smart sensors) that do not have high computing capacity. In a similar manner, we consider that HCF nodes are commonly desktop or mobile PC, which despite of not having a computing capacity similar to the Cloud, they do have more capacity than LF nodes.

5.2. Threats to validity

In the following lines we present extensive simulation results concerning the performance of Cloud computing and two distinct F2C architectures, considering two heuristics for service allocation. Four metrics are considered to evaluate the performance: the Service Response Time, Power Consumption, Network Bandwidth Occupancy and Service Disruption Probability.

To properly support the simulation results related to Service Response Time, we consider distinct LF and HCF settings in order to show how service requests are allocated at the Cloud layer. Since the main goal of the Service Response Time evaluation is to show how Fog resources can reduce stress from the Cloud, in order to assure the validity of this assessment, we plot the amount of services using Cloud computing resources versus the population of HCF and LF nodes for F2C-2 and F2C-1 architectures respectively, see Figs. 3 and 4. It is important to remark, that the last two evaluations do not consider the

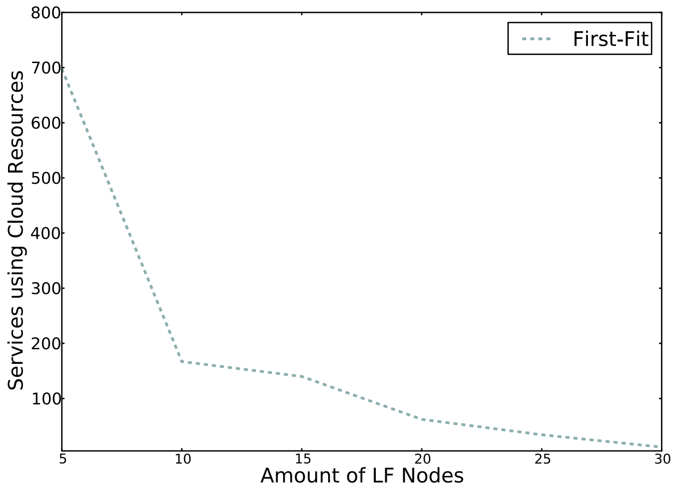

Fig. 3. Amount of services using Cloud resources vs LF nodes population.

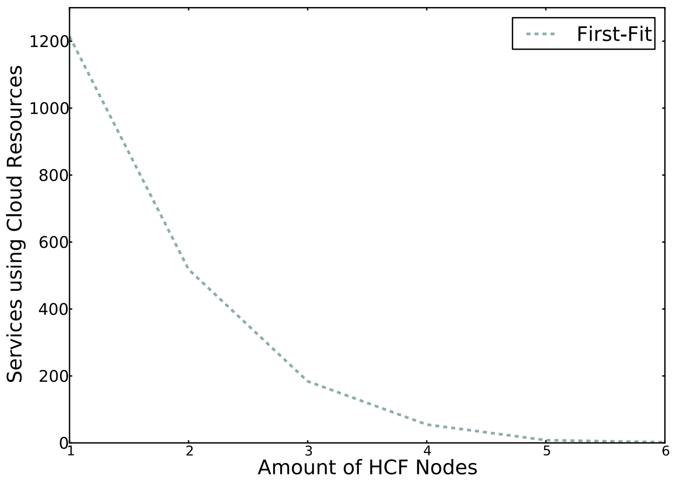

Fig. 4. Amount of services using Cloud resources vs HCF nodes population.

random-fit heuristic, since its uniform random selection strategy makes no significant variation to come up related to the service request distribution.

Based on the results shown in Figs. 3 and 4 we can see that indeed, the use of both LF and HF nodes reduce stress from the Cloud. However, the reduction degree is limited by the amount of Fog nodes available as well as the issues related to service disruption caused by the overuse of Fog nodes, see the section concerning Service Disruption Probability.

Another important performance metric is Service Disruption Probability, which measures the probability of a service being disrupted when the mobility of fog computing nodes is considered. Based on the mobility pattern assumed, LF nodes move faster than HCF nodes. Therefore, we clearly show in sub-section 5.6 that F2C-1 presents a higher disruption probability than the other architectures evaluated. To assure the validity of our simulation results, we evaluate the disruption probability of an F2C-2 architecture, with different layer settings, specifically by increasing the amount of LF nodes available. In this way, we confirm our assumption stating that the intrinsic mobility of Fog nodes can negatively affect the performance of F2C architectures. In short, the more the LF nodes available, the better the Service Response TIme (typically the case in IoT scenarios), although it can be counterproductive regarding service resilience.

5.3. Service response time

Fig. 5 shows the Service Response Time for the following

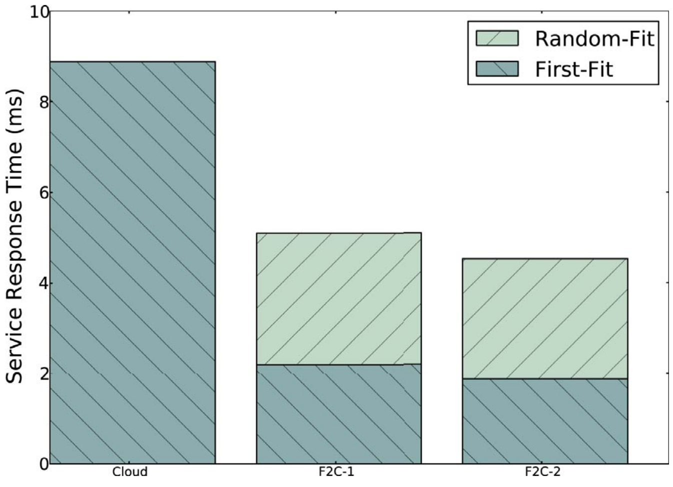

Fig. 5. Average service response time.

architectures: Cloud (C), F2C-1 (LF and Cloud, i.e., one Fog layer) and F2C-2 (LF, HCF and Cloud, i.e., two Fog layers), considering the two proposed strategies for job allocation without Fog nodes mobility capabilities, i.e., Random-Fit and First-Fit. As it can be observed, the conventional Cloud scenario exhibits the highest Service Response Time. Indeed, services with all their tasks allocated at Cloud present a substantial impact on the Service Response Time, mainly motivated by the long geographic distance between Cloud and the edge, what unquestionably, negatively impacts on the time required to acquire the information processed by the cloud layer. It is also worth mentioning that in a cloud scenario, the Service Response Time is not affected by the heuristic used for jobs allocation. The simulation results obtained for F2C-1 show a significant reduction of the Service Response Time in comparison with the conventional Cloud. This is motivated by the geographical proximity between fog servers and the edge. However, the best results are obtained for F2C-2. We clearly show in Fig. 5 that the simulation results for F2C-2 present the lowest Service Response Time vs both F2C-1 and the Conventional Cloud. Indeed, this reduction is motivated by the use of an additional fog layer (HCF). In a deeper analysis we may conclude that some of the jobs allocated to the Cloud layer in the F2C-1 scenario (there are no resources enough at the LF layer) are now executed at the additional fog layer, namely HCF –with a lower response time vs cloud–, so reducing the global Service Response Time.

It is also worth analyzing the effects of the heuristic used for jobs allocation for each individual F2C-1 and F2C-2 scenario. We may see that First-Fit has lower response time in comparison with Random-Fit. This is mainly because First-Fit intends to allocate all tasks close to the edge. Thus, First-Fit considers first the LF layer, and only in case of lack of capacity in the LF layer, the HCF layer is considered, and finally the Cloud only when HCF lacks of resources. However, Random-Fit selects in a random manner the Cloud, HCF or LF layers for each task of a service. As shown in Fig. 5, this strategy negatively impacts on the Service Response Time.

5.4. Load-dependent power consumption

Fig. 6 shows the total Power Consumption for the three different scenarios, Cloud, F2C-1 and F2C-2, also considering both Random Fit and First-Fit job allocation strategies. It should be noted that the plotted graph stands only for the power consumed by the Cloud. This is because, as said in the previous section, the power consumed by the Fog (mainly formed by mobile devices) can be neglected in comparison with the Cloud (in the current model we do not consider mini data centers as part of fog devices). Hence, the Power Consumption is directly aligned to the jobs allocated to the Cloud. As it can be observed and also as expected, the conventional Cloud exhibits the highest total power – notice that, similar to the analysis done for the Service Response Time in a cloud scenario, the power is not affected by the heuristic used for job allocation. Simulation results for F2C-1 show a significant reduction of the total power in comparison with the conventional Cloud, that is even improved when deploying the F2C scenario. The final conclusion is that by adding a new fog layer we reduce the need to allocate jobs at the cloud –reacting to a lack of resources at the LF layer–, hence reducing the overall Power Consumption. Finally, notice that similar to the Service Response Time, the First-Fit allocation strategy presents lower Power Consumption compared to the Random-Fit strategy.

5.5. Network bandwidth occupancy

A conventional Cloud infrastructure is commonly reached out through the network core of the Internet Service Providers (ISPs), that are responsible for providing the network resources required to accommodate the traffic generated by the services executed by cloud users. Recognized the fact that such traffic cannot be neglected –for instance the traffic collected from thousands of sensors or the data obtained from a huge database–, it turns out that this traffic may be large enough to significantly stress the ISPs network core. In response, ISPs must re-dimension their networks to handle the traffic generated by cloud users, which increases their CAPEX and OPEX. Fortunately, the advent of fog opens the door to ISPs as a cost-efficient solution to reduce the traffic conveyed to their network cores. In the following lines, we evaluate how efficient traffic offloading to the edge may be in this regard.

Fig. 7 shows the Network Bandwidth Occupancy for both First-Fit and Random-Fit strategies, considering the F2C-1 and F2C-2 scenarios –the Cloud scenario is not considered since in this case all traffic has moved to the network core. It is worth highlighting the reduction obtained with the introduction of the HCF layer (i.e., the F2C-2 scenario). Indeed, the total traffic conveyed to the network core is 46.8% and 28.1% for F2C-1 and F2C-2 architectures respectively, using Random-Fit. For a First-Fit strategy, the total traffic generated at the network core is even lower, 14.6% and 10.5% for F2C-1 and F2C-2 scenarios respectively. Based on the obtained results, we may conclude that a Random-Fit strategy might be more appropriate to achieve a correct balance between the network traffic conveyed at the network core, and the one conveyed at the access domain (fog layer). An important aspect to be considered is the sharing policy supporting the deployment of the intermediate fog layers. Indeed, some policies –similar to the traditional Service Level Agreement though much more dynamic–, must be deployed to enable edge resources sharing. These policies will undoubtedly impact on the overall performance. For example, we may

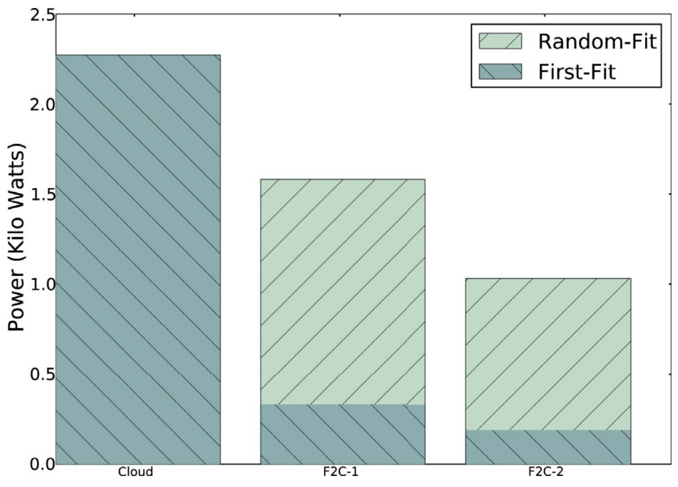

Fig. 6. Total power consumed.

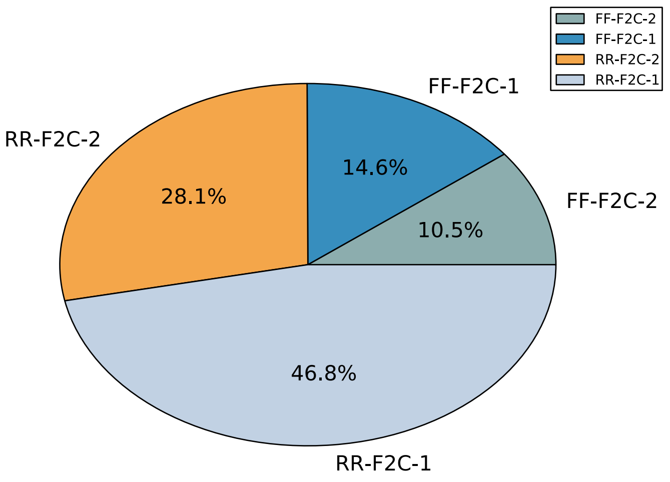

Fig. 7. Traffic generated at the network core.

with no doubt assess that an end user will only share its resources as long as no degradation is produced in the perceived quality of the services already running in his/her device. Also worth mentioning that such sharing policies will be a key part of the novel collaborative model envisioned (similar to AirBnB, etc.) with a relevant impact on the business model for Cloud and ISP providers. This paper does not deal with such policies, leaving this analysis for future studies.

5.6. Service disruption probability

The Service Disruption Probability refers to the amount of disrupted services due to the unavailability of resources caused by the intrinsic mobility of Fog nodes. The reader should recall that the mobility of Fog nodes might cause that a node loses connection to its access point, hence the fog node no longer being part of its Fog layer. This has a final impact on the services running on the fog node.

To compute the disruption probability, we have adapted the Freeway mobility model as the mobility pattern for the LF domain presented in [21]. To that end, we consider a three lanes highway with 10 LF nodes. Every time a LF node exits a highway lane a new node enters the highway with a mean Poisson arrival time of 6 time slots. We also consider a lane length of 1 Km and nodes traveling the lanes with a speed range between 50 and 80 Km per hour. We also consider a parking lot model to model the mobility pattern of the HCF layer [22], where cars in a parking lot form the HCF layer. Similar to [22], we consider that HCF nodes arrive at a parking lot according to a Poisson process, as well as an exponential distributed parking time with 15 and 60 time slots respectively.

Fig. 8 shows the Service Disruption Probability for both F2C-1 and F2C-2 scenarios –we do not consider the Cloud scenario, since we assume Cloud resources are never exhausted as well as cloud nodes are not moving. Notice that unlike the three parameters evaluated so far, the benefits brought by F2C scenarios are not that significant when analyzing the Service Disruption Probability. This is obviously motivated by the fact that no disruption is expected when allocating jobs only at Cloud premises. The Service Disruption Probability of F2C-2 is lower than F2C-1, whereas the performance of First-Fit and Random Fit is very similar, hardly affecting the overall performance. This is due to the fact that by adding a new fog layer we increase the chances to prevent service disruption. Therefore, it can be concluded from the results shown in Fig. 8 that increasing the F2C layered structure, by adding the so-called HCF layer, decreases the Service Disruption Probability.

To assure the validity of the simulation results presented above, we evaluate the disruption probability of an F2C-2 architecture by gradually increasing the amount of LF nodes available. Based on the results shown in Fig. 9 we confirm that the mobility of Fog nodes can decrease

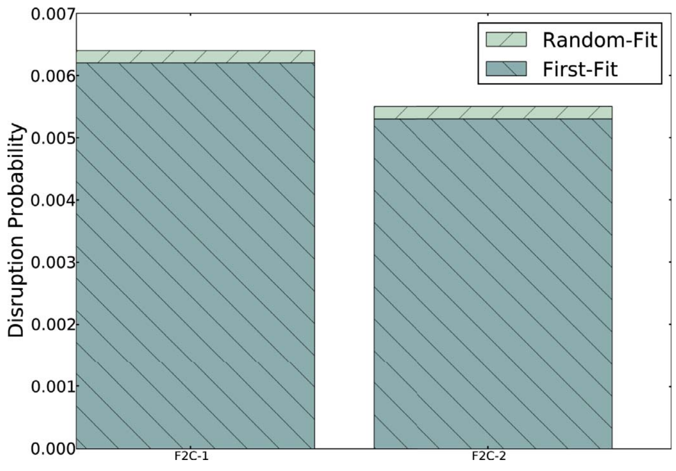

Fig. 8. Disruption probability.

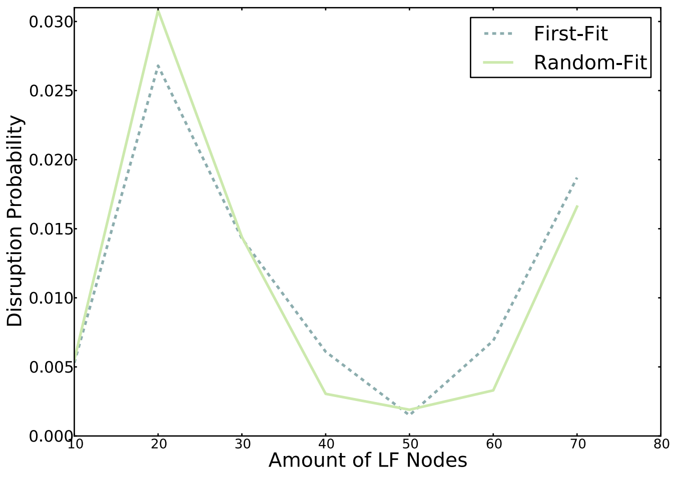

Fig. 9. Disruption probability of an F2C-2 architecture vs amount of LF-nodes.

the performance of F2C architectures and reduce the resilience of an F2C-2 architecture regarding mobility. This is, more LF nodes available can (typically the case in IoT scenarios) improve the Service Response Time, but can be counterproductive regarding service resilience. Notice that the Disruption Probability for both First-Fit and Random-Fit varies when the number of LF nodes is increased. Moreover, the Disruption Probability does not mandatory increase with a larger number of LF nodes. This is because the more the LF nodes, the more the chances to avoid Service Disruption (of a service running on the LF layer) since more LF nodes become available. However, it can be observed that there is a trend related to an increase in the Disruption Probability due to an increase in the population of LF nodes. The lower and upper limits of this trend do not fluctuate significantly as the number of LF nodes increases.

Based on the simulations presented in this section, the following lessons may be learnt.

- The use of both LF and HCF nodes decreases the Service Response Time as well as the Network Bandwidth Occupancy and Power Consumption.
- The use of both LF and HCF nodes increases the probability of service to be disrupted. Nevertheless, the lower and upper limits of the Service Disruption Probability do not fluctuate significantly as the number of fog nodes increases.
- The use of a first-fit heuristic for service allocation, which gives preference to Fog allocation, is preferable in F2C scenarios.
- A layered F2C architecture is suitable for performance metrics, such as Service Response Time, Network Bandwidth Occupancy and Power Consumption. Nevertheless, it is not suitable for reducing the Service Disruption Probability.

6. Conclusion

In this paper, we have evaluated the performance of Fog-to-Cloud (F2C) systems to show the benefits of a management architecture putting together the different capacities brought by both devices at the edge and at the Cloud. To that end, the paper focuses on the DSE problem, standing for the strategy to be defined for job allocation on such a mobile, dynamic and heterogeneous scenario. We end up proposing two basic resource allocation strategies, First-Fit and Random-Fit (not yet considering potential business models and sharing policies to come), utilized for validation purposes.

The evaluation procedure consists in testing the performance of four key metrics, Service Response Time, Power Consumption, Network Bandwidth Occupancy and Service Disruption Probability on three different architectural scenarios, Cloud, F2C-1 and F2C-2, all incremental in the set of layers (i.e., resources) to be considered. By means of simulation results, we showed that Cloud and Fog computing systems can be used in a highly complementary fashion to increase the overall performance. Indeed, building a hierarchical and layered stack of resources enables the parallel execution of services either in the cloud, fog or both fog and cloud at the same time.

As a future line of work, we plan to extend the presented evaluation model regarding F2C by considering how key performance metrics may impact on the potential business models.

Acknowledgments

This work was supported by the Spanish Ministry of Economy and Competitiveness and by the European Regional Development Fund, under contracts TEC2015-66220-R (MINECO/FEDER) and TEC2015-71932-REDT, and by the H2020 EU mF2C project reference 730929.

References

[1] D. C. Plummer, et al., Top 10 strategic predictions for 2017 and beyond: surviving the storm winds of digital disruption, gartner inc, http://www.gartner.com/binaries/content/assets/events/keywords/cio/ciode5/top_strategic_predictions_fo_315910.pdf, October 2016.

[2] L. Heilig, S. Voss, A scientometric analysis of cloud computing literature, Cloud Comput., IEEE Trans. 2 (3) (2014) 266–278, http://dx.doi.org/10.1109/TCC.2014.2321168.

[3] F. Bonomi, R. Milito, J. Zhu, S. Addepalli, Fog Computing and Its Role in the Internet of Things, Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing, MCC '12, ACM, New York, NY, USA, 2012, pp. 13–16, http://dx.doi.org/10.1145/2342509.2342513.

[4] M. Chiang, T. Zhang, Fog and IoT: an overview of research opportunities, IEEE Internet Things J. PP (99) (2016) 1–1. doi:10.1109/JIOT.2016.2584538.

[5] https://www.openfogconsortium.org/ [Online; accessed June-2016], 2017.

[6] X. Masip-Bruin, E. Marin-Tordera, G. Tashakor, A. Jukan, G.J. Ren, Foggy clouds and cloudy fogs: a real need for coordinated management of fog-to-cloud computing systems, IEEE Wireless Commun. 23 (5) (2016) 120–128, http://dx.doi.org/10.1109/MWC.2016.7721750.

[7] R. Deng, R. Lu, C. Lai, T. Luan, Towards power consumption delay tradeoff by workload allocation in cloud fog computing, Communications (ICC), 2015 IEEE International Conference on, (2015), pp. 3909–3914, http://dx.doi.org/10.1109/ICC.2015.7248934.

[8] R. Deng, R. Lu, C. Lai, T.H. Luan, H. Liang, Optimal workload allocation in fog-cloud computing towards balanced delay and power consumption, IEEE Internet Things J. PP (99) (2016) 1–1. doi:10.1109/JIOT.2016.2565516.

[9] P. Simoens, L.V. Herzeele, F. Vandeputte, L. Vermoesen, Challenges for orchestration and instance selection of composite services in distributed edge clouds, 2015 IFIP/IEEE International Symposium on Integrated Network Management (IM), (2015), pp. 1196–1201.

[10] M. Aazam, E.-N. Huh, Dynamic resource provisioning through Fog micro datacenter, Pervasive Computing and Communication Workshops (PerCom Workshops), 2015 IEEE International Conference on, (2015), pp. 105–110, http://dx.doi.org/10.1109/PERCOMW.2015.7134002.

[11] B. Tang, Z. Chen, G. Hefferman, T. Wei, H. He, Q. Yang, A hierarchical distributed fog computing architecture for big data analysis in smart cities, Proceedings of the ASE BigData & SocialInformatics 2015, ASE BD&SI '15, ACM, New York, NY, USA, 2015, pp. 28:1–28:6, http://dx.doi.org/10.1145/2818869.2818898.

[12] IBM Bluemix at http://www.ibm.com/cloud-computing/bluemix/.

[13] https://aws.amazon.com/greengrass.

[14] R.M. Arasanal, D.U. Rumani, Distributed computing and internet technology: 9th international conference, ICDCIT 2013, Bhubaneswar, India, February 5, 8, 2013. proceedings, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 115, 125. doi:10.1007/978-3-642-36071-8_8.

[15] V. Valancius, N. Laoutaris, L. Massoulié, C. Diot, P. Rodriguez, Greening the Internet with nano data centers, Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies, CoNEXT '09, ACM, New York, NY, USA, 2009, pp. 37–48, http://dx.doi.org/10.1145/1658939.1658944.

[16] http://www.craax.upc.edu/, [Online; accessed September 2017], 2017.

[17] https://omnetpp.org/, [Online; accessed September 2017], 2017.

[18] A. Curtis, W. Kim, P. Yalagandula, Mahout: low overhead datacenter traffic management using end host based elephant detection, INFOCOM, 2011 Proceedings IEEE, (2011), pp. 1629–1637, http://dx.doi.org/10.1109/INFCOM.2011.5934956.

[19] M.A. Hassan, M. Xiao, Q. Wei, S. Chen, Help your mobile applications with fog computing, 2015 12th Annual IEEE International Conference on Sensing, Communication, and Networking - Workshops (SECON Workshops), (2015), pp. 1–6, http://dx.doi.org/10.1109/SECONW.2015.7328146.

[20] D. Huang, T. Xing, H. Wu, Mobile cloud computing service models: a user-centric approach, IEEE Netw. 27 (5) (2013) 6–11, http://dx.doi.org/10.1109/MNET.2013.6616109.

[21] F. Bai, N. Sadagopan, A. Helmy, IMPORTANT: a framework to systematically analyze the impact of mobility on performance of routing protocols for adhoc networks, INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies, 2 (2003), pp. 825–835 vol.2, http://dx.doi.org/10.1109/INFCOM.2003.1208920.

[22] M. Caliskan, A. Barthels, B. Scheuermann, M. Mauve, Predicting parking lot occupancy in vehicular ad hoc networks, 2007 IEEE 65th Vehicular Technology Conference - VTC2007-Spring, (2007), pp. 277–281.