Computer Networks 93 (2015) 435–447

Contents lists available at ScienceDirect

Computer Networks

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

Elastic virtual machine placement in cloud computing network environments☆

Eleni Kavvadia[a], Spyros Sagiadinos[a], Konstantinos Oikonomou[a,*], Giorgos Tsioutsiouliklis[a], Sonia Aïssa[b]

[a] Ionian University, Department of Informatics, Tsirigoti Square 7, 49100 Corfu, Greece
[b] INRS, University of Quebec, Montreal, QC, Canada

ARTICLE INFO

Article history:
Received 15 February 2015
Revised 7 August 2015
Accepted 19 September 2015
Available online 19 October 2015

Keywords:
Virtual machine
Elastic placement
Cloud computing
Network architecture
Facility location

ABSTRACT

The growth of cloud computing and the need to support the ever increasing number of applications introduces new challenges and gives rise to various optimization problems, such as calculating the number and location of virtual machines instantiating cloud services to minimize a well-defined cost function. This paper introduces a novel cloud computing network architecture that allows for the formulation of the optimization as an Uncapacitated Facility Location (UFL) problem, where a facility corresponds to an instantiation of a particular service (e.g. a virtual machine). Since UFL is not only difficult (NP-hard and requires global information), but also its centralized solution is non-scalable, the approach followed here is distributed and elastic, and relays local information to improve scalability. In particular, virtual machine replication and merging are proposed and analyzed ensuring overall cost reduction. In addition, a policy that employs virtual machine replication and merging along with migration is proposed to reduce the overall cost for using a service. The efficiency of this policy and its limitations are analyzed and discussed, with simulation results supporting the analytical findings and demonstrating a significant overall cost reduction when the proposed policy is implemented.

1. Introduction

Cloud computing has seen significant growth over the last decade allowing for proliferation of numerous applications and gradually changing the distributed networking paradigm to being more centralized, its core being the data center of the cloud service providers [1,2]. As the number of users and cloud services increase, various scalability problems have arisen within data centers, e.g. performance, power efficiency [3], or the network architecture, e.g. bandwidth, time delays [4]. For example, one of the key optimization problems in this area is to find the location of a particular virtual machine that instantiates a cloud service, incorporating the communication costs over the network links, user traffic demands, and maintenance costs for supporting this service, while minimizing the overall cost.

The focus here is this optimization problem from a network perspective. A cloud computing network architecture is introduced that integrates recent trends, e.g. cloud, fog, and access networks [1,4,5]. Virtual machines (or facilities) that instantiate the particular cloud service offered by the corresponding cloud service provider, can be hosted by data centers within the cloud network or by fog devices

☆ Part of this work was originally published in the proceedings of the IEEE International Conference on Communications (ICC), Ottawa, Canada, 10–15 June, 2012.

* Corresponding author. Tel: +30 26610 87708; fax: +30 26610 87766.
E-mail addresses: ekavvadia@ionio.gr (E. Kavvadia), p10sagi@ionio.gr (S. Sagiadinos), okon@ionio.gr (K. Oikonomou), c10tsio@ionio.gr (G. Tsioutsiouliklis), sonia.aissa@ieee.org (S. Aïssa).

http://dx.doi.org/10.1016/j.comnet.2015.09.038
1389-1286/

within the fog network [6,7]. Further details are presented in Section 2.

Cloud computing network architecture allows for formulation of an Uncapacitated Facility Location (UFL) problem [8] to determine the optimal number of virtual machines offering a given service and their location in the network. However, UFL is a large optimization problem, NP-hard and requires global knowledge [8], and for inherently dynamic environments, such as cloud computing networks, existing centralized approaches that solve UFL do not scale. Even approximate approaches require global information, which is prohibitive in this environment. The requirement for a scalable approach is addressed here by exploiting local information in an elastic hill climbing manner that distributedly moves, replicates, and merges service facilities.

Virtual machines at the "Internet as a Service" layer [1] are assumed to instantiate a given service, as further explained in Section 2. Depending on the implementation, a service may consist of multiple components implemented as individual virtual machines. The proposed approach considers all virtual machines that instantiate a service as a single facility, and throughout this paper, service facility (or simply facility) refers to the instantiation of a given service, corresponding to one or more virtual machines. Such a facility is capable of migrating from one location to another, replicate itself and merge with other facilities that instantiate the same service. These operations are allowed only if conditions are satisfied that allow for overall cost reduction.

Specifically, this paper elaborates on facility replication when conditions of overall cost reduction are satisfied, and a facility merging approach is introduced that allows facilities to merge under conditions that ensure overall cost reduction. The previous conditions and the properties of replication and merging are also investigated. A scalable UFL (s-UFL) policy is proposed that integrates facility replication and merging along with facility migration [9] for moving service facilities within the cloud computing network. The efficiency of the proposed policy is studied, showing that cost reduction in cloud computing networks is possible under certain conditions also derived and investigated here. The ease of implementation in the network nodes (i.e., data centers and fog devices) is another advantage of the proposed policy. The main requirement for implementation is a monitoring mechanism to estimate the aggregate incoming/outgoing traffic load of the data center or fog device that hosts a service facility. It should be noted that the problem is formulated here as uncapacitated, even though hardware is never of unlimited capacity. However, the results derived here for the uncapacitated case can easily be extended to apply in capacitated scenarios.

Analytical results are supported by simulation. In particular, the proposed s-UFL policy achieves efficiency by the exploitation of local information resulting in significant cost reduction for supporting cloud services. Various simulation scenarios of more or less than the optimal number of facilities randomly scattered within the cloud computing network, used as the initial setup, reveal the proposed policys behavior.

Cloud computing network architecture is introduced in Section 2 and the corresponding formulation of the UFL problem is presented in Section 3. The analysis of facility replication and merging is presented in Section 4. The proposed s-UFL policy is presented and analyzed in Section 5, and simulation results in Section 6. Previous related work is discussed in Section 7 and conclusions are summarized in Section 8. For ease of readability, the various proofs are included in a separate technical report [10] and the list of important symbols in the Appendix.

2. Cloud computing network architecture

The hardware required to support cloud services is implemented in data centers [11], i.e., plants typically containing thousands of servers in appropriate conditions, e.g. air-conditioned, and suitably interconnected [12]. This hardware is the basis for supporting the wide range of cloud computing services. A typical layered cloud computing architecture [1] that allows for the categorization of cloud services consists of four layers. At the lowest level, the hardware (implemented in data centers) is accessed as a service from the next, infrastructure as a service (IaaS), layer. For example, virtual machines consisting of various levels of computational power, memory, and storage are a service offered at the IaaS layer. The platform as a service (PaaS) layer then abstracts the necessity to deploy applications on virtual machines. Finally, the service as a platform (PaaS) layer offers web based services, multimedia etc., to the end user. More information about the particular layers and their properties can be found in various survey papers, e.g. [1].

Cloud service providers often utilize more than one data center, depending on a variety of criteria, such as location, proximity to users, and energy consumption [13]. A cloud network is formed from a number of data centers as shown in Fig. 1. In this network, the cloud service provider assigns hardware allocated to a particular service. Depending on service type, user demands, topology, etc., the provider may decide to change the service location (i.e., the particular data center hosting it) and/or create more instances of the service and host them accordingly. As stated above, these instances of a service will be referred to as facilities of the given service. For terminology convenience, a facility corresponds to one (or more) virtual machine at the IaaS layer [1].

In some cases, e.g. when there are strict time constraints, a service facility may have to be located close to the end user to be sufficiently supported. Since data centers are expensive and generally cannot be close to the end users, fog computing [4] has been proposed to provide support services close to the end user. Fog computing infrastructure is based on hardware similar to data center infrastructure, but not at the same scale, and so the corresponding hardware is referred to here as a fog device, e.g. [6,7]. A fog device is considered to be capable of hosting a service facility similarly to a data center under the assumption that the hardware requirements will be limited compared to those for a data center. Thus, a fog device follows the layered cloud computing architecture [1]. Depending on the case, it may be a common device, such as a small server, or gateway, etc. [14].

A fog device, along with other such devices, is part of a fog network, as shown in Fig. 1. A fog network falls between the cloud network and the user access network, allowing the user equipment to use the services offered by the cloud service provider, through the network interfaces. Note

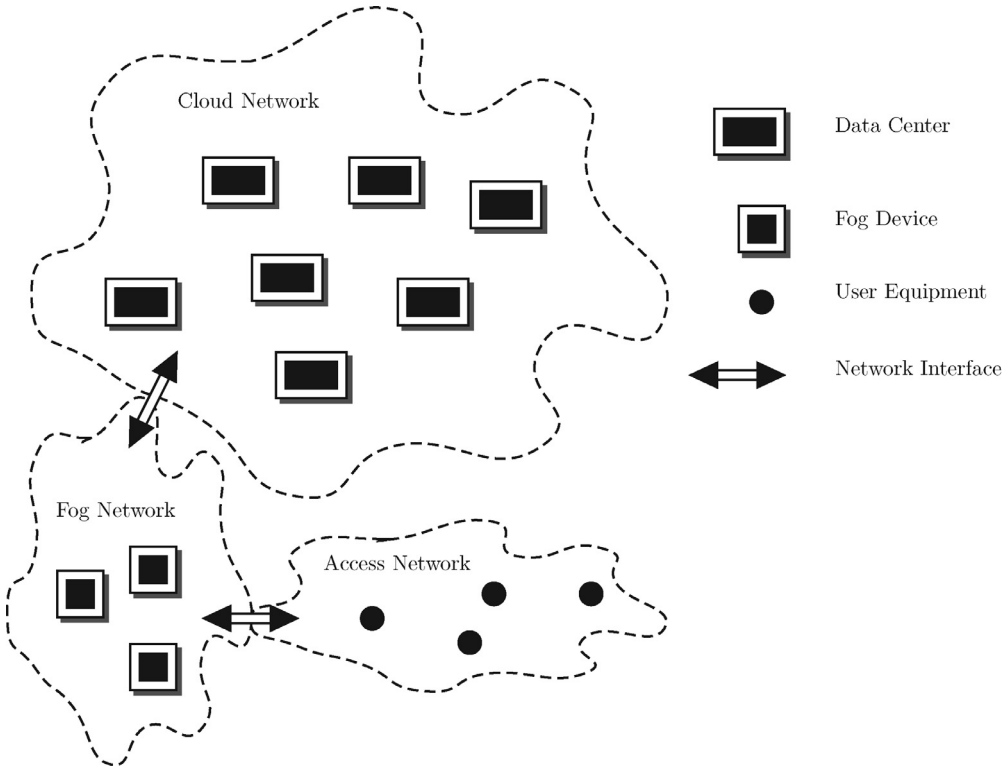


Fig. 1. An example cloud computing network architecture consisting of three subnetworks: (i) cloud network consisting of data centers; (ii) fog network consisting of fog devices; and (iii) access network consisting of user equipment. The subnetworks are connected using specific network interfaces.

that the access network consists only of user equipment, any other sophisticated devices (e.g. network operator's equipment) are assumed to be part of the fog network or the interface connecting them. These network types (cloud, fog, and access) are treated here as subnetworks of an overall network, referred to as the cloud computing network, with the corresponding architecture as shown in Fig. 1.

The cloud computing network architecture is capable of capturing various network configurations. For example, in those cases where a fog network intermediate is not required, users will be connected directly to the cloud network in the sense that a service facility would be hosted by a data center rather than a fog device. In other cases, the functionality of the fog network may be part of the access network, etc. For this paper, we consider the cloud computing network architecture to be as previously described. To simplify the terminology, network will refer to both cloud and fog networks, unless otherwise stated.

A given service may be instantiated by one or more facilities within the network. Clearly, there is a question about how many facilities should exist at any time and where (which data center and/or fog device) they should be located. There is a cost for supporting a facility that depends on the maintenance process, and a cost for data packets to travel in the network that depends on the network topology and user traffic demands.

For traffic demands, common practice is to assume that data are exchanged between user equipment and service facilities. Cloud computing alters this traditional paradigm, since data specific for a service facility and a given user may be exchanged between the facility that implements the service and some other facility that instantiates another service for the user. For example, suppose a user executes a remote desktop application on his equipment (e.g. a tablet) that controls a virtual machine (e.g. a Linux instance) located somewhere in the network. The user downloads a large file on this Linux instance, after using a cloud service located elsewhere in the network. A large data transfer occurs between the two facilities within the network for this user, and a smaller data transfer between the user equipment and the facilities (mostly control information for handling the remote desktop application).

This modification of the traditional paradigm of user data flows needs to be included when considering the problem of how many facilities should instantiate a service, and where they should be placed, which can be addressed as a facility location as shown below.

3. Facility location problem formulation

To determine the number and the location of service facilities, a facility location problem is formulated, starting from the network topology.

3.1. Topology model

Let any data center and fog device that can host a facility that corresponds to a given service, be a *node* of a graph $\mathbb{G}$, where the set of nodes of this graph are denoted as $V(\mathbb{G})$. Any

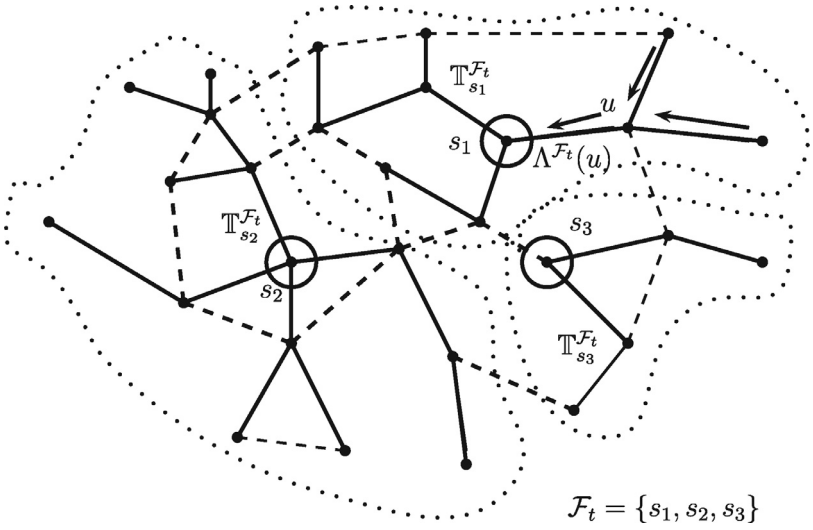


Fig. 2. An example network of three facility nodes (circles) at time t, $\mathcal{F}_t = \{s_1, s_2, s_3\}$. Dense lines correspond to the shortest path tree links. The area included in the dotted shape relates to shortest path trees $\mathbb{T}^{\mathcal{F}_t}_{s_1}$, $\mathbb{T}^{\mathcal{F}_t}_{s_2}$, $\mathbb{T}^{\mathcal{F}_t}_{s_3}$. Dashed lines correspond to the rest of the network links (i.e., $\in E(\mathbb{G}) \setminus E(\mathbb{T}^{\mathcal{F}_t}_{s_1}) \cup E(\mathbb{T}^{\mathcal{F}_t}_{s_2}) \cup E(\mathbb{T}^{\mathcal{F}_t}_{s_3})$).

node $u \in V(\mathbb{G})$ is connected to other nodes $v \in V(\mathbb{G})$, and a link (u, v) is created. Each link is assigned a (positive) link weight, denoted as $w(u, v)$. Let $\mathcal{S}(u)$ be the set of nodes that are neighbors (or adjacent) to node u. The graph, $\mathbb{G}$, will also be referred to as the network topology or simply the network.

When data are exchanged between node u and v, then the corresponding cost (per data packet) for this communication is $w(u, v)$. For simplicity, it is assumed that data are transmitted in the network in the form of equal sized data packets. The communication cost corresponds to the investment required to support a given data flow in the network (e.g. hardware) according to certain constraints (e.g. time delays, packet errors) and the corresponding maintenance expenses. For example, a data packet traveling within a cloud network from one data center (node u) to another (node v), and passing through various network devices, eventually arrives at its destination having imposed a total cost of $w(u, v)$.

From these definitions of nodes and links, graph $\mathbb{G}$ represents an overlay abstraction of the actual network topology in the sense that even though nodes represent existing elements (i.e., data centers and fog devices), links represent possible data transfer paths (introducing a cost as given by the link weight). For example, in the upper right part of the network of Fig. 2, two nodes send data packets (depicted as arrows) towards node s_1 that hosts the facility (depicted within a small circle) through node u. This representation does not imply that node u (either a data center or a fog device) necessarily plays a relay role. Rather, it shows that the cost for these packets to arrive at node s_1 corresponds to the summation of the cost to arrive at node u plus the cost $w(u, s_1)$ of link (u, s_1).

Assuming a shortest path among any two nodes u and v, the summation of the link weights along this path is referred to as the path's distance and it is denoted by $d(u, v)$.

3.2. Facility placement model

It is common to assume that user equipment are not able to host service facilities due to hardware limitations, and, hence, the topology model excludes them. However, the data flow corresponding to users for a given service is incorporated by suitably defining the traffic load, $\lambda(u)$, for any node $u \in V(\mathbb{G})$. $\lambda(u)$ corresponds to the (average) number of data packets created per time unit for the given service. Regarding the traffic load of user equipment, note that any user equipment, along with its closest data center or fog device, are at the edge of the network (see Fig. 1). Since user equipment are not represented in $\mathbb{G}$, it is realistic to map the corresponding traffic load onto the nearest (in terms of distance) data center or fog device.

The facility problem formulated below deals with only one service type. The case of multiple service types (that may simultaneously exist in the network) is supported by applying the same formulation for each different service type.

Let $\mathcal{F}_t$ denote the set of facility nodes (i.e., nodes that correspond to data centers and fog devices hosting the service facilities) at time t. The traffic requirements of each node, u, are served at time t by the closest (in terms of distance) facility node, given by the set of facility nodes $\varphi^{\mathcal{F}_t}(u)$, where $\varphi^{\mathcal{F}_t}(u) = \{s : d(u, s) \leq d(u, s'), \forall s' \in \mathcal{F}_t\}$. Thus, each facility node, s, serves all nodes, u, for which $\varphi^{\mathcal{F}_t}(u) = \{s\}$. Actually, all these nodes belong to a shortest path tree of root node s, denoted by $\mathbb{T}^{\mathcal{F}_t}_s$. Let $\mathbb{T}^{\mathcal{F}_t}_s(u)$ denote the subtree of $\mathbb{T}^{\mathcal{F}_t}_s$ pertaining to root node u, where $u \in \mathbb{T}^{\mathcal{F}_t}_s$. Fig. 2 shows an example network and graphically illustrates the shortest path trees for a set of facility nodes $\mathcal{F}_t$ at time t.

Given that each node accumulates data packets corresponding to all nodes $v \in \mathbb{T}^{\mathcal{F}_t}_s(u)$, $s = \varphi^{\mathcal{F}_t}(u)$, the aggregate traffic load, $\Lambda^{\mathcal{F}_t}(u)$, of node u is

$$\Lambda^{\mathcal{F}_t}(u) = \sum_{\forall v \in \mathbb{T}^{\mathcal{F}_t}_s(u)} \lambda(v). \tag{1}$$

Fig. 2 shows $\Lambda^{\mathcal{F}_t}(u)$ for a node $u \in \mathcal{S}(s_1)$. The arrows correspond to the paths over which data packets are forwarded through the network links towards the corresponding facility node.

Any data packet created at any node u at time t is forwarded to its corresponding facility node, $s = \varphi^{\mathcal{F}_t}(u)$, over

a shortest path, $\{u, \ldots, s\}$, contributing $d(u, s)$ units to the communication cost. Since $\lambda(u)$ packets per time unit are created (on average) at node u, they contribute (on average) $\lambda(u)d(u, s)$ units to the overall cost. Therefore, the (average) cost, $\mathcal{C}^{\mathcal{F}_t}(s)$, contributed by all nodes served by facility s, is

$$\mathcal{C}^{\mathcal{F}_t}(s) = \sum_{\forall u \in \mathbb{T}_s^{\mathcal{F}_t}} \lambda(u) d(u, s). \quad (2)$$

There is a close relation between maintenance costs and link weights from a provider's point of view. In particular, for a service to be offered to the end user, the provider has, among other duties, to ensure (i) that this service is up and running; and (ii) that there is enough bandwidth for users to access it. The latter cost, i.e., bandwidth, link capacity, etc., is captured here by the link weight definition. The former cost for ensuring that the service is running, e.g. configuration, administration, hardware, etc., is the maintenance cost, $\mathfrak{c}$. Both costs correspond to real investments and they vary from provider to provider depending on the services offered, the number of supported subscribers, user traffic profiles, etc. As shown below, the trade-off regarding investing in bandwidth or maintaining more service facilities, is successfully captured and it is further investigated in Section 6.

Given the maintenance cost, $\mathfrak{c}$, for hosting each facility, each facility, s, contributes a total cost of $\mathcal{C}^{\mathcal{F}_t}(s) + \mathfrak{c}$. Let $\mathcal{C}^{\mathcal{F}_t}$ be the overall cost in the network at time t, then

$$\mathcal{C}^{\mathcal{F}_t} = \sum_{\forall s \in \mathcal{F}_t} \mathcal{C}^{\mathcal{F}_t}(s) + |\mathcal{F}_t|\mathfrak{c}, \quad (3)$$

where $|\mathcal{F}_t|$ is the number of facilities in the network at time t.

Let $\mathcal{O}$ be the set of facility nodes, such that the cost given by Eq. (3) is minimized, then $\mathcal{O}$ corresponds to the solution of the UFL optimization problem [8].

The simplest form of a location problem is to derive the location for placing a facility that represents a particular service. Assuming that any location has the capacity to host the facility, traffic demands and link weights are considered to minimize the overall cost. When the number of facilities is fixed, this is known as the median problem, when the number of facilities also needs to be optimized, in addition to location, this is known as the UFL problem. Given that the proposed network architecture, apart from the powerful data center, includes less powerful devices, such as fog devices, capacity constraints could also be introduced in the problem formulation (e.g. limits on the number of virtual machines that can be hosted by a fog device). The proposed approach can be easily extended for the case of limited capacity (e.g. by eliminating nodes that can no longer host facilities).

3.3. Facility migration

UFL is a large optimization problem (NP-hard) requiring significant computational resources [8]. Cloud environments are typically dynamic and multiple changes (e.g. the traffic load corresponding to users) are expected to frequently occur. Consequently, traditional centralized approaches for solving UFL are not suitable, and distributed approaches should be used, based on local information. Moving facilities to more cost efficient locations based on information only available at the local facility nodes is facility migration [9].

Let $\mathbb{S}^{\mathcal{F}_t}(s) = \{u : \mathcal{S}(s) \cap \mathbb{T}_s^{\mathcal{F}_t} \wedge s \in \mathcal{F}_t\}$ be the set of neighbor nodes of a facility node, s, served by (the same) node s. When the facility is at node s, let $s \xrightarrow{t} s'$ be facility migration to some node $s' \in \mathbb{S}^{\mathcal{F}_t}(s)$ at time t, according to the facility migration rule:

Facility migration $s \xrightarrow{t} s'$ occurs iff $\Lambda^{\mathcal{F}_t}(s') > \frac{\Lambda^{\mathcal{F}_t}(s)}{2}$.

According to [9], if facility migration $s \xrightarrow{t} s'$ occurs, then $\mathcal{C}^{\mathcal{F}_{t+1}} < \mathcal{C}^{\mathcal{F}_t}$ holds, where $\mathcal{F}_{t+1} = \mathcal{F}_t \cup \{s'\} \setminus \{s\}$. Note that migration is a conservative policy and there might be cases that $\mathcal{C}^{\mathcal{F}_{t+1}} < \mathcal{C}^{\mathcal{F}_t}$ holds but the facility movement may not occur, depending on the topology [9].

Assuming that the facility node is capable of monitoring the aggregate traffic load over its own links, then, based on their relative values, a facility movement to a neighbor node is decided (or not) and the overall cost is reduced (or remains the same). Migration is a simple, low cost approach that follows a hill climbing philosophy to move facilities. Facilities are moved from a neighbor node depending on the criteria. When such a movement occurs, the end users should be aware of this change. This requirement is trivial, since the node previously hosting the facility can provide redirection information.

Facility migration, however, introduces a migration cost of supporting virtual machine movements among data centers and fog devices. Even for cases of replication and merging proposed below, some sort of service migration also occurs (e.g. movement of a virtual machine from one data center to another). This migration cost depends on several factors, such as the amount of data to be copied [15], and can be significantly increased in terms of time delay [16]. In addition, the number of users plays an important role as well as the migration distance [17]. Therefore, the gains from migration must more than compensate for the losses.

Consider the case when traffic fluctuates rapidly such that there is insufficient time to compensate for the cost of moving (and replicating/merging) with the achieved cost reduction. The key here is the estimation of aggregate traffic loads. These estimations are crucial and specific for each different service provider. One benefit of the proposed approach is that aggregate traffic loads are easier and faster to estimate than individual loads. Modeling and analyzing these particular factors is beyond this work. However, the analysis and subsequent simulations, indicate whether a facility migration, replication, or merge should occur when migration costs are also considered.

Facility migration does not change the number of facilities, only their positions. When their number should be also optimized, then replication or merging of existing facilities occurs. Inspired by the local information based scalable behavior of migration, conditions with respect to both replication and merging of a service facility are derived in Section 4. The main goal is to exploit local information to reduce overall cost in the network.

4. Facility replication and merging

As discussed above, the dynamic network characteristics of cloud environments (e.g. topology and user traffic demands) do not allow employment of centralized solutions for the UFL problem. Even approximate approaches require

global information, thus being non-scalable. The proposed approach is to change the number of facilities by a replication/merging policy based on local information and respond to the dynamic changes of the cloud network.

4.1. Facility replication

A facility replication corresponds to creating another facility in a neighbor node. From an implementation point of view, this is similar to creating a new virtual machine in a neighboring data center or fog device. The new facility is identical to the original in the sense that it instantiates a specific service. What is clearly different is the facilities' location. The new facility is now closer to some nodes than the original, and these nodes will prefer being served by the new facility. However, these nodes may not be aware of the existence of the new facility. This dissemination can be undertaken by the original facility, forwarding the user demands to the new facility. In this paper, the focus is on the conditions for facility replication to ensure overall cost reduction.

Let a facility replication event at time t be $s \overset{t}{\mapsto} s'$ for a facility node $s \in \mathcal{F}_t$ and neighboring node $s' \in \mathbb{S}^{\mathcal{F}_t}(s)$, then $\mathcal{F}_{t+1} = \mathcal{F}_t \cup \{s'\}$. The aim is to reduce the overall cost, or $\mathcal{C}^{\mathcal{F}_{t+1}} < \mathcal{C}^{\mathcal{F}_t}$. Let $\Phi_{s \overset{t}{\mapsto} s'}$ be the set of nodes whose distance to node s' is smaller than the distance to their (former) facility node at time t, then $\Phi_{s \overset{t}{\mapsto} s'} = \{u : d(u,s') < d(u, \varphi^{\mathcal{F}_t}(u))\}$. Also, let $\Phi_{s \overset{t}{\mapsto} s'}(z) = \Phi_{s \overset{t}{\mapsto} s'} \cap \mathbb{T}_z^{\mathcal{F}_t}$ for any facility node $z \in \mathcal{F}_t$. Fig. 3 shows an example facility replication. A number of lemmas and theorems follow that allow proof of Proposition 1 that is the basis of the proposed replication policy.

Lemma 1. *For a facility replication, $s \overset{t}{\mapsto} s'$, $\Phi_{s \overset{t}{\mapsto} s'} = \mathbb{T}_{s'}^{\mathcal{F}_{t+1}}$ and $\Phi_{s \overset{t}{\mapsto} s'} = \cup_{\forall z \in \mathcal{F}_t} \Phi_{s \overset{t}{\mapsto} s'}(z)$.*

Proof. The proof is included in [10]. □

Lemma 2. *For a facility replication $s \overset{t}{\mapsto} s'$ and any facility node $z \in \mathcal{F}_t$, then $\mathbb{T}_z^{\mathcal{F}_t} = \mathbb{T}_z^{\mathcal{F}_{t+1}} \cup \Phi_{s \overset{t}{\mapsto} s'}(z)$ and $\mathbb{T}_z^{\mathcal{F}_{t+1}} \cap \Phi_{s \overset{t}{\mapsto} s'}(z) = \emptyset$.*

Proof. The proof is included in [10]. □

Theorem 1. *For a facility replication, $s \overset{t}{\mapsto} s'$, $\mathcal{C}^{\mathcal{F}_{t+1}} < \mathcal{C}^{\mathcal{F}_t}$ is satisfied iff $\mathfrak{c} < \sum_{\forall z \in \mathcal{F}_t} \sum_{\forall u \in \Phi_{s \overset{t}{\mapsto} s'}(z)} \lambda(u)(d(u,z) - d(u,s'))$.*

Proof. The proof is included in [10]. □

Lemma 3. *For a facility replication $s \overset{t}{\mapsto} s'$, $\Phi_{s \overset{t}{\mapsto} s'}(s) = \mathbb{T}_s^{\mathcal{F}_t}(s')$.*

Proof. The proof is included in [10]. □

The right term of the condition in Theorem 1 is based on non-local information, since it requires knowledge of sets $\mathcal{F}_t$, $\Phi_{s \overset{t}{\mapsto} s'}(z)$, for all $z \in \mathcal{F}_t$, and knowledge of product $\lambda(u)(d(u,s') - d(u,z))$ for all $u \in \Phi_{s \overset{t}{\mapsto} s'}(z)$. The following proposition provides for a local information based condition that serves as the basis for facility replication as detailed below.

Proposition 1. *For a facility replication $s \overset{t}{\mapsto} s'$, if $\Lambda^{\mathcal{F}_t}(s') w(s,s') > \mathfrak{c}$, then $\mathcal{C}^{\mathcal{F}_{t+1}} < \mathcal{C}^{\mathcal{F}_t}$ is satisfied.*

Proof. The proof is included in [10]. □

Based on Proposition 1, the following rule for facility replication is proposed.

Facility replication: facility replication $s \overset{t}{\mapsto} s'$ occurs iff $\Lambda^{\mathcal{F}_t}(s') w(s,s') > \mathfrak{c}$.

Any data center is expected to be aware of the data traffic coming and going through certain links. The same applies for a fog device. Consequently, any facility node is considered to be capable of monitoring the aggregate traffic load over

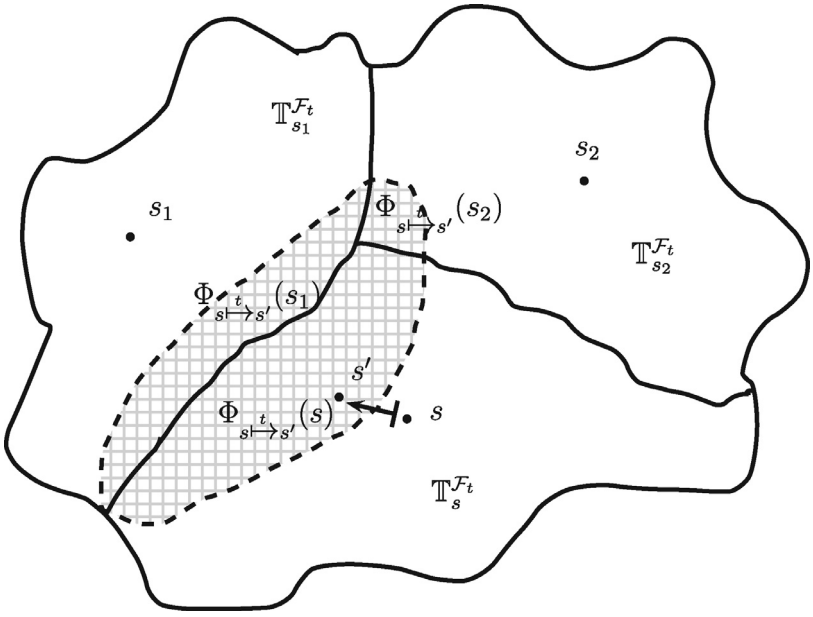


Fig. 3. Example network areas of three facility nodes at time t, $\mathcal{F}_t = \{s, s_1, s_2\}$, and facility replication $s \overset{t}{\mapsto} s'$. The areas within the shapes of dense lines correspond to sets $\mathbb{T}_s^{\mathcal{F}_t}$, $\mathbb{T}_{s_1}^{\mathcal{F}_t}$ and $\mathbb{T}_{s_2}^{\mathcal{F}_t}$. The squared area corresponds to $\Phi_{s \overset{t}{\mapsto} s'}$. Sets $\Phi_{s \overset{t}{\mapsto} s'}(s)$, $\Phi_{s \overset{t}{\mapsto} s'}(s_1)$ and $\Phi_{s \overset{t}{\mapsto} s'}(s_2)$ are also depicted.

its own links, $\Lambda^{\mathcal{F}_t}(s')$, and it has knowledge of its own link weights, $w(s,s')$, and the cost for hosting a facility, $\mathfrak{c}$. Thus, condition $\Lambda^{\mathcal{F}_t}(s')w(s,s') > \mathfrak{c}$ is based on information that is locally available at the facility node. Note that this replication mechanism is a conservative one in the sense that there might be cases where $\mathcal{C}^{\mathcal{F}_{t+1}} < \mathcal{C}^{\mathcal{F}_t}$ is satisfied but a new facility may not be created (e.g. Proposition 1 may not be satisfied). This is not the case for only one facility in the network, as shown Corollary 1.

Corollary 1. *For a facility replication $s \overset{t}{\mapsto} s'$ and $|\mathcal{F}_t| = 1$, $\mathcal{C}^{\mathcal{F}_{t+1}} < \mathcal{C}^{\mathcal{F}_t}$ iff $\Lambda^{\mathcal{F}_t}(s')w(s,s') > \mathfrak{c}$.*

Proof. The proof is included in [10]. □

4.2. Facility merging

Facility replication has been proposed to increase the number of facilities when required by the network. However, due to changes in the cloud computing network environment, there might be cases where the number of facilities needs to be reduced. Any reduction of the facilities should also be decided based on local information as for facility replication.

The number of facilities in the network is reduced by merging two facilities. The core concept is to check all facility pairs if merging them would decrease the overall cost. When two facilities merge, there is a maintenance cost reduction, but there is consequential cost increment due to longer distances between the merged facility and end nodes, and the increment should be less than the cost reduction. After merging, only one of the facilities is left in the network at its original position. It is trivial for end users to become aware of the new facility they should direct their traffic demands (e.g. a message is sent when a merging decision occurs).

Let $s_2 \overset{t}{\hookrightarrow} s_1$ denote facility s_1 being merged with facility s_2 at time t, with facility s_1 remaining in the network after the merging. For example, for the case shown in Fig. 3, if a facility merging $s_2 \overset{t}{\hookrightarrow} s_1$ occurs at time t, then facility s_2 no longer exists in the network, i.e., $\mathcal{F}_{t+1} = \mathcal{F}_t \setminus \{s_2\}$, and some nodes in $\mathbb{T}_{s_2}^{\mathcal{F}_t}$ (which were served by facility s_2), are now served by facility s_1 with the rest by facility s, depending on which is the closest to each node. Let $\Phi_{s_2 \overset{t}{\hookrightarrow} s_1}(z)$ be the set of nodes in $\mathbb{T}_{s_2}^{\mathcal{F}_t}$ whose distance to node z is smaller than the distance to any other node at time t, then $\Phi_{s_2 \overset{t}{\hookrightarrow} s_1}(z) = \{u \in \mathbb{T}_{s_2}^{\mathcal{F}_t} : d(u,z) = \min_{\forall s \in \mathcal{F}_{t+1}} d(u,s)\}$, and $\cap_{\forall z \in \mathcal{F}_{t+1}} \Phi_{s_2 \overset{t}{\hookrightarrow} s_1}(z) = \emptyset$.

Lemma 4. *For a facility merging $s_2 \overset{t}{\hookrightarrow} s_1$ and any facility node $z \in \mathcal{F}_{t+1}$, then $\mathbb{T}_z^{\mathcal{F}_t} \cap \Phi_{s_2 \overset{t}{\hookrightarrow} s_1}(z) = \emptyset$, $\mathbb{T}_z^{\mathcal{F}_{t+1}} = \mathbb{T}_z^{\mathcal{F}_t} \cup \Phi_{s_2 \overset{t}{\hookrightarrow} s_1}(z)$ and $\cup_{\forall z \in \mathcal{F}_{t+1}} \Phi_{s_2 \overset{t}{\hookrightarrow} s_1}(z) = \mathbb{T}_{s_2}^{\mathcal{F}_t}$.*

Proof. The proof is included in [10]. □

Theorem 2. *For a facility merging, $s_2 \overset{t}{\hookrightarrow} s_1$, $\mathcal{C}^{\mathcal{F}_{t+1}} < \mathcal{C}^{\mathcal{F}_t}$ is satisfied iff $\mathfrak{c} > \sum_{\forall z \in \mathcal{F}_{t+1}} \sum_{\forall u \in \Phi_{s_2 \overset{t}{\hookrightarrow} s_1}(z)} \lambda(u)(d(u,z) - d(u,s_2))$.*

Proof. The proof is included in [10]. □

Similar to Theorem 1, Theorem 2 is based on non-local information, and therefore, a local information based condition is required as derived below.

Proposition 2. *For a facility merging $s_2 \overset{t}{\hookrightarrow} s_1$, if $\Lambda^{\mathcal{F}_t}(s_2)d(s_2,s_1) < \mathfrak{c}$, then $\mathcal{C}^{\mathcal{F}_{t+1}} < \mathcal{C}^{\mathcal{F}_t}$ is satisfied.*

Proof. The proof is included in [10]. □

Proposition 2 motivates the following facility merging rule:

Facility merging: facility merging $s_2 \overset{t}{\hookrightarrow} s_1$ occurs iff $\Lambda^{\mathcal{F}_t}(s_2)d(s_2,s_1) < \mathfrak{c}$.

Similar to facility replication, the proposed facility merging is based on local information and, particularly, on monitoring the aggregate traffic load at the facility node. Note that the distance among the merging facilities should also be known, but this does not necessarily require global knowledge. For example, even if a control message exchanging mechanism is introduced, it is expected to be scalable given that the number of facilities is generally small.

As before, this merging mechanism is also conservative, in the sense that there might be cases where $\mathcal{C}^{\mathcal{F}_{t+1}} < \mathcal{C}^{\mathcal{F}_t}$ is satisfied but merging does not occur (Proposition 2 may not be satisfied). It is also interesting to note the symmetry of replication and merging, which ensures there will be no undesired loops (i.e., no simultaneous satisfaction of the conditions for replication and merging). These local information based approaches motivate the proposal of a combined (replication and merging) policy in the following section for reducing the overall cost in cloud computing networks.

5. Proposed s-UFL policy

Assume that at time $t = 0$ there will be $|\mathcal{F}_0| \geq 0$ service facilities in the cloud computing network at some arbitrarily selected initial locations. The facilities are allowed to migrate according to facility migration [9]. When completed, i.e., the facility migration condition is no longer satisfied, facility replication occurs if the conditions are satisfied. If facility replication does occur, facility migration is resumed. If facility replication is not satisfied and the condition of facility merging is satisfied, then facility merging occurs. If any merging occurs, the process repeats. The details of the proposed s-UFL policy are presented in Policy 1 , for time step t.

As discussed above, the conditions for replication and merging cannot be simultaneously satisfied. Some properties of the s-UFL policy are investigated below.

Proposition 3. *According to the s-UFL policy, condition $\mathfrak{c} < \frac{\Lambda^{\mathcal{F}_t}(s)}{2} w(s,s')$ is necessary (not sufficient) for a facility replication $s \overset{t}{\mapsto} s'$ to occur.*

Proof. The proof is included in [10]. □

Proposition 3 provides a necessary but not sufficient condition for replication to occur and, particularly, provides for an upper bound of the value of $\mathfrak{c}$. As link weight $w(s,s')$ decreases, a facility replication is less likely to occur. $\Lambda^{\mathcal{F}_t}(s)$ corresponds to the aggregate traffic load accumulated at node s. As the number of facilities increases in the network, each facility node serves a decreasing number of nodes, hence

Policy 1: The proposed s-UFL policy.

```
for ∀s ∈ F_t do
    migration=FALSE                          ▷Migration part
    for ∀s' ∈ S^{F_t}(s) do
        if Λ^{F_t}(s') > Λ^{F_t}(s)/2 then
            facility migration s --t--> s' occurs
            F_{t+1} = F_t ∪ {s'} \ {s}
            migration=TRUE
    if !migration then
        replication=FALSE                    ▷Replication part
        for ∀s' ∈ S^{F_t}(s) ∧ !replication do
            if Λ^{F_t}(s')w(s,s') > c then
                facility replication s --t--> s' occurs
                F_{t+1} = F_t ∪ {s'}
                replication = TRUE
    if !migration ∧ !replication then
        meging=FALSE                         ▷Merging part
        for ∀s'' ∈ F_t \ {s} ∧ !merging do
            if Λ^{F_t}(s'')d(s,s'') < c then
                facility merging s'' --t--> s occurs
                F_{t+1} = F_t \ {s''}
                merging = TRUE
```

the aggregate traffic load decreases. Eventually, a point is reached where no more facilities are created. Another interpretation is that $\frac{\Lambda^{\mathcal{F}_t}(s)}{2}w(s,s')$ corresponds to the cost benefit for nodes $u \in \mathbb{T}^{\mathcal{F}_t}(s')$ when facility replication $s \overset{t}{\mapsto} s'$ occurs. This should be ensured to be greater than the cost of hosting a new facility (i.e., $\mathfrak{c}$) for a facility replication to occur.

Proposition 4. *According to the* s*-UFL policy, condition* $\Lambda^{\mathcal{F}_t}(s'') < \frac{\Lambda^{\mathcal{F}_t}(s)w(s,s')}{2d(s,s'')}$ *is necessary (not sufficient) for a facility merging* $s'' \overset{t}{\hookrightarrow} s$ *to occur.*

Proof. The proof is included in [10]. □

Proposition 4 provides for a necessary but not sufficient condition for merging to occur and, particularly, provides for an upper bound of the value of $\Lambda^{\mathcal{F}_t}(s'')$, i.e., $\frac{\Lambda^{\mathcal{F}_t}(s)w(s,s')}{2d(s,s'')}$. The traffic demands at node s are related to those of node s''. As distance $d(s,s'')$ increases, the upper bound decreases and it is more likely for a facility merging to occur.

The previous propositions revealed some interesting aspects of the s-UFL policy. A key question is the behavior of the s-UFL policy when the optimal positions (that is the solution of the UFL problem) have been reached. As shown below, the facilities stop moving, replicating, or merging when the optimal positions are reached.

Lemma 5. *When* $|\mathcal{F}_t| = |\mathcal{O}|$, *then* $\mathcal{C}^{\mathcal{F}_{t+1}} > \mathcal{C}^{\mathcal{O}}$ *is satisfied.*

Proof. The proof is included in [10]. □

Theorem 3. *If* $|\mathcal{F}_t| = |\mathcal{O}|$, *then no facility replication or facility merging occurs under the* s*-UFL policy, provided that the migration moves the facilities to the optimal position.*

Proof. The proof is included in [10]. □

There is a limitation with respect to the proposed s-UFL policy since its distributed nature allows for simultaneous facility replication or facility merging in the network. This is the case when more than one facility simultaneously decides on replication or merging. However, the migration moves the facilities to locations of decreased overall costs.

It should be emphasized that under the assumption of a static environment (i.e., no change of the traffic demands and the network topology), the s-UFL policy converges to a minimum value, larger or equal to the optimal $\mathcal{C}^{\mathcal{O}}$, as indicated by Theorem 3. In general, the fact that no migration, replication or merging occurs without overall cost reduction, ensures the convergence of the proposed policy. This is clearly demonstrated by the simulations results presented below.

6. Simulation

Using omnet++ [18], a simulation platform was developed and topologies of geometric random graphs [19] of 100 nodes in the $[0, 1] \times [0, 1]$ square area were considered for three different values of the connectivity radius, $r_c = 0.25$, 0.30, and 0.35. These relatively small values for r_c allow for the creation of a connected but not dense topology, in the sense that the average number of neighbor nodes is kept small, as opposed to large values of r_c (i.e., close to $\sqrt{2}$), where the topology becomes as dense as the complete graph. The motivation behind the selection of geometric random graphs [19] and these values for r_c, is that the topology of the cloud computing network environment is not expected to be a dense network. For example, fog devices within a fog network are expected to have a limited number of neighbor nodes (data centers or other fog devices).

Thus, the simulation results highlight qualitative aspects to confirm the analytical findings presented earlier. The main aim is to show that cost reduction is achieved when the s-UFL policy is implemented. The approach is primarily to present the results of individual runs and reveal the details of the approach to facilitate identifying the exact details of the behavior of the s-UFL policy that would not be possible with averaged results. However, average results of 10 independent runs are also presented to give a macroscopic view of the proposed policy.

Each node was uniformly assigned a random number, $\lambda_u \in [0, 1]$, and weights were set to 1 for all links in the network. The topology and traffic demands were fixed throughout the simulation. The maintenance cost $\mathfrak{c}$ for hosting a facility was set to 1, unless otherwise stated. Data flows were forwarded in the network over shortest paths derived from Dijkstra's algorithm. Since $\mathcal{C}^{\mathcal{F}_t}$ denotes the cost of the s-UFL policy, then $\mathcal{C}^{\mathcal{O}}$ corresponds to the optimal cost (i.e., the solution of UFL), and $\frac{\mathcal{C}^{\mathcal{F}_t}}{\mathcal{C}^{\mathcal{O}}}$ shows how close the s-UFL policy is to optimal, i.e., $\frac{\mathcal{C}^{\mathcal{F}_t}}{\mathcal{C}^{\mathcal{O}}} \to 1$. $\mathcal{T}$ is the end of each simulation run (termination of the s-UFL policy).

Two scenarios were considered to investigate the s-UFL policy effects. The first started with a small number of

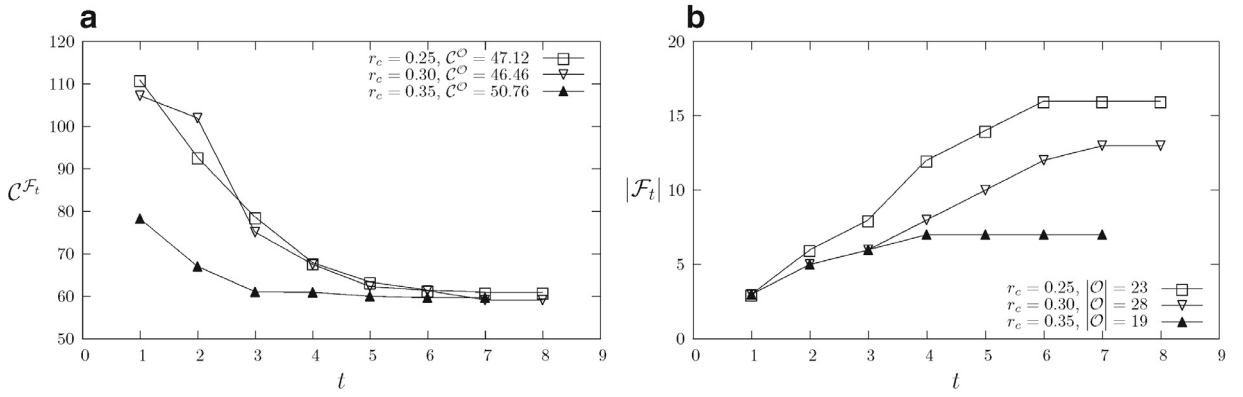


Fig. 4. Simulation for the replication part of the s-UFL policy. (a) Cost $\mathcal{C}^{\mathcal{F}_t}$ as a function of time t. (b) Number of facilities $|\mathcal{F}_t|$ as a function of time t.

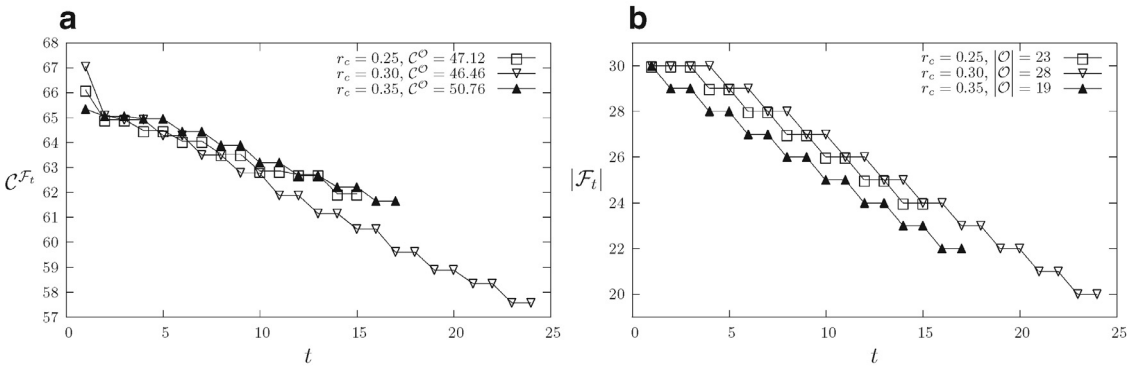


Fig. 5. Simulation results for the merging part of the s-UFL policy. (a) Cost $\mathcal{C}^{\mathcal{F}_t}$ as a function of time t. (b) Number of facilities $|\mathcal{F}_t|$ as a function of time t.

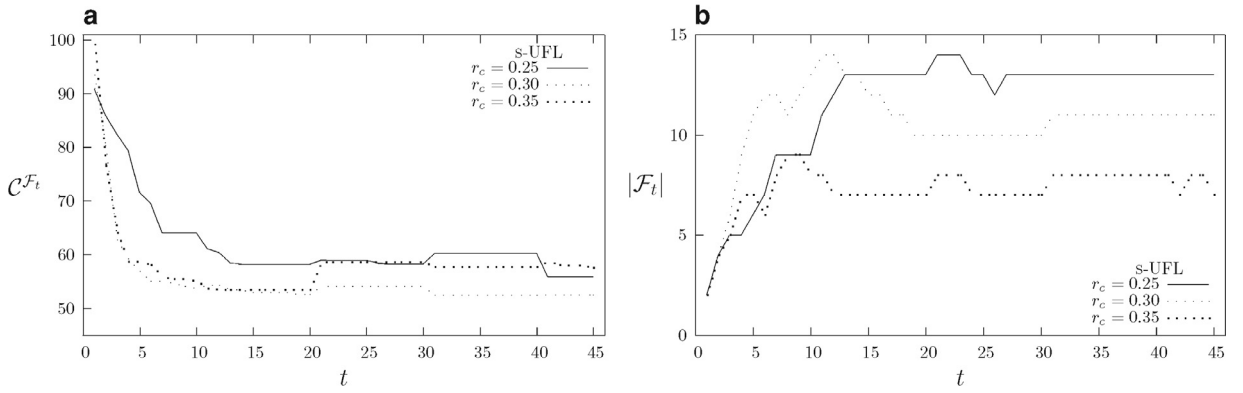


Fig. 6. Simulation results for the s-UFL policy when traffic load is re-initialized for half the network nodes every 10 time units. (a) Cost $\mathcal{C}^{\mathcal{F}_t}$ as a function of time t. (b) Number of facilities $|\mathcal{F}_t|$ as a function of time t.

facilities (3), and focuses on the replication aspects, the second started with a large number of facilities (30), and focuses on merging aspects, in arbitrarily selected nodes.

The results for the first scenario (replication) are presented in Fig. 4. Fig. 4a shows the cost, $\mathcal{C}^{\mathcal{F}_t}$, as a function of time t. Optimal values, $\mathcal{C}^{\mathcal{O}}$ and $|\mathcal{O}|$, are shown for each topology. Cost decreases between 20% and 44%. Fig. 4b shows that the number of facilities increases to a constant, less than optimal. Note that each facility replication (Fig. 4b)corresponds to an overall cost decrement (Fig. 4a).

Simulation results in Fig. 5 show the merging behavior. As time increases, overall cost decrease (Fig. 5a), and the number of facilities, decrease (Fig. 5b) due to merged facilities. As previously discussed, for each facility merging cost $\mathcal{C}^{\mathcal{F}_t}$ decreases between 6% and 13%.

Fig. 6 shows the case where user traffic load (arbitrarily) changes as time passes. Consequently, the s-UFL policy migrates, replicates, or merges facilities. This allows a closer look at the proposed policy behavior. In particular, starting from one facility in the network, half the nodes of the

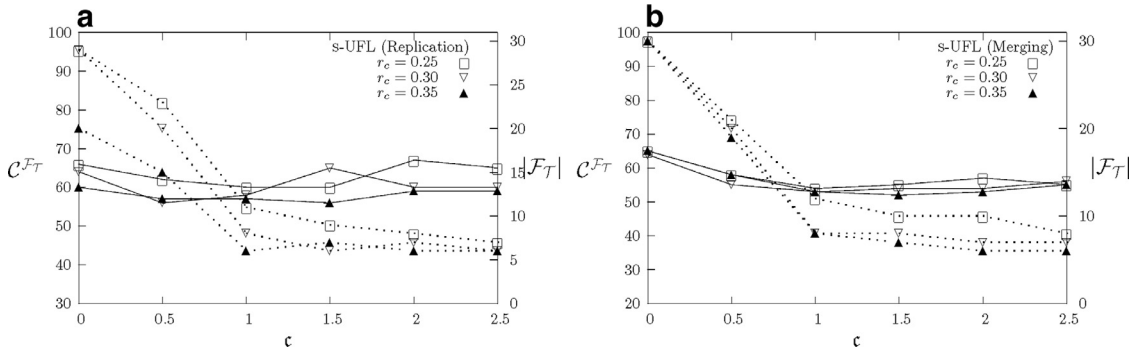


Fig. 7. Simulation results for cost $\mathcal{C}^{\mathcal{F}_\mathcal{T}}$ (dense lines) and number of facilities $|\mathcal{F}_\mathcal{T}|$ (dotted lines) at termination time $\mathcal{T}$ as a function of maintenance cost c averaged over 10 independent runs. (a) The replication part of the s-UFL policy. (b) The merging part of the s-UFL policy.

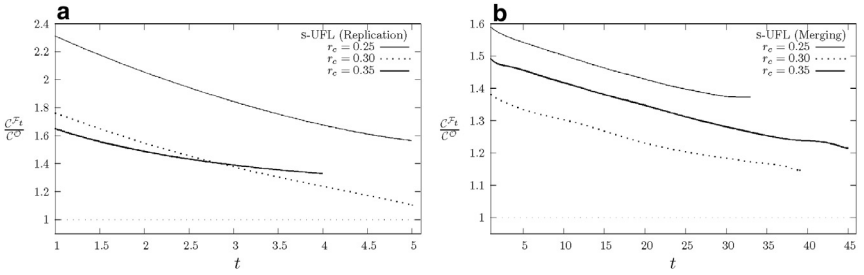


Fig. 8. Simulation results for $\frac{\mathcal{C}^{\mathcal{F}_t}}{\mathcal{C}^{\mathcal{O}}}$, as a function of time t averaged over 10 independent runs. (a) The replication part of the s-UFL policy. (b) The merging part of the s-UFL policy.

network reinitialize their traffic load every 10 time slots. There is an immediate overall cost decrement for each case (Fig. 6a) and every reinitialization may or may not trigger migration, or merging following the s-UFL policy (Fig. 6b). For example (Fig. 6b), after reinitialization at $t = 20$ there is no change of the number of facilities for the case $r_c = 0.30$, even though, as shown in Fig. 6a, this produced a (small) cost increment.

Fig. 6b, for $t = 0, \ldots, 9$, shows that the number of facilities does not increase monotonically as it is the case for $r_c = 0.30$ and 0.35. Following the s-UFL policy, the initial facility has been replicated and as the facilities move in the network (i.e., migration) they eventually reach positions such that merging is required to further decrease the overall cost. The s-UFL policy requirement here is to monotonically reduce the overall cost, as in Fig. 6a.

Fig. 7 shows simulation results for cost, $\mathcal{C}^{\mathcal{F}_\mathcal{T}}$, (dense lines) and the number of facilities, $|\mathcal{F}_\mathcal{T}|$, (dotted lines) as a function of the maintenance cost, c. These are averaged values for the 10 independent runs (the 95% confidence interval is small and not depicted). As c increases, the number of facilities in the network decreases. This is expected since large values of c correspond to large costs for supporting a service facility. It is interesting that overall cost does not depend on c for replication and merging of the s-UFL policy. This is due to high maintenance cost, resulting in a smaller number of facilities in the network, and is compensated by the additional cost for longer distances traveled by data packets within the network.

Fig. 8 shows $\frac{\mathcal{C}^{\mathcal{F}_t}}{\mathcal{C}^{\mathcal{O}}}$ as a function of time, averaged over 10 independent runs (as before, the 95% confidence interval is small and not depicted). $\frac{\mathcal{C}^{\mathcal{F}_t}}{\mathcal{C}^{\mathcal{O}}}$ decreases towards its minimum (assumed to be $t = \mathcal{T}$), for replication and merging. Cost reduction ranges from 7% to 35% for the replication part and from 13% to 20% for the merging part of the s-UFL policy.

The cost reduction of the proposed simple and easily implemented s-UFL policy, indicates the level of support for migration costs (e.g. moving a virtual machine from one data center to another). As discussed above, study of the migration costs is beyond the aims of this work and a future work plan, motivated by the cost reduction observed here, is to extend the analytical modeling within the context of the cloud computing network architecture proposed here.

7. Previous related work

There is extensive literature with respect to cloud computing. A comprehensive survey was presented by Zhang et al. [1]. Another by Fernando et al.[2] focuses on mobile clouds since the proliferation of smart devices (e.g. tablets, smartphones) constantly increases user demands in a dynamic way given the portability of the devices. A Cisco report from 2011 [20] expected a significant increase of end-systems by 2020. The requirement for intermediate networks (i.e., fog computing), at the edge of the cloud, was proposed by Bonomi et al. [4] in 2012 and more recently in 2014 [5]. Bonomi et al. [5] also mention scalability problems that arise as the size of the network increases. A different approach regarding fog computing was proposed by Vaquero and Rodero-Merino [14]. Hong et al. [6] focused on mobile fog devices called fog computing nodes, to support cloud computing services, while Valancius et al. [7] proposed Nano Data Centers to create a distributed data center infrastructure using service provider controlled home gateways.

Migration was proposed by Clark et al. [21] as suitable for moving an entire operating system. However, recent works in the area have shown dependencies on a number of factors such as the amount of data to be copied [15], time delay issues [16], and the number of users [17].

Several researchers have attempted to improve performance by formulating and solving relevant optimization problems. For example, Liu et al. [22] focused on mobile cloud computing and resource allocation, while Wang et al. [23] focused on bulk data transfers in data centers. From a different perspective, Reaz et al. [24] determined how the cloud components should be placed in a wireless optical broadband access network and they formulated (and solved) resource optimization as a mixed-integer linear program. Meng et al. [25] proposed a traffic aware virtual machine placement problem to improve scalability.

Even though many research works focus on optimization problems in cloud computing environments, they differ from the current paper that focuses on network aspects from a facility location theory point of view. Although Tso et al. [26] followed a scalable facility location approach, their focus was on the reduction of network costs within a data center as opposed to the reduction of overall network costs. Similarly, Meng et al. [27] formulated a facility location problem to improve scalability in data centers.

It is well-known that facility location problems, such as UFL, are NP-hard [8]. A variety of approximation algorithms have been proposed, incorporating various techniques such as rounding of linear programs [28], local search [29,30], and primal-dual methods [31]. However, these approaches do not scale in the dynamic environment of cloud computing due to their requirements for global information.

Ragusa et al. [32] proposed an heuristic approach for partitioning the network into a number of clusters allowing them to be partitioned or merged in response to dynamic network changes based on empirically selected threshold values. Laoutaris et al. [33] addressed the facility location problem in a distributed manner solving UFL inside suitably defined areas. In contrast to these works, the approach here relies on strictly local information, passively collected by the facilities themselves, as opposed to requiring the deployment of a mechanism providing the demand and topology information within each cluster. The present work employs a policy of replicating (part of a previous version of this work [34]), merging, and moving service facilities in the cloud computing network.

8. Conclusions

In this paper, the problem of virtual machine placement using information locally available at the nodes hosting them within a cloud computing network was addressed and studied. Based on a cloud computing network architecture, virtual machine replication and a complementary merging mechanisms were proposed and analyzed, capable of exploiting the locally available information to reduce the communication and support cost in the network. Virtual machine replication and merging, along with migration, were the basis for a proposed (combined) scalable facility location policy: the s-UFL policy. The efficiency of this elastic policy and its characteristics were studied and discussed. The proposed policy is simple and easily deployed, since it is based on a simple monitoring mechanism estimate aggregate traffic load of incoming/outgoing traffic. Simulation results were presented, which confirmed the analytical findings and demonstrated a significant cost reduction when the s-UFL policy was implemented.

Appendix A. Notation

$s \xrightarrow{t} s'$	Facility migration from node s to the neighbor node s' at time t
$s \overset{t}{\mapsto} s'$	Facility replication from node s to the neighbor node s' at time t
$s_2 \overset{t}{\hookrightarrow} s_1$	Facility merging of facility nodes s_1 and s_2 at time t
$\mathfrak{c}$	Maintenance cost for hosting a single facility
$\mathcal{C}^{\mathcal{F}_t}$	Cost for a set of facilities $\mathcal{F}_t$ (overall)
$\mathcal{C}^{\mathcal{F}_t}(s)$	Cost for facility node s
$d(u,v)$	Distance over a shortest path among nodes u and v
$\mathcal{F}_t$	Set of facility nodes at time t
$\mathcal{O}$	Optimal facility set (UFL solution)
$\mathcal{S}(u)$	Neighbor nodes of u
$\mathbb{S}^{\mathcal{F}_t}(u)$	Neighbor nodes of facility node u served by node u
$\mathbb{T}_s^{\mathcal{F}_t}$	Subtree accumulating traffic load for node s
$w(u,v)$	Weight of link (u,v)
$\lambda(u)$	Traffic load for node u
$\Lambda^{\mathcal{F}_t}(u)$	Aggregate traffic load for node u over $\mathbb{T}_s^{\mathcal{F}_t}$
$\varphi^{\mathcal{F}_t}(u)$	The closest facility node to node u at time t
$\Phi_{s \overset{t}{\mapsto} s'}$	Nodes closer to s' than to their (former) facility node
$\Phi_{s \overset{t}{\mapsto} s'}(z)$	Nodes closer to s' than to their (former) facility node formerly served by node z
$\Phi_{s_2 \overset{t}{\hookrightarrow} s_1}(z)$	Nodes close to facility z when initially being served by facility s_2 after merging with facility s_1

References

[1] Q. Zhang, L. Cheng, R. Boutaba, Cloud computing: state-of-the-art and research challenges, J. Internet Serv. Appl. 1 (1) (2010) 7–18.

[2] N. Fernando, S.W. Loke, W. Rahayu, Mobile cloud computing: a survey, Futur. Gener. Comput. Syst. 29 (1) (2013) 84–106. Including Special section: AIRCC-NetCoM 2009 and Special section: Clouds and Service-Oriented Architectures.

[3] Y. Choi, S. Lee, J. Kim, Y. Kim, H. Pak, G. Moon, J. Ra, Y.-G. Jung, The method to secure scalability and high density in cloud data-center, Inf. Syst. 48 (2015) 274– 278.

[4] F. Bonomi, R. Milito, J. Zhu, S. Addepalli, Fog computing and its role in the internet of things, in: Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing, MCC '12, ACM, New York, NY, USA, 2012, pp. 13–16.

[5] F. Bonomi, R. Milito, P. Natarajan, J. Zhu, Fog computing: a platform for internet of things and analytics, Big Data and Internet of Things: A Roadmap for Smart Environments, Springer, 2014, pp. 169–186.

[6] K. Hong, D. Lillethun, U. Ramachandran, B. Ottenwälder, B. Koldehofe, Mobile fog: a programming model for large-scale applications on the internet of things, in: Proceedings of the Second ACM SIGCOMM Workshop on Mobile Cloud Computing, MCC '13, ACM, New York, NY, USA, 2013, pp. 15–20.

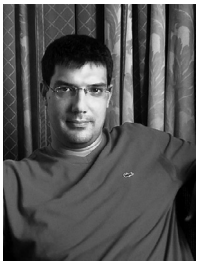
[7] V. Valancius, N. Laoutaris, L. Massoulié, C. Diot, P. Rodriguez, Greening the internet with nano data centers, in: Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies, ACM, 2009, pp. 37–48.

[8] P. Mirchandani, R. Francis, Discrete Location Theory, John Wiley & Sons, 1990.

[9] K. Oikonomou, I. Stavrakakis, Scalable service migration in autonomic network environments, IEEE J. Sel. Areas Commun. 28 (1) (2010) 84–94.
[10] E. Kavvadia, S. Sagiadinos, K. Oikonomou, G. Tsioutsiouliklis, A. Sonia, Proofs regarding elastic virtual machine placement in cloud computing network environments, URL http://eliki.ionio.gr/Repository/TECH-REP-APPENDIX-OCT-2015-1.pdf 2015.
[11] M. Al-Fares, A. Loukissas, A. Vahdat, A scalable, commodity data center network architecture, SIGCOMM Comput. Commun. Rev. 38 (4) (2008) 63–74.
[12] C. Guo, H. Wu, K. Tan, L. Shi, Y. Zhang, S. Lu, Dcell: a scalable and fault-tolerant network structure for data centers, in: Proceedings of the ACM SIGCOMM 2008 Conference on Data Communication, SIGCOMM '08, ACM, New York, NY, USA, 2008, pp. 75–86, doi:10.1145/1402958.1402968.
[13] A. Agust–Torra, F. Raspall, D. Remondo, D. Rincn, G. Giuliani, On the feasibility of collaborative green data center ecosystems, Ad Hoc Netw. 25 (Part B) (2015) 565–580. New Research Challenges in Mobile, Opportunistic and Delay-Tolerant Networks Energy-Aware Data Centers: Architecture, Infrastructure, and Communication.
[14] L.M. Vaquero, L. Rodero-Merino, Finding your way in the fog: towards a comprehensive definition of fog computing, SIGCOMM Comput. Commun. Rev. 44 (5) (2014) 27–32.
[15] V. Mann, A. Gupta, P. Dutta, A. Vishnoi, P. Bhattacharya, R. Poddar, A. Iyer, Remedy: network-aware steady state vm management for data centers, in: Networking 2012, Springer, 2012, pp. 190–204.
[16] W. Voorsluys, J. Broberg, S. Venugopal, R. Buyya, Cost of virtual machine live migration in clouds: a performance evaluation, in: Cloud Computing, Springer, 2009, pp. 254–265.
[17] G. Jung, K.R. Joshi, M.A. Hiltunen, R.D. Schlichting, C. Pu, A cost-sensitive adaptation engine for server consolidation of multitier applications, in: Middleware 2009, Springer, 2009, pp. 163–183.
[18] Omnet++ simulation program URL http://www.omnetpp.org, 2015.
[19] M. Penrose, Random Geometric Graphs, Oxford University Press, 2003.
[20] D. Evans, The Internet of Things: How the Next Evolution of the Internet is Changing Everything, vol. 1, CISCO White Paper, 2011.
[21] C. Clark, K. Fraser, S. Hand, J.G. Hansen, E. Jul, C. Limpach, I. Pratt, A. Warfield, Live migration of virtual machines, in: Proceedings of the 2nd Conference on Symposium on Networked Systems Design & Implementation, NSDI'05, 2, USENIX Association, Berkeley, CA, USA, 2005, pp. 273–286.
[22] W. Liu, T. Nishio, R. Shinkuma, T. Takahashi, Adaptive resource discovery in mobile cloud computing, Comput. Commun. 50 (2014) 119–129.Green Networking
[23] Y. Wang, S. Su, A.X. Liu, Z. Zhang, Multiple bulk data transfers scheduling among datacenters, Comput. Netw.s 68 (2014) 123– 137. Communications and Networking in the Cloud.
[24] A.S. Reaz, V. Ramamurthi, M. Tornatore, B. Mukherjee, Cloud-integrated woban: an offloading-enabled architecture for service-oriented access networks, Comput. Netw. 68 (2014) 5– 19. Communications and Networking in the Cloud.
[25] X. Meng, V. Pappas, L. Zhang, Improving the scalability of data center networks with traffic-aware virtual machine placement, in: Proceedings of the 29th Conference on Information Communications, INFOCOM'10, IEEE Press, Piscataway, NJ, USA, 2010, pp. 1154–1162.
[26] F.P. Tso, K. Oikonomou, E. Kavvadia, D.P. Pezaros, Scalable traffic-aware virtual machine management for cloud data centers, in: Proceedings of the IEEE 34th International Conference on Distributed Computing Systems (ICDCS), IEEE, 2014, pp. 238–247.
[27] X. Meng, V. Pappas, L. Zhang, Improving the scalability of data center networks with traffic-aware virtual machine placement, in: Proceedings of IEEE INFOCOM, IEEE, 2010, pp. 1–9.
[28] M. Charikar, S. Guha, É. Tardos, D.B. Shmoys, A constant-factor approximation algorithm for the k-median problem, in: Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing, ACM, 1999, pp. 1–10.
[29] M.R. Korupolu, C.G. Plaxton, R. Rajaraman, Analysis of a local search heuristic for facility location problems, J. Algorithms 37 (1) (2000) 146–188.
[30] V. Arya, N. Garg, R. Khandekar, A. Meyerson, K. Munagala, V. Pandit, Local search heuristics for k-median and facility location problems, SIAM J. Comput. 33 (3) (2004) 544–562.
[31] K. Jain, V.V. Vazirani, Primal-dual approximation algorithms for metric facility location and k-median problems, in: Proceedings of the 40th Annual Symposium on Foundations of Computer Science, IEEE, 1999, pp. 2–13.
[32] C. Ragusa, A. Liotta, G. Pavlou, An adaptive clustering approach for the management of dynamic systems, IEEE J. Sel. Areas Commun. 23 (12) (2005) 2223–2235.
[33] N. Laoutaris, G. Smaragdakis, K. Oikonomou, I. Stavrakakis, A. Bestavros, Distributed placement of service facilities in large-scale networks, in: Proceedings of the 26th IEEE INFOCOM International Conference on Computer Communications, IEEE, 2007, pp. 2144–2152.
[34] K. Oikonomou, G. Tsioutsiouliklis, S. Aissa, Scalable facility placement for communication cost reduction in wireless networks, in: Proceedings of IEEE International Conference on Communications (ICC), IEEE, 2012, pp. 5118–5123.

Eleni Kavvadia received her diploma in Electrical and Computer Engineering, with specialization in telecommunications and networks, from the National Technical University of Athens (NTUA) in 2000. In September 2011 she received her postgraduate degree M.Sc. in Informatics, in the subject area of networks, from Department of Informatics, Ionian University, Greece. She is currently a Ph.D.candidate in the field of computer networks at the same university. Her current research interests comprise networks and virtual environments in general. She has wide experience from participating as a software developer, and technical manager in EU and national funded development projects in the field of computer networks and information systems. She is currently chief of Networks & Telecommunications Services of the Ionian University.

Spyros Sagiadinos received his degree in Informatics from the Technological Educational Institute of Athens in 1994. From 2010 is an undergraduate student at Department of Informatics, Ionian University, Greece. From 1996 he is an employ at the Ministry of Public Order as Systems Manager. He has wide experience in software development – using various languages (Javascript, Python, PHP, C etc.) – and database (PostgreSQL) administration. He is currently interested in Web development (AngularJS framework) and NoSQL (MongoDB) databases .



Konstantinos Oikonomou received his diploma in Computer Engineering and Informatics from University of Patras in 1998. In September 1999 he received his postgraduate degree: M.Sc. in Communication and Signal Processing from the Electrical and Electronic Engineering Department, Imperial College (London). He received his Ph.D. degree in 2004 from the Informatics and Telecommunications Department, University of Athens, Greece. His Ph.D. thesis focuses on Topology-Unaware TDMA MAC Policies for Ad Hoc Networks. Between December 1999 and January 2005 he was employed at INTRACOM S.A, as a research and development engineer. His current interests involve medium access control in ad hoc networks, performance issues in autonomic networks, information dissemination, service discovery and facility location. He has a long experience regarding specification processes for MAC/DLC wireless systems and he has been co-ordinating a number of EC research projects in the computer networks research area. He has been a reviewer and TPC member of numerous conferences and journals in the area. He is currently an Assistant Professor in Computer Networks at Department of Informatics of the Ionian University, Greece. He holds an award for the best paper from the Hawaii International Conference on System Service and another one from the Hellenic Mathematical Society.

Giorgos Tsioutsiouliklis has acquired his B.Sc. degree in Computer Science in the Ionian University in Corfu, Greece. He continued to his M.Sc. degree in Communication Networks in the same institute. Currently he is researching real-world applications of distributed NoSQL databases.

Sonia Aïssa [S'93-M'00-SM'03] received her Ph.D. degree in Electrical and Computer Engineering from McGill University, Montreal, QC, Canada, in 1998. Since then, she has been with the Institut National de la Recherche Scientifique – Energy, Materials and Telecommunications (INRS-EMT), University of Quebec, Montreal, QC, Canada, where she is a Full Professor. From 1996 to 1997, she was a Researcher with the Department of Electronics and Communications of Kyoto University, Kyoto, Japan, and with the Wireless Systems Laboratories of NTT, Kanagawa, Japan. From 1998 to 2000, she was a Research Associate at INRS-EMT, Montreal. From 2000 to 2002, while she was an Assistant Professor, she was a Principal Investigator in the major program of personal and mobile communications of the Canadian Institute for Telecommunications Research, leading research in radio resource management for wireless networks. From 2004 to 2007, she was an Adjunct Professor with Concordia University, Montreal. In 2006, she was Visiting Invited Professor with the Graduate School of Informatics, Kyoto University, Japan. Her research interests lie in the area of wireless and mobile communications, and include radio resource management, cross-layer design and optimization, design and analysis of multiple antenna (MIMO) systems, cognitive and cooperative transmission techniques, and performance evaluation, with a current focus on cellular and cognitive radio networks. Dr. Aïssa is the Founding Chair of the IEEE Women in Engineering Affinity Group in Montreal, 2004–2007; acted as Technical Program Leading Chair or Cochair of the Wireless Communications Symposium at IEEE ICC in 2006, 2009, 2011 and 2012; PHY/MAC Program Chair of the 2007 IEEE WCNC; Technical Program Committee Cochair of the 2013 IEEE VTC-spring; and Symposia Chair of the 2014 IEEE Globecom. Her main editorial activities include: Editor, IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, 2004-2012; Associate Editor, IEEE COMMUNICATIONS MAGAZINE, 2004–2009; Technical Editor, IEEE WIRELESS COMMUNICATIONS MAGAZINE, 2006-2010; and Associate Editor, Wiley Security and Communication Networks Journal, 2007–2012. She currently serves as Area Editor for the IEEE Transactions on Wireless Communications and as Technical Editor for the IEEE Communications Magazine. Awards to her credit include the NSERC (Natural Sciences and Engineering Research Council of Canada) University Faculty Award in 1999; the Quebec Government FQRNT Strategic Faculty Fellowship in 2001–2006; the INRS-EMT Performance Award multiple times since 2004, for outstanding achievements in research, teaching and service; and the Technical Community Service Award from the FQRNT Centre for Advanced Systems and Technologies in Communications, in 2007. She is corecipient of five IEEE Best Paper Awards and of the 2012 IEICE Best Paper Award; and recipient of NSERC Discovery Accelerator Supplement Award. She is a Distinguished Lecturer of the IEEE Communications Society (ComSoc) and an Elected Member of the Board of Governors of ComSoc.