Virtual Functions Placement with Time Constraints in Fog Computing: a Matching Theory Perspective

Francesco Chiti, *Senior Member, IEEE,* Romano Fantacci, *Fellow, IEEE,* Federica Paganelli, *Member, IEEE,* and Benedetta Picano, *Student Member, IEEE*

***Abstract*—This paper proposes two virtual function (VFs) placement approaches in a Fog domain. The considered solutions formulate a matching game with externalities, aiming at minimizing both the worst application completion time and the number of applications in outage, i.e., the number of applications with an overall completion time greater than a given deadline. The first proposed matching game is established between the VFs set and the Fog Nodes (FNs) set by taking into account the ordered sequence of services (i.e., chain) requested by each application. Conversely, the second proposed method overlooks the applications service chain structure in formulating the VF placement problem, with the aim at lowering the computation complexity without loosing the performance. Furthermore, in order to complete our analysis, the stability of the reached matchings has been theoretically proved for both the proposed solutions. Finally, performance comparisons of the proposed MT approaches with different alternatives are provided to highlight the superior performance of the proposed methods.**

***Index Terms*—Fog Computing, Virtual Functions Placement, Matching Theory.**

I. Introduction

FOG Computing (FC) is a term coined by Cisco referring to the evolution of the Cloud paradigm from a centralized model towards a *distributed* model of *heterogeneous*, *federated* Clouds located at the edge of the network [1], that is a "Cloud close to the ground" [2].

According to this vision, the widespread adoption of the virtualization and *infrastructure as a service* models at the network edge lets envisage a landscape of heterogeneous service capabilities and resources pervasively distributed and interconnected close both to end users and physical objects. FC is, thus, a highly virtualized platform that provides processing, storage, and networking capabilities made available by *edge* nodes (e.g., mobile devices, routers, and micro data centers) to support the development of distributed applications. Devices endowed with sensing and actuation capabilities could be considered nodes that offer specialized services (e.g., acquisition of measurements from the environment in which they are deployed and, whenever possible or needed, perform control actions) [3].

FC is expected to support services and applications that demand widely distributed deployments and real-time interactions. Especially in the area of the Internet of Things (IoT), the availability of processing, network and storage resources close to sensors, actuators and end users can help in coping with low-latency application requirements, while also partially relieving the central Cloud from processing big amounts of data.

An application deployed on a Fog infrastructure can be conceived as a set of independent services which cooperate each other to realize the application goal, and typically interact in a *sequential* order, especially in IoT scenarios demanding for *sense-process-actuate* workflows [4], [5]. These services, ultimately composed of atomic services, are expected to be provisioned by software capabilities, i.e., programs, that can be executed on top of a virtualized resource infrastructure. One example is represented by the container-based virtualization [6]. In fact, these novel service provisioning solutions are resulted to be extremely suitable for IoT resource-constrained environments and, more in general, for high dynamic and heterogeneous scenarios. This is due to the flexibility provided by these frameworks, since they are not strictly dependent on a given technology or specific application domain, and to their reduced overhead with respect to hypervisor-based virtualization [6]. For instance, Docker-compatible containers are used in Microsoft Azure IoT Edge [7] for deploying computation on devices, while Amazon AWS Greengrass [8] allows deploying and running Lambda functions on the Greengrass core software in edge devices [9]. We refer here to such software capabilities as Virtual Functions (VFs). It is worth clarifying that a VF may serve more than one service and, consequently, more than one application. However, the usage of a VF may differ from service to service, according to specific applications requirements, and, consequently, it requires different computation times and resource usage.

In this context, *orchestration* mechanisms should be put in place to achieve a cost-effective utilization of infrastructural resources for Quality of Service (QoS) aware deployment and management of application components (i.e., services) [10]. A major problem is, thus, how to effectively distribute services to support composite applications provisioning on top of a Fog resource infrastructure.

Although placement problems have been extensively tackled in the Cloud computing domain for the optimal deployment of Virtual Machines (VMs) [11] and in the networking domain for optimal placement of Virtual Network Functions (VNFs) for service chaining [12], the problem of service placement on top of Fog resources infrastructures has been marginally

F. Chiti, R. Fantacci and B. Picano are with the Department of Information Engineering, University of Florence, Florence, Italy e-mail: name.surname@unifi.it.

F. Paganelli is with the Department of Computer Science, University of Pisa, Pisa, Italy, e-mail: federica.paganelli@unipi.it

addressed so far [13].

Placement on FC infrastructures differs from previous work in Cloud computing domain in that a substantial amount of functions are carried out even close to the end user on a distributed network of small to medium size nodes with intermittent connectivity with the Cloud [14]. FC does not necessarily requires virtualization, however, FC and virtualizaton, together with related paradigms, such as Network Function Virtualization (NFV) in the networking domain, may complement and benefit each other. With respect to placement in the NFV domain, FC copes with a wider set of functions types, such as processing, storage, control, and not merely with networking ones and it typically has to accomodate delay sensitive application requirements (e.g., real time data analytics), in terms of on-demand provisioning of resources and low response time [15].

In [4], [16]–[18] some deployment strategies are proposed, that focus on the effective usage of infrastructure resources in a distributed scenario, typically including Fog as well as Cloud resources. However, the increased demand of applications fulfilling specific delay constraints have been only recently considered in the definition of the deployment strategies [5], [13], [19].

In this perspective, we propose two VFs placement strategies aiming at minimizing both the worst application overall completion time and the number of applications that suffer an overall completion time greater than their deadlines (*outage*). Our approaches apply the principles of matching theory (MT), which is a powerful mathematical tool that optimally matches the elements of two distinct sets, on the basis of the individual preferences of each element [20]. According to this, we formulate the problem of VF placement within a FC environment as a matching game with externalities between the VFs and the FNs sets.

Performance results derived by resorting to computer simulations are presented in order to validate the good behavior of the proposed solutions in comparison with different alternatives, i.e., greedy and random algorithms, and a modified version of the potential game presented in [21].

Finally, we provide a theoretical proof of the stability of the proposed placement policies that is usually not a trivial issue in the matching game with externalities class.

The rest of paper is organized as follows. Section II provides an in-depth review of the related literature and illustrates the motivation of this work. In Section III, we propose the system model and the problem formulation. The proposed strategy is presented in Section IV, and in Section V the stability analysis is discussed. In Section VI the experimental results are shown. Finally, the conclusions are drawn in Section VII.

II. Related work

Resource management and allocation problems have been extensively investigated in the fields of Cloud Computing (CC) [22] and, more recently, of NFV [12].

Several approaches have been proposed for addressing the problem of optimizing the use of physical resources in Data Centers to achieve an effective placement of VMs onto physical hosts to account for conflicting requirements on performance and operational costs [22]. Wang et al. [23] address the problem of service composition in a data center network by formulating a multi-objective constraint optimization problem that aims at maximizing an overall QoS utility value, and minimizing network resource utilization. To this purpose, Anchor [24] is a MT based resource management architecture that performs VM allocation requests to servers, using both offline and online algorithms, allowing clients and operators to express a variety of distinct resource management policies.

Herrera and Botero [12] provide a state of the art of resource allocation strategy for the deployment of network services in VNF enabled network infrastructures. Several works proposed approaches for VNF placement and service chaining with the goal of minimizing costs, while fulfilling Service Level Agreements. Liberati et al. [25] propose a stochastic algorithm based on reinforcement learning, that maximizes an expected mapping reward in the long term. The reward function can be formulated to steer the controlled system towards the desired performance (e.g., costs minimization, load balancing, maximization of the acceptance rate). Pham et al. [26] address the problem of VNF placement for deploying service chains on virtual resource infrastructures, while minimizing both traffic and operational costs. They propose an approach combining a sampling-based Markov approximation technique with MT.

Resource management in a FC environment differs from such previous works on several aspects [4], [13]: *i)* FNs have limited resources *ii)* clusters of FNs are typically distributed on a suitable area, and *iii)* computation tasks are typically delay sensitive. Several works have focused on the efficient utilization of resources of the network infrastructure, typically including FC, as well as Cloud resources. Brogi and Forti [4] propose a model for representing FC infrastructures and applications and propose a two-step algorithmic approach for finding eligible deployments for an application into a FC infrastructure. The proposed approach consists in a pre-processing phase aiming at reducing the search space and a backward search procedure that provides an eligible deployment. On the other hand, Foglets [16] is a programming model and software infrastructure for the deployment of applications on FC environments which embodies algorithms and protocol for the discovery and allocation of Fog resources for application deployment. Taneja and Davy [17] propose a network-aware module, mapping algorithm aiming at guaranteeing efficient resource utilization in distributed application deployment in a Cloud-FC environment. Both resource nodes and application modules are sorted according to the available capacity and requirements. Moreover, associations are performed whenever QoS constraints are satisfied. Zhang et al. [18] focus on a resource allocation problem in a FC environment consisting of a large number of FNs deployed by services providers at different locations to provide data services to subscribers. They propose a joint optimization approach that combines Stackelberg game and many-to-many matching to achieve an optimal and stable performance in the resource selection and allocation between the FNs, services providers and subscribers.

To the best of our knowledge, applications delay requirements have been only recently considered in the literature [5], [13], [19]. Specifically, Skarlat et al. propose in [5] a resource model for a FC architecture and also formulate a Fog service placement problem that maximizes utilization of FC resources, while taking into account application QoS constraints, especially deadlines on the execution time. The problem is formulated as an Integer Linear Program. These results have been successively extended in [13] by proposing a genetic-based algorithm heuristics. Finally, Bittencourt et al. compared in [19] the performance of three different resource allocation policies (i.e., concurrent, First Come-First Served, and the delay-priority strategies) in terms of efficient resource utilization and support to application QoS constraints. Differently from [13], [19] and [5], where an hierarchical three-layers Fog network[1] with the Cloud at the highest level of the network is considered, we focus here on a two-layers network, without considering the presence of the Cloud. Furthermore, differently from [19], in which service processing is allowed just one hop away from the origin of the request, we consider the possibility to offload the service computation on FNs that are distant more than one hop from the origin of the request. Finally, as in [13], [19] and [5], we associate at each application a time deadline for the application completion. However, differently from [13], [5], we do not introduce any prioritization mechanism for the applications, and we do not consider the presence of the Cloud in the network. Moreover, by focusing on a homogeneous scenario in terms of classes of devices priorities, we provide a more general planning strategy , able to perform an efficient placement, even without a deep knowledge about the context or the application requests state, generally a critical issue in many allocation strategies.

Summarizing, the main contributions of this paper are the design and analysis of two matching based VFs placement strategies, that allow to minimize both the worst application overall completion time and the outage probability. More in depth this paper:

- formulates a matching game with externalities between the VFs and the FNs sets;
- proposes two different heuristics to solve the problem, one supposing a full-knowledge about the surrounding network, and the other one blind about the network information, with a consequent lower computational complexity;
- provides the theoretical proof of the stability of both the proposed solutions;
- analyzes the performance comparisons of the proposed strategies with some different approaches.

III. Problem Statement

A. Motivating Scenario

To motivate our work, we provide a possible scenario as an example of application deployment in an FC landscape focusing on image processing as in [27]. A camera (or a set of spatially deployed cameras) takes pictures in an area (e.g. town square, park, hall in a commercial centre) and different image processing workflows are required for different application purposes (e.g., visitor identification, abnormal situation detection) so that proper actions are performed if relevant events are detected (e.g. a notification is sent to a security officer). In order to cope with such goals, we can figure out an application composed of the following services: data collection, image filtering (e.g., applying filtering techniques for reducing noise), analytics (e.g. applying object detection techniques), event management (for instance a notification service). Such services are provided by VFs, i.e., programs actually performing the required computation, storage and communication tasks. The problem addressed here consists in properly placing VFs in a network of FNs to accommodate the requirements of a batch of applications, so that the worst application time and the number of applications in outage are minimized.

B. System Model

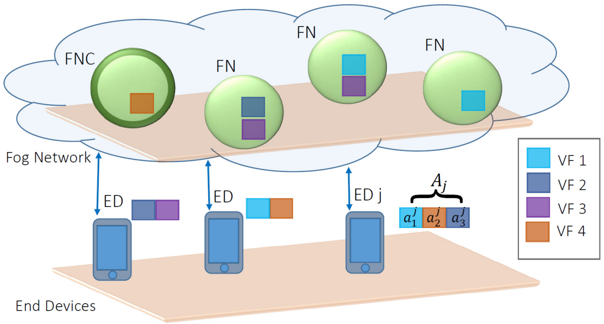


Fig. 1. Reference Scenario. The EDs request to FNs the computation of their applications that are constituted by one or more services, each of which requires a VF.

In our analysis, we refer to the scenario depicted in Figure1, where we assumed to have a set $\mathcal{F}$ of n FNs. Furthermore, we suppose the presence of only one FN, among the n in $\mathcal{F}$, referred as FNs controller (FNC), able to acquire a full knowledge about the FNs capabilities, applications requirements and able to control all the FNs of the network. Likewise, assuming a number of applications equals to m, we denote with $\mathcal{A} = \{A_1, \dots, A_m\}$, the set of applications, where each application A_j is associated to a suitable services chain, i.e., $A_j = \{a^j_1, \dots, a^j_g\}$, for $j = 1, 2, ..., m$, where g is the number of services that compose A_j, and each service a^j_k, $k = 1, \dots, g$ exploits a specific virtual function (VF) belonging to the set $\mathcal{V} = \{v_1, \dots, v_p\}$, where p represents the number of VF types. Moreover, we denote with h_z, for $z = 1, 2, ..., p$, the occurrence frequency of the VF v_z in the set $\mathcal{A}$. Finally, we consider a time constraint for each application, i.e., A_j has to be completely executed within a given time deadline d_j.

Each $v_z \in \mathcal{V}$ can be loaded on a FN $f \in \mathcal{F}$ if it has sufficient computational resource blocks (CRBs) to satisfy the service demand. To this purpose, a VF v_z is associated with the amount r_z of requested CRBs. In addition, we assume that each FN $f \in \mathcal{F}$ is equipped with a central processing unit (CPU), whose capability is expressed in terms of number

[1] Due to the different network architecture adopted in [13] and [5], their genetic and evolutionary approaches cannot be considered as proper benchmark methods for our approach.

of CPU cycles per second c_f and r_f available CRBs, with $0 \leq r_f \leq r_{MAX}$, where r_{MAX} is the maximum number of available CRBs for each FN. As a consequence, we have that the execution time of the k-th service of A_j requiring v_z, on the FN f results in

$$t^j_{k,f,v_z} = \frac{s^j_{k,v_z}}{c_f}, \tag{1}$$

where $s^j_{k,z}$ is the number of cycles required to execute the service a_j using v_z. In order to evaluate the total time demanded by a^j_k to be accomplished on FN f, we have to account for the possibility of experiencing the waiting time ω^j_f, i.e., the time needed to complete the execution of all the services, requesting a v_z, previously assigned to FN f. Furthermore, we consider the communication cost of end-user (ED) j in reaching the desired computation site, i.e., FN f, as given by the sum of two different contributions:

1) the communication cost due to link the ED j and its closest FN f^*, r_{j,f^*}, and given by the ratio between the size in bits of a^j_k and the available transmission rate of the link connecting ED j to FN f^*;
2) the communication cost associated to the intra-cluster connection (i.e., among FNs), $q_{f^*,f}$, given by the sum of the communication cost of the links that service a^j_k has to get across in order to reach FN f from FN f^*.

Hence, the total time $T^j_{k,f,z}$ spent by a^j_k at the FN f before being completed is

$$T^j_{k,f,v_z} = \left(t^j_{k,f,v_z} + \omega^j_f + r_{j,f^*} + q_{f^*,f} \right), \tag{2}$$

From (2), it follows that the overall completion time of application A_j results in

$$C_j = \sum_{k \in \{1,\ldots,g\}} \sum_{f \in \mathcal{F}} T^j_{k,f,v_z} \phi_{k,f}; \tag{3}$$

where $\phi_{k,f}$ is a binary variable equals to 1 if service k is executed on FN f, or to 0 otherwise. In addition to this, we refer in our analysis to the forthcoming 5G network with a transmission rate of 1 Gbit/s to estimate communication costs [28].

C. Problem Formulation

The main objective of this paper is to provide a VFs placement aiming at minimizing both the worst overall application time and the number of applications in outage. In formal terms, the optimization problem can be defined as follows

$$\min_{\mathbf{\Phi}} \max_{j \in \mathcal{A}} C_j \quad and \quad \min |\mathcal{O}| \tag{4}$$

s.t.

$$\sum_{j \in \mathcal{A}} \delta_j = m, \tag{5}$$

$$0 \leq r_f \leq r_{MAX}, \quad \forall f \in \mathcal{F} \tag{6}$$

$$0 \leq \sum_{z \in \mathcal{V}} \tau_{f,z} \leq p, \quad \forall f \in \mathcal{F} \tag{7}$$

$$\text{if} \quad \phi_{k,f} = 1 \Rightarrow \tau_{f,z} = 1, \forall f \in \mathcal{F}, k = 1, \ldots, g, z = 1, \ldots, p. \tag{8}$$

where $\mathbf{\Phi} = \{\phi_{k,f}, \forall k \in A_j, \forall f \in \mathcal{F}\}$, $\mathcal{O}$ is the set of the unsatisfied applications defined as

$$\mathcal{O} = \{A_j \in \mathcal{A} | C_j > d_j, j = 1, \ldots, m\}, \tag{9}$$

and $|\mathcal{O}|$ denotes the number of elements in $\mathcal{O}$. In (9), δ_j is equal to 1 if A_j has all its services accomplished, or to 0 otherwise. Hence, (5) expresses that all the applications belonging to $\mathcal{A}$ have to be completely executed, hence all their services chain has to be computed. Therefore, it implies that any service of each application has to be run on *at least one* FN. Furthermore, constraint (6) imposes a maximum number of available CRBs for each FN in $\mathcal{F}$. Finally, $\tau_{f,z}$ is a binary variable equals to 1 if the FN f already contains v_z, and (7) means that each FN can contain at most one instance of a same v_z. Finally, (8) means that the execution of service k on FN f implies the presence of the VF required by k, i.e., z, on FN f. In order to handle the case of a number of different VFs greater than the available CRBs at each FN, while fulfilling constraint 5, we adopt a discipline of VFs replacement. In other words, this consists in allowing loading VFs at runtime, i.e., during the application execution. We introduce a temporal *penalty* due to the additional loading time required by the VFs runtime replacement. The temporal penalties in which each adopted approach occurred during its execution have been taken into account in the performance evaluation.

IV. Matching algorithm

MT is recently emerged in different application scenarios as a suitable framework to match together the elements of two distinct sets, taking into account the individual preferences of each element of one set toward the elements of the opposite set and vice versa [20], [29], [30]. These preferences express the level of satisfaction of each element of one set in being matched with each element of the other one set according to an appropriate criterion. Under our system assumptions we consider the matching game (MG) logically decentralized, despite its actual implementation is in the FNC. It is straighforward to note that this solution allows to significantly reduce the signaling overhead and speed up the convergence for the two proposed algorithms.

In order to solve the problem (4)-(8) by resorting to MT, we first propose a matching procedure, named Chain Based Matching Algorithm (CBMA) based on the well known Gale-Shapley algorithm (GSA) [20], [30]–[32]. In this case, the MG takes into account the services order, i.e., the position (level) of the services within the chain associated to each application. At each run, the matching procedure considers the VFs requested by the services that occupy the same level in the respective services chains. Thus, the proposed procedure acts at each level of the services chains and the VFs belonging to a given level cannot be allocated until all the ones of the previous level have been matched. Let v_z be the VF requested by the service occupying the k-th level in the A_j application services chain.

The most preferred FN by A_j, i.e., f_j^*, to allocate v_z has to satisfy the following condition:

$$f_j^* = \arg\min_f T^j_{k,f,v_z}. \tag{10}$$

Likewise, the FNs preferences lists are formed by favoring the allocations of VFs associated to applications with a closer deadline. It is straightforward to note that the preferences lists change during the MG. Indeed, the allocation of a VF on a FN impacts on many factors, as the waiting time of that FN, its memory availability and so on, i.e., the presence of a matching influences the future assignments. This type of MGs is referred as MGs with externalities. Since the assumption that the preferences of each element of the matching do not depend on the choices of other elements is not satisfied, the GSA in this class of matching games does not reach a stable matching and, more in depth, there not exist an approach that solves these games and achieves a stable matching. For such reason the study of stability is not trivial and it is extremely relevant.

Summarizing, the CBMA for each level of the applications services chains acts as follows:

1) Each application A_j builds its own preferences list on the FNs set;
2) each application j, requiring a VF z_j, proposes to allocate z_j on its most preferred FN, according to (10);
3) each FN that receives more than one proposal decides which one to accept in accordance with its preference criterion;
4) each FN f that at the previous step has accepted a proposal, decrements its CRBs according to the resource request by the selected VF placement proposal;
5) repeat $1) - 4)$ until all the services of a same level have been processed.
6) repeat $1) - 5)$ until all the services in all the applications services chains have been processed.

In order to define a faster and less complex services placement solution, this paper proposes a novel MT based heuristic between the VFs in $\mathcal{V}$ and the FNs in $\mathcal{F}$, aiming at providing a sub optimal solution to the VFs placement problem (4).

In order to reduce the computation complexity of the service planning approach, without loosing performance, we propose below an alternative MG, named Blind Matching Game (BMG), that removes the constraint to allocate the VFs to FNs according to the order they appear in the applications services chains. In this case, it is only taken into account the occurrences of any VF within the set $\mathcal{A}$. Moreover, the VFs preferences lists are formed with the aim at minimizing the allocation space waste at the FNs.

As previously introduced, a given VF, i.e., v_z, to be loaded on the FN f, needs a CRBs amount equal to r_z. Hence, we have in this case that the utility function of v_z in being loaded on FN f results in

$$U_z(f) = r_f - r_z, \forall f \in \mathcal{F}, z \in \mathcal{V}, \tag{11}$$

where r_f, representing the available CRBs on FN f, given by

$$r_f = r_{MAX} - \sum_{z\in\mathcal{V}} r_z \xi_{z,f}, \tag{12}$$

with $\xi_{z,f}$ a binary function equals to 1 if VF z is present on the FN f, or to 0, otherwise and, as previously introduced, r_{MAX} denoting the maximum number of available CRBs for each FN. In order to build the VFs preference lists, we sort the utility functions of each VF, according to (11), in an ascending order, hence favoring VFs allocation on FNs for which the remaining available CRBs are minimum.

The utility function definition of a FN f with respect to a given v_z, $U_f(z)$, has been based on the criterion of pursuing the allocation of v_z, to a suitable number of FNs in order to have the value of $U_f(z)$, defined as

$$U_f(z) = h_z\left(1 - \sum_{f\in\mathcal{F}} \frac{\tau_{f,z}}{n}\right), \tag{13}$$

as much as possible close to zero, i.e., all FNs have allocated the v_z. In this way, the aim is to favor the individual allocation of the VFs having the higher values of the occurrence frequency in the set $\mathcal{A}$, i.e., having the higher number of requests of performing individual services.

Hence, the FN f preferences list is sorted, according to (13), in an ascending order with respect to the values of $U_f(z)$ for all the VFs in $\mathcal{V}$, hence favoring allocation of the VF for which the resulting updated value of the utility function is minimized.

The proposed BMG algorithm is also based on a modified version of the GSA and consists of the following steps:

1) Each VF in $\mathcal{V}$ builds its preference list according to (11);
2) each FN in $\mathcal{F}$ builds its preference list according to (13);
3) each VF in $\mathcal{V}$ proposes the match to its most preferred FN;
4) each FN in $\mathcal{F}$ that receives one or more proposals accepts the preferred one among the alternatives and rejects the others;
5) each FN f that at the previous step has accepted a proposal, decrements its CRBs according to the resource request by the selected VF placement proposal;
6) repeat $1) - 5)$ until at least one FN has sufficient CRBs.

It is important to note that the matching strategy solves the *a priori* placement problem but, due to the physical network limitations, after application of a placement algorithm, all the VFs may not result loaded on at least one FN. For such reason, we have introduced the VFs replacement at run-time, allowing all the algorithm proposed and analyzed in the performance evaluation to load a VF during its execution if it is not present on any FN. As previously introduced, the VF replacement introduces a time penalty that we have taken into account in the performance analysis.

More in depth, due to the presence of the externalities, the two players sets involved in the MG, i.e., FNs and VFs, have to necessarily update their preference lists (PLs) in order to make decisions consistent with the current system conditions. Furthermore, aiming at maximizing the consistence between the decisions and the actual conditions, each FN simultaneously accept at most one VF, that is its preferred one among the proposals received. Hence, the final many-to-many matching is iteratively built during the algorithm. The algorithm details are reported on pseudocode in Algorithm 1.

Algorithm 1 Modified-GSA

```
1:  for each VF $z \in \mathcal{V}$ do
2:      Build the preference list according to (11);
3:  end for
4:  for each FN $f$ with sufficient CRBs do
5:      Build the preference list according to (13);
6:  end for
7:  end for
8:  for each VF $z \in \mathcal{V}$ do
9:      Send proposal to its preferred FN $f^\star$ its preference
    list;
10: end for
11: end for
12: for each FN $f$ that receives at least one proposal do
13:     accept the most preferred VF $z^\star$ among those received
    according to its preference list;
14:     load VF $z^\star$ on FN $f$;
15:     update $\tau_{f,z^\star} = 1$;
16: end for
17: end for
18: repeat 1) - 16) until there is at least one FN with sufficient
    CRBs to allocate one VF that it still does not contain
```

A. Practical Consideration

As previously introduced, it is straightforward to note that the proposed heuristic BMG limits the need of context information. In particular, BMG does not require the knowledge of the number of cycles required by the application services, the FNs computation capabilities, the time deadline associated to each application, the waiting and execution time at each FN. Moreover, we verify below that the BMG algorithm requires a lower computational complexity, compared to the CBMA and other considered alternatives.

In performing our computation complexity analysis, we focus on a worst case scenario in which all the applications require all the possible VF types. More in depth, each of the m applications has to sort the n FNs in accordance with its preferences. This procedure exhibits a complexity equals to

$$O(m \cdot n \log n). \tag{14}$$

Then, such procedure is repeated a number of times equals to the maximum chain length among the applications in $\mathcal{A}$. Supposing all the A_j with the same length and requiring all the VFs types, we can conclude that the CBMA exhibits a computational complexity given by

$$O(p \cdot m \cdot n \log n). \tag{15}$$

Likewise, in the BMG approach, each FN has to sort the VFs, hence the complexity results to be

$$O(n \cdot p \cdot \log p). \tag{16}$$

Considering that $p \approx n$ and $m >> p, n$, the complexity of the BMG approach is lower than the CBMA one.

V. Stability analysis

It is important to note that although in the classical matching game the stability condition is reached by applying the GSA, in the matching game with externalities to obtain a stable matching outcome is a very complex and challenging issue [20], [29], [32].
In order to discuss the stability convergence of the proposed algorithms, we refer in what follows to the *strictly-two-sided exchange-stability* (S2ES) stability definition derived from the one proposed in [33] .

Definition 1. Let $\mathcal{M}$ be a matching function, and let $\mathcal{M}(v_z)$ be the FN matched with the VF v_z in the matching $\mathcal{M}$, $\mathcal{M}$ is a S2ES matching *if* there not exists a pair of VFs (v_x, v_y) s.t.:

1) $U_{v_x}(\mathcal{M}(v_y)) \leq U_{v_x}(\mathcal{M}(v_x))$ and
2) $U_{v_y}(\mathcal{M}(v_x)) \leq U_{v_y}(\mathcal{M}(v_y))$ and
3) $U_{\mathcal{M}(v_x)}(v_y) \leq U_{\mathcal{M}(v_x)}(v_x)$ and
4) $U_{\mathcal{M}(v_y)}(v_x) \leq U_{\mathcal{M}(v_y)}(v_y)$ and
5) $\exists \psi \in \{v_x, v_y, \}$ s.t. at least one of the conditions $1) - 2)$ is strictly verified and
6) $\exists \phi \in \{\mathcal{M}(v_x), \mathcal{M}(v_y)\}$ s.t. at least one of the conditions $3) - 4)$ is strictly verified.

The above definition means that at least one of among the VFs and at least one among the FNs involved in the swap have to get benefit from the swapping improving its condition.
In what follows, we verify that both the CBMA and the BMG approaches converge to a $S2ES$ matching outcome.
We start the stability analysis by focusing on the CBMA. In particular, we suppose that there exists a pair of VFs (v_x, v_y), belonging to applications x and y respectively, s.t. the conditions $1) - 2)$ of Definition 1 are verified. Supposing VFs v_x and v_y s.t. $\mathcal{M}(v_x) = f$ and $\mathcal{M}(v_y) = g$ respectively, from conditions $1) - 4)$, we have:

$$T^x_{k_x,g,v_x} \leq T^x_{k_x,f,v_x}, \tag{17}$$

$$T^y_{k_y,f,v_y} \leq T^y_{k_y,g,v_y}. \tag{18}$$

As detailed in Section III, the completion time may or may not vary during time, so we cannot make any assumption about (17) and (18) and about the satisfaction of condition 5) of Definition 1. Instead, the FNs preferences lists are built taking into account the time deadlines of each application that proposes a given VF on such node. Let b_x and b_y be the time deadlines associated to applications x and y respectively. According to our assumptions that whether f prefers y to x, then $b_y \leq b_x$. In the same way we can show that whether g prefers x to y, we have $b_x \leq b_y$. Consequently, we necessarily have $b_x = b_y$ and none of f and g gains in switching. In conclusion the CBMA reaches a $S2ES$ matching outcome.
Likewise, we can conduct the stability analysis of the BMG approach. In order to discuss the stability, in this case we start considering the situation in which there exists a pair of VFs (v_x, v_y), with v_x allocated on FN f and v_y assigned to FN g, s.t. v_x prefers g and v_y prefers f to their actual partners. Hence, we have:

$$(r_{\mathcal{M}(v_y)} - r_{v_x}) \leq (r_{\mathcal{M}(v_x)} - r_{v_x}), \tag{19}$$

$$(r_{\mathcal{M}(v_x)} - r_{v_y}) \leq (r_{\mathcal{M}(v_y)} - r_{v_y}). \tag{20}$$

It is straightforward that (19) and (20) can be verified only if

$$r_{\mathcal{M}(v_x)} = r_{\mathcal{M}(v_y)}. \tag{21}$$

From (21) follows that v_x or v_y cannot improve their condition and that there not exists any ψ s.t. condition 5) of Definition 1 results true. Hence we can conclude that the reached matching is S2ES.

VI. Simulation results

In carrying out the performance evaluation for the proposed approaches, we have considered a cluster formed by 8 heterogeneous FNs, distributed with a Poisson Point Process in a circular area with a radius equals to 40m, each having a CPU frequency uniformly selected in the set $\{2.4, 3.6, 4.0\}$ GHz. Moreover, we have also assumed that:

- the FN computational capability uniformly selected, in terms of CRBs, in the set of values $\{100, 200, 300\}$;
- the number of different VFs, for each A_j service chain, uniformly distributed in the integer interval $[0, p]$, with $p = 10$, and the associated CRBs values uniformly selected in the range values $[35, 80]$;
- heavy tailed distribution of the occurrences of the VF types;
- 64 bits instructions for each requested service uniformly distributed in the range of integer values $[300, 500]$ with a mean cycles per instructions (CPI) equals to 8.
- the temporal penalty, associated to each FN, uniformly selected in $[0.5, 1.5]$ ms. Furthermore, each missed VF type in the FN network results in the addition, in the overall completion time, of the temporal penalty associated to the FN on which the VF is dynamically loaded.
- the temporal deadline, associated to each A_j, uniformly selected in $[0.2, 2.8]$ ms.

For the sake of simplicity, all the simulation parameters and their corresponding values have been reported in Table I, while Table II reports the system parameters details.

According to these assumptions, we have derived the system performance by resorting to computer simulations in terms of *wasted CRBs*, calculated over all the algorithm iterations, *mean overall completion time per application* (T_C), calculated as the ratio between the sum of the individual application completion times and the total number of applications, *worst overall completion time per application* (T_W) and *outage probability* P_O. In order to highlight the advantages of the proposed solutions, we compare their performance with those ones of some alternative approaches. In particular, we consider as benchmark the following methods:

- The *random placement method* (RPM) that, for each VF type, selects the FN to allocate it with uniform probability;
- The *greedy placement method* (GPM), based on the selection of the FN, for each VF, with the highest amount of CRBs.
- The *weighted greedy placement method* (GGPM) a heuristic that, iteratively, places the highly requested VFs on the FN with the highest CPU frequency.
- The near optimal potential game (POT) proposed in [21], adapted to our scenario, for which the utility function of each VF is equal to the execution time (1).

In particular, with reference to the POT scheme, due the prohibitive temporal complexity (exponentially growing with n), we have considered its sub-optimal version [21], according to which the players involved in the game can deviate from the Nash equilibrium up to ϵ, with $\epsilon = 0.01$. This modification results in a strong reduction of the time complexity, that results equal to $\mathcal{O}(\frac{n}{\epsilon})$, as detailed in [21]. All the proposed RPM, GPM and GGPM start by allocating one instance of each VF. Then, if there are still available CRBs, iteratively repeat the allocation until all the FNs are full.

Figure 2 provides comparisons in terms of *wasted CRBs* as a function of the number of different VFs to be allocated. This Figure points out the better behavior of the proposed methods with, in particular, the BMG scheme clearly outperforming all the other considered alternatives, hence achieving the best management of the computational resources. As direct consequence of the performance presented in Figure 2 we have that our method guarantees a greater number of VFs types loaded on the FNs, in comparison with the other alternatives. Then, in general terms, we can affirm that the greater the number of VFs types on the FNs, the higher is the odds of finding the required VF in the network, avoiding to occur in the temporal penalties. Such insight has been confirmed by Figures 3-4 that show performance comparisons in terms of the mean overall completion time and worst overall completion time per application, respectively, as a function of the number of applications to be performed. In particular, in terms of mean overall completion time, in Figure 3, the CBMA and BMG achieve better results considering the other alternatives, with the CBMA overcoming the BMG. The same behavior is evident in Figure 4, where the worst T_W is given as a function of the number of applications to be considered. This is due to the intrinsic trend of both the MT based approaches to deploy on the FNs the most requested VFs. In fact, in both approaches the presence of more than one instances of the highly requested VFs in the network increments the chances to properly select the FN that brings more advantages in terms of overall completion time. As evident in Figure 5, increasing the VF types, the system performance get worse. However, also in this case, the proposed approaches reduce the worst overall completion time, by ensuring higher responsiveness despite in presence of heavy applications demand. By guaranteeing better performance in Figure 3 and Figure 4, the proposed approaches provide a better compliance with the applications deadlines, that implies lower values of outage probability, hence, of the number of applications which do not match their temporal deadlines as depicted in Figure 6. From Figures 3-6, it is evident that all the considered methods are overcome by the CBMA. However, by taking into account the resulting temporal complexity presented in Sec. IV, we can say that the BMG alternative represents the best tradeoff between high performance and low temporal complexity. Furthermore, in terms of convergence time, the BMG and CBMA complete their computation in 0.023926 and 0.051621 s respectively, as

TABLE I
SIMULATION PARAMETERS

Parameter	value
Network radius	40 m
CPU frequency	{2.4, 3.6, 4.0} GHz
FN CRBs	{100, 200, 300}
Number of VFs types	10
VF CRBs	[35,80]
Bits per instruction	64
Number of instructions	[300, 500]
CPI	8
Time penalty	[0.5,1.5] ms
Time deadline	[0.2, 2.8] ms

TABLE II
SYSTEM PARAMETER

Operating System	Ubuntu 16.04
Programming Platform	Matlab
CPU	Intel i7-3770 3.40Ghz
RAM	16 GB
Convergence BMG	0.023926 s
Convergence CBMA	0.051621 s

reported in Table II. Finally, we point out that the convergence times have been obtained by considering a scenario with 140 applications, 8 FNs and a number of VF types equals to 10.

VII. CONCLUSIONS

In this paper the VFs placement problem in a FC scenario is addressed. The proposed VFs placement schemes, namely CBMA and BMG, apply the MT framework aiming at minimizing both the worst overall application time and the

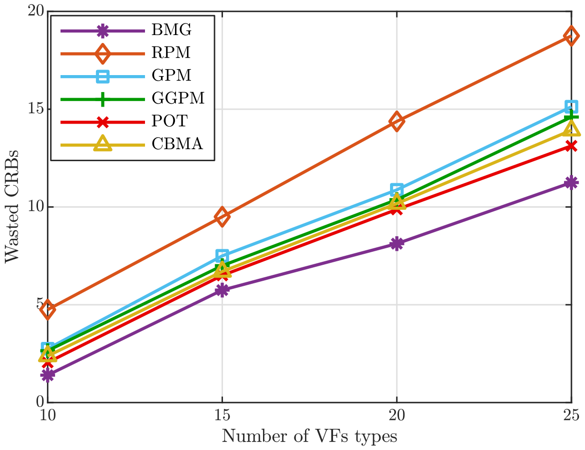


Fig. 2. Wasted CRBs, supposing $n = 8$ FNs, and $m = 100$ applications requests.

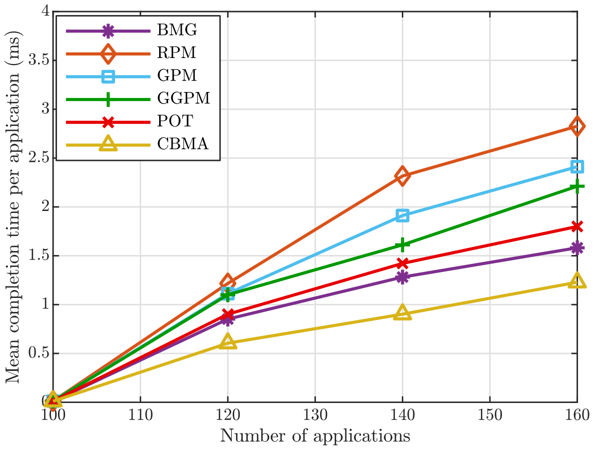


Fig. 3. Average application completion time, for $n = 8$ FNs, and $p = 10$ VFs types.

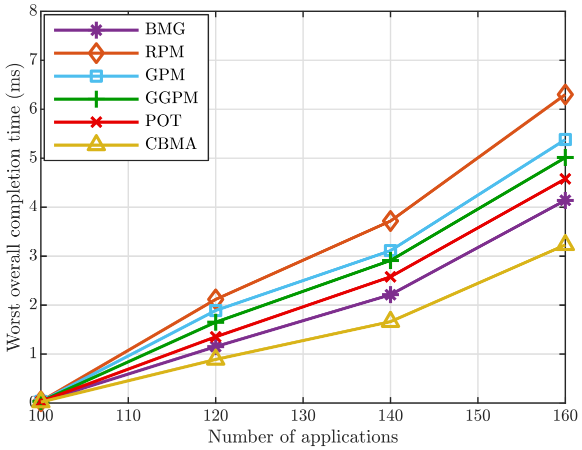


Fig. 4. Worst application completion time, for $n = 8$ FNs, and $p = 10$ VFs types.

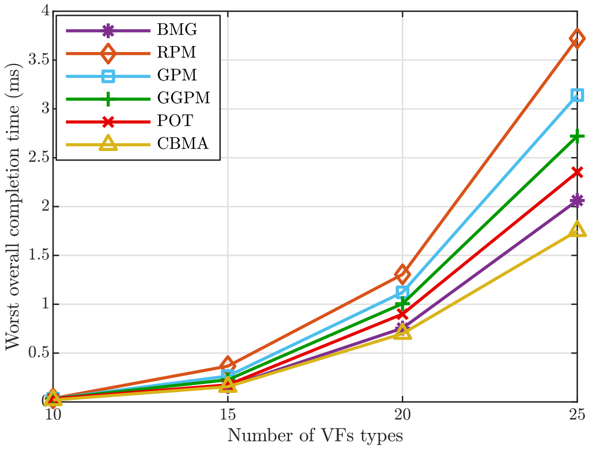


Fig. 5. Worst application completion time, for $n = 8$ FNs, by varying the maximum size of the service chain, i.e., the number of VFs types.

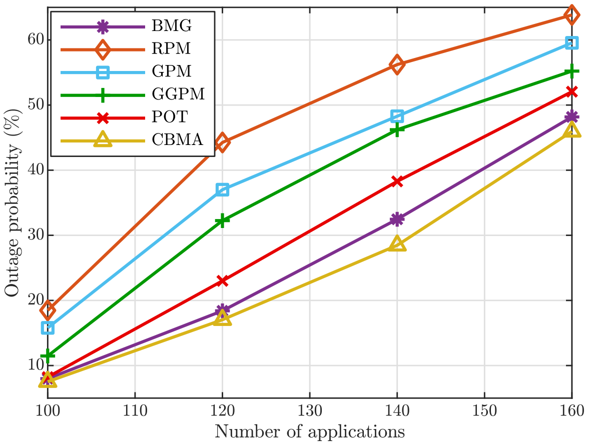


Fig. 6. Outage probability comparison for $n = 8$ FNs, and $p = 10$ VFs types.

applications outage probability. Performance comparisons with different services placement methods, i.e., a random scheme, two greedy disciplines and a Game Theory based approach, have been presented in order to highlight the advantages of the proposed schemes. In particular, we have validated that the first proposed CBMA approach achieves the best performance, while at the expense of an higher computation complexity and a more heavy signaling overhead with respect to the second BMG solution.

Finally, for both the CBMA and BMG matching approaches we have discussed and theoretically proved the matching stability according to the strictly-two-sided exchange-stability definition.

Acknowledgment

This work has been supported by the project ”GAUChO - A Green Adaptive Fog Computing and Networking Architecture” funded by the MIUR Progetti di Ricerca di Rilevante Interesse Nazionale (PRIN) Bando 2015 - grant 2015YPXH4W_004.

References

[1] A. Dastjerdi, H. Gupta, R. Calheiros, S. Ghosh, and R. Buyya, “Chapter 4 - fog computing: principles, architectures, andapplications,” in *Internet of Things*, R. Buyya and A. V. Dastjerdi, Eds. Morgan Kaufmann, 2016, pp. 61 – 75.

[2] F. Bonomi, R. Milito, J. Zhu, and 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*, ser. MCC ’12. New York, NY, USA: ACM, 2012, pp. 13–16. [Online]. Available: http://doi.acm.org/10.1145/2342509.2342513

[3] C. Alippi, R. Fantacci, D. Marabissi, and M. Roveri, “A cloud to the ground: The new frontier of intelligent and autonomous networks of things,” *IEEE Communications Magazine*, vol. 54, no. 12, pp. 14–20, December 2016.

[4] A. Brogi and S. Forti, “Qos-aware deployment of iot applications through the fog,” *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1185–1192, Oct 2017.

[5] O. Skarlat, M. Nardelli, S. Schulte, and S. Dustdar, “Towards qos-aware fog service placement,” in *2017 IEEE 1st International Conference on Fog and Edge Computing (ICFEC)*, May 2017, pp. 89–96.

[6] R. Morabito, I. Farris, A. Iera, and T. Taleb, “Evaluating performance of containerized iot services for clustered devices at the network edge,” *IEEE Internet of Things Journal*, vol. 4, no. 4, pp. 1019–1030, Aug 2017.

[7] “Microsoft azure,” *Azure IoT Edge Documentation*, Jul. 2018.

[8] “Amazon web services,” *AWS Greengrass Documentation*, May 2019. [Online]. Available: https://docs.aws.amazon.com/greengrass/index.html

[9] A. Das, S. Patterson, and M. Wittie, “Edgebench: Benchmarking edge computing platforms,” in *2018 IEEE/ACM International Conference on Utility and Cloud Computing Companion (UCC Companion)*, Dec 2018, pp. 175–180.

[10] Z. Wen, R. Yang, P. Garraghan, T. Lin, J. Xu, and M. Rovatsos, “Fog orchestration for internet of things services,” *IEEE Internet Computing*, vol. 21, no. 2, pp. 16–24, Mar 2017.

[11] M. Masdari, S. S. Nabavi, and V. Ahmadi, “An overview of virtual machine placement schemes in cloud computing,” *Journal of Network and Computer Applications*, vol. 66, pp. 106 – 127, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1084804516000291

[12] J. G. Herrera and J. F. Botero, “Resource allocation in nfv: A comprehensive survey,” *IEEE Transactions on Network and Service Management*, vol. 13, no. 3, pp. 518–532, 2016.

[13] O. Skarlat, M. Nardelli, S. Schulte, M. Borkowski, and P. Leitner, “Optimized iot service placement in the fog,” *Serv. Oriented Comput. Appl.*, vol. 11, no. 4, pp. 427–443, Dec. 2017. [Online]. Available: https://doi.org/10.1007/s11761-017-0219-8

[14] M. Chiang and T. Zhang, “Fog and iot: An overview of research opportunities,” *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 854–864, Dec 2016.

[15] R. Moreno-Vozmediano, R. S. Montero, E. Huedo, and I. M. Llorente, “Cross-site virtual network in cloud and fog computing,” *IEEE Cloud Computing*, vol. 4, no. 2, pp. 46–53, March 2017.

[16] E. Saurez, K. Hong, D. Lillethun, U. Ramachandran, and B. Ottenwälder, “Incremental deployment and migration of geo-distributed situation awareness applications in the fog,” in *Proceedings of the 10th ACM International Conference on Distributed and Event-based Systems*, ser. DEBS ’16. New York, NY, USA: ACM, 2016, pp. 258–269. [Online]. Available: http://doi.acm.org/10.1145/2933267.2933317

[17] M. Taneja and A. Davy, “Resource aware placement of iot application modules in fog-cloud computing paradigm,” in *Integrated Network and Service Management (IM), 2017 IFIP/IEEE Symposium on*. IEEE, 2017, pp. 1222–1228.

[18] H. Zhang, Y. Xiao, S. Bu, D. Niyato, F. R. Yu, and Z. Han, “Computing resource allocation in three-tier iot fog networks: A joint optimization approach combining stackelberg game and matching,” *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1204–1215, Oct 2017.

[19] L. F. Bittencourt, J. Diaz-Montes, R. Buyya, O. F. Rana, and M. Parashar, “Mobility-aware application scheduling in fog computing,” *IEEE Cloud Computing*, vol. 4, no. 2, pp. 26–35, 2017.

[20] S. Bayat, Y. Li, L. Song, and Z. Han, “Matching theory: Applications in wireless communications,” *IEEE Signal Processing Magazine*, vol. 33, no. 6, pp. 103–122, Nov 2016.

[21] H. Shah-Mansouri and V. W. S. Wong, “Hierarchical fog-cloud computing for iot systems: A computation offloading game,” *CoRR*, vol. abs/1710.06089, 2017.

[22] Z. A. Mann, “Allocation of virtual machines in cloud data centers&mdash;a survey of problem models and optimization algorithms,” *ACM Comput. Surv.*, vol. 48, no. 1, pp. 11:1–11:34, Aug. 2015. [Online]. Available: http://doi.acm.org/10.1145/2797211

[23] S. Wang, A. Zhou, F. Yang, and R. N. Chang, “Towards network-aware service composition in the cloud,” *IEEE Transactions on Cloud Computing*, pp. 1–1, 2017.

[24] H. Xu and B. Li, “Anchor: A versatile and efficient framework for resource management in the cloud,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 6, pp. 1066–1076, 2013.

[25] L. Francesco, G. Alessandro, P. Antonio, S. Vincenzo, D. G. Alessandro, T. Marco, D. David, P. Panagiotis, and D. P. Francesco, “Stochastic and exact methods for service mapping in virtualized network infrastructures,” *International Journal of Network Management*, vol. 27, no. 6, p. e1985, 2017, e1985 nem.1985. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/nem.1985

[26] C. Pham, N. H. Tran, S. Ren, W. Saad, and C. S. Hong, “Traffic-aware and energy-efficient vnf placement for service chaining: Joint sampling and matching approach,” *IEEE Transactions on Services Computing*, 2017.

[27] R. Mahmud, K. Ramamohanarao, and R. Buyya, “Latency-aware application module management for fog computing environments,” *ACM Trans. Internet Technol.*, vol. 19, no. 1, pp. 9:1–9:21, Nov. 2018. [Online]. Available: http://doi.acm.org/10.1145/3186592

[28] M. Agiwal, A. Roy, and N. Saxena, "Next generation 5g wireless networks: A comprehensive survey," *IEEE Communications Surveys Tutorials*, vol. 18, no. 3, pp. 1617–1655, thirdquarter 2016.

[29] Z. Han, D. Niyato, W. Saad, T. Basar, and A. Hjørungnes, *Matching Theory for Wireless Networks*. UK: Springer International Publishing, 2017.

[30] D. Manlove, *Algorithmics Of Matching Under Preferences*, ser. Theoretical computer science. World Scientific Publishing, 2013. [Online]. Available: http://eprints.gla.ac.uk/79820/

[31] D. Gale and L. S. Shapley, "College admissions and the stability of marriage," *The American Mathematical Monthly*, vol. 69, no. 1, pp. 9–15, 1962. [Online]. Available: http://www.jstor.org/stable/2312726

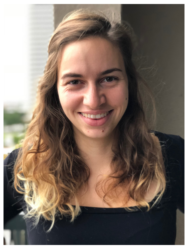
[32] A. E. Roth, "Deferred acceptance algorithms: history, theory, practice, and open questions," *International Journal of Game Theory*, vol. 36, no. 3, pp. 537–569, Mar 2008. [Online]. Available: https://doi.org/10.1007/s00182-008-0117-6

[33] E. Bodine-Baron, C. Lee, B. Chong, Anthonyand Hassibi, and A. Wierman, *Peer Effects and Stability in Matching Markets*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 117–129.

Francesco Chiti (M01, S13) received the PhD degrees in Informatics and Telecommunications Engineering from the University of Florence in 2004, where he is presently working as Associate Professor. His current research topics are in the Internet of Things domain with a special focus on Ad Hoc and Sensor Networks. He took part in several European research projects as REGPOT AgroSense, IP GoodFood, STREP DustBot, NoEs NEWCOM and CRUISE, GJU TWIST, ETSI STF179 and COST 289 action. He is presently Vice Chair of IEEE Communications and Information Security Technical Committee and serves as Associate Editor of Wiley International Journal on Security and Communication Networks, Wiley Security and Privacy Journal, and Springer Peer-to-Peer Networking and Applications Journal. He is author of 65 book chapters, more than 40 journal papers and 60 conference papers.

Romano Fantacci (F'05) is a Full Professor of Computer Networks at the University of Florence, Florence, Italy, where he heads the Wireless Networks Research Laboratory. He is the founding director of the Information Communication Technology Inc. (TiCOM), with Leonardo spa and of the Wireless Communications Research Centre (LiRS) with Telecom Italia. He received the M.S. degree in Electrical Engineering from the University of Florence, Italy and the Ph.D. degree in Computer Networks from the University of Florence, Italy. His current research interests encompass several fields of wireless engineering and computer communication networking including, in particular, performance evaluation and optimization of wireless networks, emerging generations of wireless standards, cognitive wireless communications and networks, satellite communications and systems. Dr. Fantacci was elected Fellow of the IEEE in 2005 for contributions to wireless communication networks. He received several awards for his research, including the IEE Benefactor Premium, the 2002 IEEE Distinguished Contributions to Satellite Communications Award, the 2015 IEEE WTC Recognition Award, the IEEE sister society AEIT Young Research Award and the IARIA Best Paper Award, the IEEE IWCMC'16 Best Paper Award and the IEEE Globecom'16 Best Paper Award. He served as Area Editor for IEEE Trans. Wireless Commun., Associate Editor for IEEE Trans. on Commun., IEEE Trans. Wireless Commun. Regional Editor for IET Communications and Associate Editor for several non-IEEE Technical Journals. He guest edited special issues for IEEE Journals and Magazines and served as symposium chair of several IEEE conferences, including VTC, WCNC, PIRMC, ICC and Globecom. Dr. Fantacci currently serves on the Board of Governors of the IEEE sister society AEIT, as Area Editor for IEEE IoT Journal and as member of the Steering Committee of IEEE Wireless Comm. Letters.

Federica Paganelli Federica Paganelli (M'07) received the Ph.D. degree in telematics and information society from the University of Florence, Italy, in 2004. She is currently Assistant Professor at the Department of Computer Science, University of Pisa. Previously, she was with the National Interuniversity Consortium for Telecommunications (CNIT). She has participated to several national and European research projects. Her recent research interests include: resource management in software-defined network infrastructures, network virtualization, protocols and services for the Internet of Things. She serves as an Associated Editor of Future Internet (MDPI). She participated to the standardization activities of the IEEE NGSON Working Group.



Benedetta Picano (S'17) received the B.S. degree in Computer Science, as the M.Sc. degree in Computer Engineering, from the University of Florence, where she is a Ph.D. student in the Department of Information Engineering. She was a visiting researcher at the University of Houston. Her research fields include matching theory, nonlinear time series analysis, and resource allocation in edge and fog computing infrastructures.