

An overview of service placement problem in Fog and Edge Computing

Farah Ait Salaht, Frédéric Desprez, Adrien Lebre

▶ **To cite this version:**

Farah Ait Salaht, Frédéric Desprez, Adrien Lebre. An overview of service placement problem in Fog and Edge Computing. ACM Computing Surveys, 2020, 53 (3), Article 65, 35 p. 10.1145/3391196 . hal-02596419

HAL Id: hal-02596419

https://inria.hal.science/hal-02596419

Submitted on 20 May 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

An overview of service placement problem in Fog and Edge Computing

Farah AÏT-SALAHT[1] Frédéric DESPREZ[1] Adrien LEBRE[2]

[1]Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LIG, France
[2]STACK Research Group - IMT-Atlantique, Inria, LS2N, Nantes, France
email: firstname.lastname@inria.fr

Abstract

To support the large and various applications generated by the Internet of Things (IoT), Fog Computing was introduced to complement the Cloud Computing and offer Cloud-like services at the edge of the network with low latency and real-time responses. Large-scale, geographical distribution and heterogeneity of edge computational nodes make service placement in such infrastructure a challenging issue. Diversity of user expectations and IoT devices characteristics also complexify the deployment problem. This paper presents a survey of current research conducted on Service Placement Problem (SPP) in the Fog/Edge Computing. Based on a new classification scheme, a categorization of current proposals is given and identified issues and challenges are discussed.

Keywords— Fog Computing, Edge Computing, Services placement, Placement taxonomy

1 Introduction

In recent years, the Internet of Things (IoT) becomes ingrained in our society by transforming objects of everyday life like wearable, transportation, augmented reality, etc., in communicating devices, and thus introduces new challenges and opportunities. With more than 50 billion devices connected to the network by 2020 according to Cisco [39], it is clear that the current infrastructures will not be able to support all the data that will be generated. Indeed, the current Cloud infrastructure alone can not support a large number of the current IoT applications and those essentially for three main reasons: First, the huge amount of generated data makes their transfer from where they are created (end-devices), to where they are processed (Cloud servers), impractical due to bandwidth limitations, processing overhead, and transmission costs. Second, the significant end-to-end delay from end-devices to the Cloud servers that are often too far from end-users can deter the performance of applications that require real-time analysis, such as online gaming, video applications, etc. Finally, some data may have implications in terms of privacy and security and it is advisable or even forbidden for this data to cross the entire Internet [159].

To cope with these issues, a promising paradigm able to avoids network bottlenecks, overcome communication overheads and reduce the delay of data transfer has been identified [24]. This new conceptual approach that combines the benefits of Cloud and the decentralized processing of services on edge devices is known as Fog or Edge Computing [24]. The community has not yet converged against crisp definitions of these terms [21, 38, 51, 74]. In the following, we use the term Fog Computing for simplicity.

Fog Computing [41] extends Cloud Computing and services to the edge of the network, bringing processing, analysis, and storage closer to where requests are created and used. The objective is to reduce the amount of data sent to the Cloud, reduce latency and computation costs. As a new paradigm, Fog Computing poses old and new challenges, and one of the main open issues is service management and

more precisely the service placement problem. Indeed, one of the major barriers to the adoption of Fog is "how to efficiently deploy services on available Fog nodes". Unlike Cloud data centers, Fog devices are geographically distributed, resource-constrained, and highly dynamic, which makes the problem quite challenging. Therefore, the definition of an efficient, effective, and fair provisioning for the IoT applications will be important to provide and hence ensure end-to-end guaranteed services to end-users. Moreover, depending on the context and the interest, different aspects may come into focus: resource utilization [140], Quality of Service (QoS) [29, 134, 137], Quality of Experience (QoE) [96], etc. These last few years, several works have been carried out and attempt to address this issue. Different assumptions, characteristics, and strategies have been considered to propose an efficient service placement. In this paper, we review a wide range of works that studied this issue in Fog environments and explore the methodologies and the strategies proposed in the literature. We underline that the survey goes beyond just describing the main approaches developed in the literature. It first identifies five main scenarios, based on user expectations, problem descriptions, and deployment objectives. Second, it provides a new *classification* scheme where the different variants of SPP, and the various solutions coming from the research community are classified. The following aspects are considered: *problem statement*, placement *characteristics*, technical *formulation*, problem *objectives*, *optimization strategies*, and *experimental tools*.

The paper is organized as follows. Section 2 summarizes existing surveys and tutorials on Fog Computing and resource management in the related area and highlights the contributions of our paper. Section 3 gives an overview of Fog systems: definition, architecture, main characteristics and advantages. Section 4 introduces the service placement problem and summarizes the most common formulations, optimization strategies, major design objectives, and evaluation tools proposed in the literature to address this issue. The provided classification for the SPP approaches is presented in Section 5. Section 6 outlines the open challenges and highlights emerging research directions. Finally, Section 7 concludes this survey.

2 Existing surveys

There are several surveys that address different aspects of Fog Computing and the related challenges [25, 30, 38, 41, 74, 82, 91, 95, 106, 117, 120, 123, 142, 158, 160]. Indeed different works have been proposed to discuss the concept and the role of Fog computing [25, 38, 40, 117, 138, 146, 155, 158]. For instance, Bonomi et al. [25] investigate the role of Fog Computing in the IoTs domain, its characteristics, and its applications. Saharan and Vaquero et al. [146] give an overview of the concept in terms of enabling technologies and emerging trends. Chiang and Zhang [38] discuss, in the context of IoT, the need for a new architecture for computing, and storage. In [158], the authors give the definition of Fog Computing and closely related concepts and present three motivating applications: augmented reality, Content delivery, and Mobile Data Analytics. Many other papers discuss the Fog Computing characteristics, application domains, and related research challenges as [40, 138, 155], however they do not investigate and discuss the problem of service placement is such geo-distributed and large scale environments. Indeed, these studies do not provide insights into how IoT applications are deployed over the network (what are the applications requirements, infrastructures characteristics, domain constraints; mapping strategies; metrics to optimize, etc.).

Recent surveys attempt to fill the resource management and service placement problems, among them we quote the work of Yousefpour et al. [160], that provides a tutorial on Fog Computing, compares the Fog to the related computing paradigms, and discusses the resource management and service deployment (orchestration and migration) in Fog environment. In [95], the authors present a taxonomy of Fog Computing, its related challenges, and features, and discuss briefly the problem of service management. Nath et al. [109] focus on the architectures and features of Fog Computing systems, applications of Fog, security and privacy of Fog, and discuss the future scopes and open research. In their survey, the authors briefly discuss challenges related to resource management, orchestration between fog nodes and the cloud, and give some comparison of different QoS aspects of Fog computing. In [105], Mouradian et al. present a detailed review on Fog architectures and algorithms and illustrate two application domains namely IoT and Content Delivery Networks (CDN). Li et al. [91, 142] provide a survey on Edge Computing architecture and system management. They propose to characterize Edge Computing by considering the

following aspects: architecture characteristics, management approaches, and design objectives. Brogi et al. [30] propose also to review the existing SPP proposals. They pursue three main objectives: elaborate an overview of the current algorithms, available prototypes, and experiments; classify the works based on the application and Fog infrastructure characteristics; And, identify and discuss some open challenges.

Although these surveys explore the resource management and service placement problem in Fog/Edge Computing, we note that these works are limited in at least one of the following: 1) limited review, and discussion on SPP in Fog/Edge Computing; 2) lack of comprehensive descriptions of the problem; state-of-the-art efforts regarding problem taxonomy, resolution approaches, evaluation environments; concrete research directions; 3) do not provide an in-depth comparison, classification or some useful insights regarding the existing works (e.g., how we can evaluate/compare the different proposals).

Our paper is different in the content and research issues. Mainly dedicated to achieving an exhaustive and very clear overview of SPP in the Fog environment, our survey aims at simplifying the user's access to references and identify a flavor of challenges. It is characterized by the following contributions.

2.1 Our contributions

The main contributions of our paper can be summarized as follows:

1. Provide an exhaustive and very clear description and overview of the SPP in the Fog environment.
2. Provide a taxonomy of the problem in the area of large-scale, geo-distributed and heterogeneous systems.
3. Propose a classification of surveyed works based on the identified scenarios, provided taxonomy and optimization strategies.
4. Finally, highlight the open challenges and discusses future research directions.

3 Overview of Fog Computing

In this section, we provide a brief overview of Fog Computing: definition, architecture, main characteristics and advantages.

3.1 Definition

Fog Computing (FC) is a highly virtualized platform, that offers computational resources, storage and control between end-users and Cloud servers. Introduced by Cisco in 2012 [25], FC is a new paradigm in which centralized Cloud coexists with distributed edge nodes and where the local and global analysis are performed at the edge devices or forwarded to the Cloud.

3.2 Architecture

Several architectures have been provided for FC. Mostly derived from the fundamental three-layers structure (as depicted in Fig. 1), a Fog infrastructure consists of IoT devices (End layer), one or more layers of Fog Nodes, and at least one Cloud Data Center (Cloud layer).

- *End layer:* Bottom-most layer and closest to the end-users. It comprises various IoT devices (e.g., cameras, mobile phones, smart cars, smoke detectors, ...). Widely geographically distributed, these devices enable sensing events and forwarding them to their immediate upper layer in the hierarchy for analysis and storage.
- *Fog layer:* Middle layer, consists of a set of devices that are able to process, and store the received requests. Denoted by *Fog Nodes* (FNs), these devices that include access points, routers, gateways, switches, base stations, laptops, specific Fog servers, etc., are connected to the Cloud servers and are able to send requests to data centers. Distributed between the end-users and DCs, these resources can be fixed devices (static) at some location or mobile (like smartphones, vehicles, intelligent transportation systems, Drones ...).

- *Cloud layer:* Upper-most layer in this architecture. It is composed of several servers and Data Centers (DCs) able to performing complex analyses and storing a large amount of data.

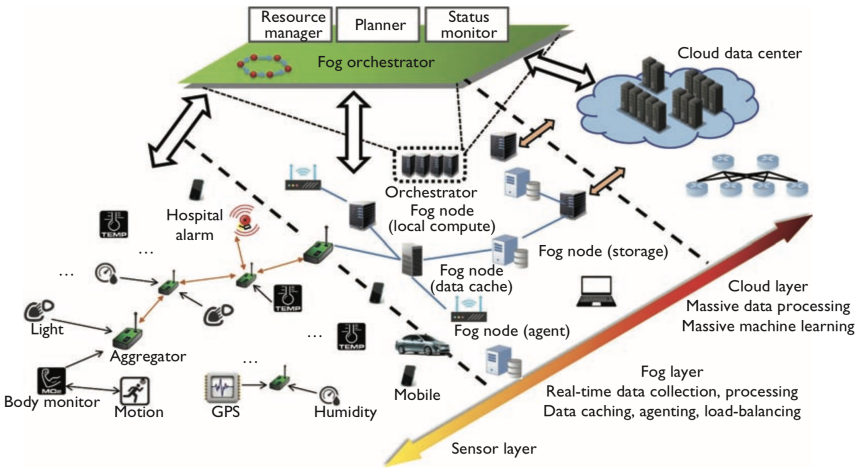


Figure 1: Generic Fog computing architecture [152].

3.3 Architectural characteristics and advantages

Considered as the future of Cloud systems and the Internet of Things, the Fog Computing involves a number of characteristics and advantages where the main ones are mentioned below:

1. *Location awareness and low latency.* Most latency-sensitive applications such as augmented reality, gaming, or video streaming, require sub-second processing time and do not necessarily need to be sent across long routes to data centers or Cloud services to be processed. With the Fog Computing, the support of these aspects is provided through the geo-distribution of the various Fog nodes in different locations and their proximity to end-users. Sarkar and Misra [124] proved by theoretical modeling that the service latency in FC is significantly lower than that with Cloud Computing.

2. *Save bandwidth.* FC helps to unclog the network and speed up the processing of certain tasks by performing locally some computation tasks and sending only part of useful data or those that require significant analysis to the Cloud.

3. *Scalability.* The number of connected devices grows rapidly, and the IoT data and application generated by these trillions of things [92] increases also exponentially. Given this large amount of data, processing the whole IoT applications in the Cloud is neither efficient nor feasible, so Fog intervenes as a complementary paradigm able to support all these requests and help the scalability of such systems.

4. *Support for mobility.* Having a widely distributed fog devices that providing computational and storage capabilities over the network, the Fog Computing is more suited to support the mobility of end-users than the traditional centralized Cloud servers and thus will allow to provide service guarantees for the mobile end-users without interruptions.

4 Service Placement in Fog Computing

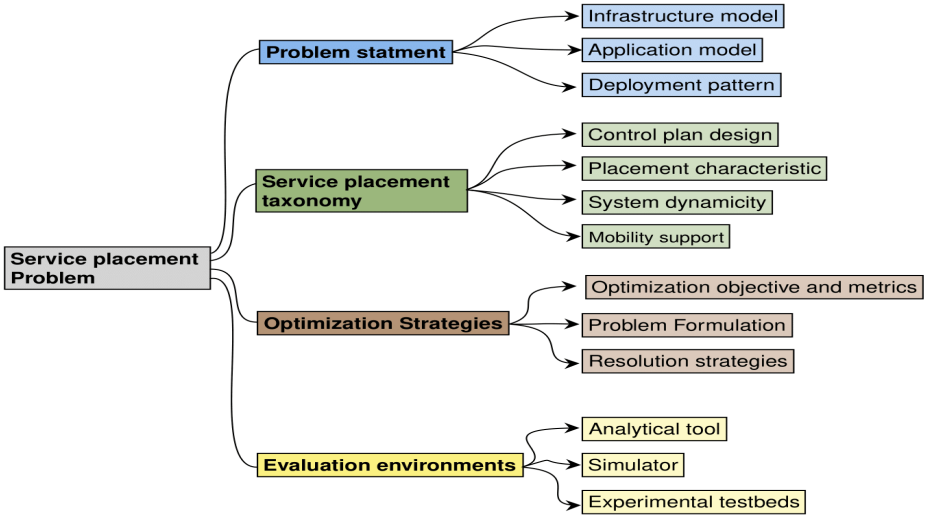


Figure 2: The structure of the section.

The service placement problem has been highly discussed in the literature and several proposals have emerged. Based on different application descriptions, network assumptions, and expected outcomes, these solutions are generally difficult to compare with each other. In this section, we propose to describe the methodology usually employed to address the SPP and give an overview of the following aspects: problem statement, placement taxonomy, optimization strategies, and evaluation tools (as depicted in Figure 2). These elements will allow us to clearly describe the problem, and define the different aspects on which we will base our comparison and classification of the surveyed works.

4.1 Problem statement

The statement of the SPP problem goes through the description of the following three parts: the infrastructure model, the application model, and the deployment pattern with its related constraints.

4.1.1 Infrastructure model

The physical Fog infrastructure (see Fig. 1) consists respectively of a set of devices with no computational capabilities (sensors, and actuators), and a set of resources that possess computational power and/or storage capacity (Fog nodes, and Cloud data centers). Due to their physical structure [97], the fog nodes are resource-constrained and heterogeneous to each other. And, any devices equipped with computational resources (in terms of CPU, memory, storage, bandwidth, ...) can be considered as potential FNs, such as routers, small servers, access points, laptops, or gateway, etc.

The infrastructure network is generally abstracted as a connected graph where the vertices denote the set of IoT devices, Fog nodes, and Cloud servers, and the edges denote the links between the nodes. We mention hereafter the most common resources type and characteristics depicted in the literature to describe the Fog infrastructure.

- Resources type. *Computing:* servers, PCs, cloudlets [8, 125], etc. *Networking:* gateways, routers, switches, base stations, etc. *Storage:* every node that can provide storage. *Mobile:* vehicles,

smartphones, etc. *Endpoint abstraction:* sensors, actuators (*e.g.,* GPS devices, wireless sensors, cameras, voice collector, radar, ...).

- Characteristics. *Computing:* CPU power, number of cores, RAM, battery life, etc. *Networking:* Type: wireless, wired; Capabilities: latency, bandwidth, error rate, etc. *Storage:* Disk, etc. *Virtualization:* VMs, containers, unikernel, etc. *Hardware:* GPU, NUMA, FPGA, etc.

4.1.2 Application(s) model

Several abstractions and model definitions are used in the literature to characterize the applications generated by the IoT devices and treated at Fog resources and Cloud servers. According to the surveyed papers, we identify the following main descriptions: *a)* a monolithic service, *b)* a set of inter-dependent components, and *c)* a connected graph.

a) Monolithic service The application sent by end-users or IoT devices is represented in the form of a single component (monolithic service). As an example, we can mention the case of an image processing application or data instance that needs to be proceeded or stored in a single physical node. The application can be defined in this case as a monolithic service.

b) Set of inter-dependent services This case assumes that the application is pre-partitioned into a set of components (services), each performs some specific operation (functionality) in the application. In that case, dependencies between the application components are not considered.

c) A connected graph The application, in this case, is composed of a set of inter-dependent components represented as a connected graph. The vertices represent the processing/computational components of the application, and edges represent the inter-dependencies and communication demand between nodes [64].

Different topologies of a graph can be identified and among them, we have respectively: line graph, tree application graph, and Directed Acyclic Graph (DAG). The DAG application topology is the most often used because it models a large range of realistic IoT applications like Video processing [14, 22, 127], gaming [165], or healthcare [59] applications. Figure 3.a) illustrates an example of DAG application (cognitive assistance application).

Regarding the application requirements, we can summarize some of them in the following: *Computing:* CPU power, number of cores, RAM, etc. *Network-oriented:* Bandwidth, Latency, Error-rate, Jitter (per link, end-to-end). *Task-oriented:* Deadline. *Location-oriented:* the application must run in a specific geographical location (for instance in Paris); the application can run only at some Fog node, etc.

4.1.3 Deployment pattern

The application placement problem defines a mapping pattern by which applications components and links are mapped onto an infrastructure graph (i.e., computing devices, and physical edges). Figure 3 shows a mapping example of an application modeled as a DAG (Fig. 3.a) to available Fog nodes (Fig. 3.b).

Typically, application placement involves finding the available resources in the network (nodes and links) that satisfy the application(s) requirements, satisfy the constraints, and optimize the objective (if any). For instance, respect the applications (services) requirements, not exceed the resource capacities, satisfy the locality constraints, minimize the energy consumed, etc. Service providers have to take into account these constraints to first, limit the research space and second, provide an optimum or near optimum placement. We propose hereafter to depict some of the constraints mostly considered in the literature.

- ⋆ Resource constraints (C_R). An infrastructure node is limited by finite capabilities in term of CPU, RAM, storage, bandwidth, etc. Therefore, when placing application(s) (service components), we

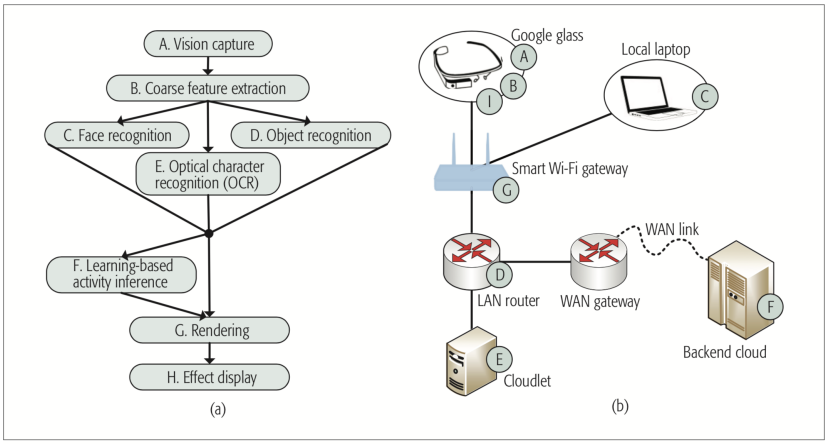


Figure 3: Cognitive assistance application [67], shown in (a), and deployed onto Fog network, shown in (b).

need to respect the resource requirements, i.e., ensures that the resources of the components deployed on the infrastructure nodes do not exceed their capabilities.

- ⋆ Network constraints (C_N). A network link can also be bounded by constraints like latency, bandwidth, etc., and these constraints need to be satisfied when deploying applications.
- ⋆ Application constraints: We highlight here two kind of application constraints:
  - Locality requirement (C_L). Locality requirement typically restricts certain services' executions to specific locations. Due to specific hardware, privacy requirements or given policy, some components can be restricted to be deployed on specific areas (zone) or devices. Locality constraints can be based on: a set of Fog nodes [154, 164]; a specific geo-spatial location using GPS for instance [124], impose a co-localization of components [3], etc.
  - Delay sensitivity (C_D). Some applications can specify a deadline for processing operation, or deploying the whole application in the network. This constraint is generally specified by defining a threshold to not exceed.

4.2 Service placement taxonomy

Addressing the SPP involves considering some specificities and criteria when designing the deployment strategies. Denoted as a service placement taxonomy, we propose in this paper to pay attention to the following four main aspects as depicted in Figure 4.

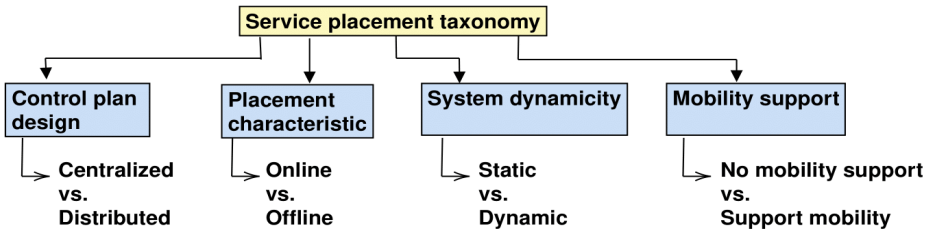


Figure 4: Service placement taxonomy.

As such, the first specificity considers whether the mapping coordination is done in a centralized or distributed manner. The second is based on whether the problem is tackled as an offline or online

deployment. The third proposes to observe whether the dynamicity of the system is handled or not (i.e., handle the changes in the system, or not). Finally, the fourth characteristic describes whether the mobility of end-users and/or Fog devices is supported by the provided solution or not.

These eight characteristics described hereafter are used in Section 5.1 to classify the SPP proposals coming from the literature.

4.2.1 Control plan design: Centralized vs. Distributed

The development of a placement strategy and service management starts first by selecting the coordination strategy to adopt. Two common control plane models are presented in this paper: *centralized* and *Distributed* coordination. Relevant for multi-layered and geo-distributed systems, these approaches are relatively different and each has its own advantages and inconveniences as presented in the following.

a) Centralized. A centralized control plane requires global information about application demands and infrastructure resources to take and disseminate global deployment decisions. The advantage of centralized placement algorithms is to potentially find a globally optimal solution, however, they are vulnerable regarding the scalability and the computational complexity issue.

When surveying papers, we observed that a large number of works considers a centralized control plane when addressing the SPP. Among these works, we mention the work of Hong et al. [71] that considers a central coordinator to make deployment decisions of IoT applications over Fog infrastructure.

b) Distributed

Unlike centralized solution, a distributed approach considers multiple authority and orchestrator nodes to control the services mapping. Generally distributed in the network (as illustrated in Fig.5), the management elements compute placement decisions based on local resources and information. This control plane is more flexible and can be more efficient to handle the dynamic changes of infrastructure like a Fog Computing without resorting to network-wide computations. The distributed approach helps to address the scalability and the locality awareness issues and allows providing services that best fit with the local context, however, no guarantees are provided regarding the global optimality of the computed solutions.

Among the works that considered a distributed control plane, we quote the work of Wang et al. [150] that considers a fog-based architecture composed of fog nodes and fog nodes coordination. The FNs sub-layer deals with the tasks that need real-time processing. The complex tasks that require more computational capabilities are transmitted to the FNs coordination sub-layer or forwarded to the Cloud.

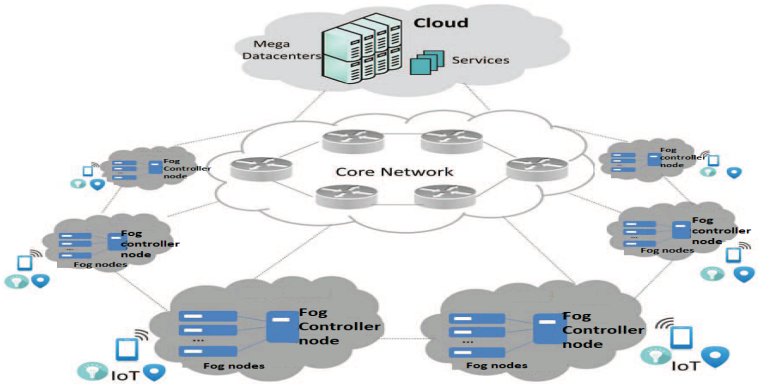


Figure 5: Example of a distributed control plane.

4.2.2 Offline vs. Online placement

The service placement problem can be tackled in an offline or online manner. More precisely, we say that the placement is offline if it takes a deployment decision at the compile-time, where all required information are available. It needs complete information about the system activities and provides solutions that satisfy the given requirements. For online placement, the deployment decisions are made during the run-time of the system. The decisions are based on both process characteristics and the current state of the system. In most real use-cases, the SPP has to be addressed as an online problem. That is, the related algorithms have to consider the services as they arrive, rather than computing the placement in advance before their execution. We notice that an online placement can accommodate dynamic behavior (changes) of the system but it cannot make the best use of system resources (provide optimal placement decision).

As an example of offline placement algorithms provided in the literature, we mention [113, 137] that assumes full information knowledge for the Fog network. For the online placement, we have for instance the work of Lee et al. [87] that proposes an online placement strategy that minimizes the computational latency for Fog system under uncertainty of available FNs. The provided online optimization framework proposes to sequentially observe the information on the network.

4.2.3 Static vs. Dynamic placement

This criterion tackles the fact that proposals handle or not the dynamicity of the system. We can identify respectively two aspects: the dynamicity of Fog infrastructure and the dynamicity of applications. The Fog network is highly dynamic where entities join and leave network due to instability of network links or failures. Resources capabilities can also vary over time. From an application point of view, the application graph structure can evolve over time in response to changes in real-life conditions. New sources or devices may appear or existing ones may disappear (adding new cameras, breakdown of certain components, user's can decide at any time to start or stop sending their data for a service, etc.). In addition, changes in the application information can also be observed (e.g., on the amount of data, component requirements, etc.).

To deal with the dynamic nature of Fog infrastructure and/or applications, it is required to define reactive strategies able to determine when adaptation is required, provide a transparent mechanism, and deliver the satisfying QoS. Thus, an approach is said to be dynamic if the provided placement strategy is able to deploy new services, replace or release services already deployed, in order to meet the QoS constraints and optimize a given objective (if any).

By surveying the literature, we identified a number of works that propose to manage the dynamic nature of Fog environment, among them we mention the work of Yousefpour et al. [163] that proposes a dynamic provisioning of services in Fog infrastructure that satisfies the QoS requirements and the Service Level Agreements (SLA), and minimizes the resources cost. To handle the dynamicity of IoT applications, Mahmud et al. [97] propose a policy that dynamically determines host nodes for the deployed components and handle sudden changes in frequencies of the received services.

4.2.4 Mobility support

Manage mobility is a major challenge in Fog Computing. Provide a solution that supports the mobility of end-users and/or fog devices and ensures that the users always receive the associated services and desired performance without interruptions is a complex issue in Fog Computing. Frequent changes in locations for end nodes (or fog nodes, e.g., smartphone, smart car) can lead to excessive delays or to packet loss. In such a situation, the system manager (coordinator) should be able to move the service transparently from the previous devices to the new ones so as to ensure its continuity.

In [17, 73, 114, 122, 126, 145, 147, 149, 150], for instance, the authors attempt to address the problem of end-user mobility by providing dynamic placement approaches. In [126], Saurez et al. propose to handle the mobility of end-users by providing strategy based on the following decisions: "When-to-Migrate" based on the latency parameter; and "Where-to-Migrate" based on the proximity of a FN to the mobile devices and the current processing node.

4.3 Optimization strategies

Optimizing the service placement problem in a Fog infrastructure has been tackled from several different objectives, with different formulations and diverse algorithm proposals. This section discusses the possible objectives that may pursue in such systems, the metrics considered to evaluate the provided deployment, the problem formulation used by existing proposals, the resolution strategies, and algorithms used to solve the SPP.

4.3.1 Optimization objective and metrics

We propose to present first a global classification of optimization strategies proposed in the literature. On one hand, we distinguish the following two categories: mono-objective and multi-objective optimization. On the other hand, we present the metrics most often considered during optimization.

a) Mono vs. Multi-objective optimization Mono-objective optimization proposes to optimize only one objective function. While multi-objective proposes to optimize simultaneously a collection of objective functions [99]. A first classification of SPP solutions regarding these two optimizations is given in Table 1. Works that have studied both aspects (mono-objective and multi-objective) are marked with an asterisk (*).

Optimization objective	References
Mono-objective	[10, 12, 14, 16, 17, 19, 20, 43, 44], [45]*, [46, 50, 57, 58, 62, 69, 71, 72, 75, 77, 79, 81, 86–89, 93, 94, 96, 97, 101–103, 107, 108, 115, 117, 127, 129, 132–135, 137, 139–141, 143, 145, 147–151, 154, 157, 161–164], [166]*, [167]
Multi-Objectives	[15, 36], [45]*, [98, 118, 122], [166]*

Table 1: Classification regarding the optimization objectives.

b) Optimization metrics We provide hereafter a list of optimization metrics usually considered in the literature in the context of the works provided so far. The optimization can be addressed to maximize or minimize the value of the following metrics.

Metrics		References
Latency		[7, 12, 13, 15, 20, 26, 43, 50, 55, 58, 75, 85–87, 89, 98, 107, 108, 113, 115, 118, 119, 124, 137, 139, 140, 143, 147, 150, 154, 157, 161, 162, 166]
Resource utilization		[10, 15, 16, 71, 72, 81, 88, 88, 102, 117, 122, 133–135, 148]
Cost		[13–15, 17, 19, 28, 34, 36, 46, 57, 62, 69, 71, 79, 98, 103, 112, 114, 118, 132, 145, 149, 151, 156, 163, 168]
Energy consumption		[11, 15, 26, 43–45, 94, 94, 98, 101, 113, 122–124, 141]
Others	Quality-of-experience	[36, 55, 96, 127, 129]
	Congestion ratio	[93, 164, 167]
	Blocking probability / Failed requests / Number of computationally active FNs	[77] / [16] / [97]

Table 2: Optimization metrics.

• **Latency** Low latency for delay-sensitive applications. Achieving lower latency has attracted attention in several surveyed papers. Indeed, several works aim at minimizing services latency deployed on available resources while satisfying the set of requirements and constraints. For instance, in [161, 162], the authors propose to minimize the service delay of deploying IoT applications on the IoT-Fog-Cloud framework.

• **Resource utilization** An important issue in Fog Computing is how to optimize resource utilization while deploying the maximum number of service over appropriate fog nodes. Among the works found in the literature that investigate this goal, we can cite the work of Hong et al. [71] that provides deployment decisions while maximizing the number of satisfied IoT analytics requests. Skarlat et al. in [133] propose

also to maximize the number of satisfied application requests by prioritizing the applications with the closest deadline.

• **Cost** Cost-related factors become very influential in Fog service management, from the service provider's point of view or from the users' point of view. We can identify two main types of costs: the networking cost for the data transmission charges and associated expenses; execution cost related to the computational expenses of Fog nodes. Other expenses can also be identified: costs related to storage, deployment, security safeguards, migration, and so on.

In [163], the authors propose to minimize a total cost that includes the cost of processing and storage in Cloud and Fog, the cost of communication between Fog and Cloud and between Fog nodes, and the communication cost of service deployment from the Fog service controller to FNs.

• **Energy consumption** Energy efficiency is one of the main concerns in IoT systems and a significant performance metric that several works attempt to investigate within the Fog context. The energy consumption encompasses mostly two things. The type of service to process, and the energy consumption at the service level that includes three main aspects: when the service is sent by the end-user to the fog device; when the service is processed by the FN; and when the Fog needs the Cloud. For instance, Sarkar et al. [123, 124], and Nishio et al. [113] investigate the energy consumption issue in the Fog environment by considering energy consumption in both the network and the device side.

• **Other metrics** Other metrics can be considered when addressing the service placement problem. Some of these metrics are quality of experience, congestion ratio, blocking probability, failed requests, etc. Accepted as the user-centric measurement, QoE encapsulates the user's requirements, perceptions, and intentions when deploying services [84]. As an example study, we mention the work done by Mahmud et al. [96] that provides a QoE-aware application placement strategy by which the deployment prioritizes the user expectations. Regarding the congestion ratio, Yu et al. [164] propose to consider the minimum ratio between the flow and the capacity of link to address the service placement and data routing of real-time processing applications in IoT.

4.3.2 Technical formulation

The SPP is generally formalized using one of these two main categories: Integer programming or Constrained optimization. Table 3 groups the surveyed works according to the identified problem formulation briefly described below.

Technical formulation	**Model**	**References**
Integer programming	Integer Linear Programming (ILP)	[19, 20, 34, 57, 71, 72, 77, 87, 88, 96, 101, 102, 107, 108, 122, 127, 133–135, 137, 141, 147, 154, 154, 164]
	Integer Nonlinear Programming (INLP)	[87, 163]
	Mixed Integer Linear Programming (MILP) / Mixed-integer non-linear programming (MINLP)	[14, 16, 17, 52, 110, 166] / [14, 62, 157, 166]
	Mixed Integer Quadratic Programming (MIQP)	[164]
Constrained optimization		[7, 9, 12, 12, 13, 26, 26, 29, 31, 35, 43, 44, 55, 58, 64, 69, 75, 79, 87, 91, 98, 114, 116, 118, 124, 128, 131, 132, 139, 140, 148, 149, 167]
Others	Matching game	[50]
	Machine learning	[36, 145]

	Stochastic optimization / Petri nets / Potential games / Quadratic Assignment Problem / General convex optimization / Linear programming	[115, 168] / [112] / [129] / [103] / [45]
No technical formulation is provided		[55, 73]

Table 3: Classification according to technical formulation.

a) Integer programming Class of problems that encompasses mathematical optimization problems in which some or all of the variables are integers. We have different variants of integer programming approaches. We list the main ones hereafter.

• **Integer Linear Programming (ILP)** This category of problems expresses the optimization of a linear function subject to a set of linear constraints over integer variables. Several works, such as [34, 72, 108, 133, 134], formulate the placement problem with ILP.

• **Integer NonLinear Programming (INLP)** An integer nonlinear program is an ILP which present nonlinear constraints. For instance, in [163], Yousefpour et al. formulate the dynamic Fog service provisioning problem as an INLP.

• **Mixed Integer Linear Programming (MILP) / Mixed Integer NonLinear Programming (MINLP)** The class of a problem known as a Mixed Integer Programming Problem assumes that some decision variables are not discrete. In [16], to study the latency-critical services management in an Edge-Cloud, the authors formulate the problem as a MILP that minimizes the number of failed requests. Mixed Integer NonLinear Programming (MINLP) considers continuous and discrete variables and nonlinear objective function and/or constraints. Due to the high computational complexity for solving this class of problems, most of the work proposes to linearize it into a MILP. As an example, to investigate the cost-efficient service deployment problem in the Fog architecture, Arkian et al. [14] first formulate the cost minimization problem as an MINLP, and then linearize it into MILP to solve it more easily.

• **Mixed Integer Quadratic Programming (MIQP)** This problem refers to optimization problems with quadratic objective function in the integer and in the continuous variables, and linear constraints in the variables of both types. As an example, in [164], the authors formulate the problem of real-time processing applications provisioning in IoT as a MIQP. To overcome the high complexity of solving such a problem, the authors propose to relax some of the constraints and present an approximation scheme.

b) Constrained optimization Constrained optimization is a set of methods designed to find out the best possible values of certain variables (i.e., optimizing process) in the presence of some restrictions (constraints). It uses a set of constraints that can easily be extended further to involve more aspects.

For instance, in [9], Ait-Salaht et al. propose to handle the SPP problem by provided a generic and easy-to-upgrade constraint programming model. Brogi et al. [29, 31] propose a constrained model to determine the feasible deployments (if any) of an application in the Fog infrastructure.

c) Other technical formulations As other formulations found in the literature we quote: Matching Game [63], Markov Decision Process (MDP) [121], stochastic optimization [111], petri nets[4], potential games [104], quadratic assignment problem [32], general convex optimization [27].

For instance, in [50], a matching game is employed to formulate the task placement problem in Mobile Edge Computing systems while minimizing the computation latency. In [145], Urgaonkar et al. model the migration problem as an MDP (also known as reinforcement learning). In [112], the authors use priced timed Petri nets (PTPNs) to study the service placement in Fog Computing while optimizing price and time costs for completing a task.

4.3.3 Resolution strategies

Compute optimal application scheduling in Fog infrastructure is an NP-hard problem [23, 49, 83]. Indeed, several issues complicate the compute of effective services placement in such a context. First, the heterogeneous nature and the limited capacities of most Fog nodes (resource-constrained). Second, the dynamicity of the environment, resources may appear and disappear, others are moving, infrastructure and application information may change over time (e.g., the variation of the workload). Third, the geographical distribution of fog devices over a large-scale infrastructure. Several specificities and constraints that make the SPP problem in Fog environment a challenging task. In the literature, we identify four main approaches used to solve such a problem: exact resolution, approximation strategies, heuristic or meta-heuristic policies. We briefly describe these procedures in the following.

a) Exact solution The definition of an exact solution is often computed by using an ILP solver, or by performing exhaustive research (by enumerating all solutions). Among the works that attempted to solve the SPP in an exact way, we mention the works [102, 122, 133, 141, 147], that use ILP formulation and exact optimization solver to define an optimal solution, and [167] that uses exhaustive placement research.

However, it is important to notice that performing an exact resolution requires long processing time before reaching the optimal solutions and can only be used for small problem instances. Indeed, finding an exact solution can be extremely time-consuming and not appropriate for large problems such as Fog environments. Thus, the main focus of works within the research community is based on providing an effective approximation, heuristic or meta-heuristic approaches where suboptimal solutions can be computed in a short time.

b) Approximations Approximation techniques are used to compute solutions with provable guarantees regarding their distance to the optimal one. Approximations are efficient algorithms that allow to compute suboptimal solutions of NP-hard optimization problems. For example, in [164], the authors propose to use a fully polynomial-time approximation algorithm to address the problem of IoT application provisioning. This approach allows computing a suboptimal solution and bounds for SPP in a relatively small time.

c) Heuristics Because of the scale, the dynamic and mobile aspects of Fog infrastructures, that make exact analysis almost inapplicable, heuristics are often investigated. Designed to obtain a solution in a reasonable time frame, a heuristic is a set of rules and methods that aim at finding a feasible solution for a given problem. However, with heuristic-based solutions, no performance guarantees are provided. As heuristic approaches we can find for instance fail-first/fail-last heuristics used by Brogi et al. in [29, 31]. The authors in this work propose to adopt these two strategies to determine in a relatively short time, a feasible deployment for an application in the FC. The fail-first heuristic is used in that case to selects the undeployed component that has fewer compatible nodes. The fail-last policy sorts the candidate nodes by decreasing the number of end-devices required by a software component, and their hardware capabilities. To guarantee that all requests are satisfied, these heuristics compute the best resource in terms of spatial proximity and the most powerful devices that can support them.

d) Meta-heuristics Typically inspired by nature, the meta-heuristic solutions aim at providing the best solution by iteratively improving the quality of the result and helping the search process to escape from local optima within a reasonable time (unlike heuristics that can be stuck in a local optimum). Several meta-heuristic algorithms are provided in the literature, like Genetic Algorithms [70], Ant Colony Optimization [47], Particle Swarm Optimization [80], or Tabu Search [61]. These algorithms are based on population evolution where, at each evolution, the best-founded population (solution) is kept into the next evolution, in order to define at the end the best solution regarding a given objective (metric). As an example, we mention the work of Skarlat et al. [133] that uses a Genetic Algorithm (GA) to solve the SPP in FC. A GA [153] is an evolutionary algorithm that mimics the process of a natural evolution of chromosome. In their work, the authors assume that each gene in a chromosome denotes a service

placement decision. The placement is iteratively improved according to a fitness function based on the principle of encouragement.

4.4 Evaluation environments

To evaluate the performance of their proposals, the research community uses different programming tools to perform extensive experiments and test their solutions in relevant environments and preferably in realistic setups. We depict thereafter, the most common tools used in the surveyed papers. Table 4 summarizes the programming environment adopted in the literature.

Environment	Evaluation Environment	References
Analytical tool	Java Tool	[29, 163]
	C++	[77, 126, 164]
	Matlab	[43, 44, 69, 79, 101]
	Optimisation engine (IBM CPLEX, Gurobi, Xpress-MP, ...)	[9, 17, 19, 34, 62, 108, 122, 133, 134, 137, 141, 164, 166]
	Others	[15, 43, 57, 103, 167]
Simulator	CloudSim	[118, 127, 135]
	iFogSim	[64, 78, 94, 96, 102, 107, 108, 108, 117, 131–134, 140, 166]
	SimGrid	[154]
	OMNeT++	[12, 26, 114]
	Others (FogTorchΠ, YAFS [90],. . .)	[6, 7, 16, 31, 75, 81, 87, 87–89, 98, 103, 112, 115, 144, 145, 151, 167]
Testbed	Grid'5000 & Fit IOT-Lab	[46]
	OpenStack	[100]
	Adhoc testbed	[55, 58, 60, 71, 72, 149, 157]

Table 4: A Summary of evaluation environments.

4.4.1 Analytical tool

Analytical tool is one of the common approaches used to compute and evaluate the performance of the formulated strategies. The most frequently mentioned tools are: Java [29, 163], C++ [126, 164], Matlab [79]. For instance, Brogi et al. [29] prototype a proof-of-concept Java tool named FogTorch[1] that implements and solve the SPP. These tools are sometimes associated with other tools or APIs such as ILP solvers. As solvers frequently mentioned in the literature we have IBM CPLEX[2] ([133], [108]), the commercial solver Gorubi [3] ([62]), or Choco-Solver [4] ([9]).

4.4.2 Simulator

Another commonly used approach is performing *simulation*. Among the simulators cited in the literature, we find a simulator CloudSim [33] designed for regular cloud environments, most often used with some extensions. A SimGrid[5] framework designed for large-scale distributed systems. A simulator iFogSim [64] designed for Fog Computing, which extends CloudSim. iFogSim is a simulation toolkit proposed by Gupta et al. [64] for evaluating application design and resource management techniques in Fog systems. The simulator performs discrete event simulation and allows users to run applications over Fog infrastructure and measure metrics like latency, energy consumption, and network usage. Other simulators are also cited like network generic simulator OMNeT++[6] (used of instance in [12]); FogTorchΠ

[1] https://github.com/di-unipi-socc/FogTorch
[2] https://www-01.ibm.com/software/commerce/optimization/ cplex-optimizer/.
[3] http://www.gurobi.com
[4] http://www.choco-solver.org
[5] https://github.com/simgrid/simgrid
[6] https://omnetpp.org/

provided by [31] that employs the Monte Carlo method [48] to estimate the QoS-assurance of output deployments; Event-driven simulator based on SimPy[7] performed by Borylo et al. [26], etc.

4.4.3 Experimental testbeds

Finally, we have physical *testbeds*. As realistic environments, we can cite FIT/IoT-LAB [5] and Grid5000 [18] large-scale experimentation environments. Designed for testing the Future Internet of Things, FIT IoT-LAB testbed is a France large scale open-platform that provides access to a variety of fixed and mobile technologies and services (with more than 2700 heterogeneous sensors spread in six different sites). More focused on parallel and distributed computing, Grid'5000 is a scientific instrument for experimental research on large future infrastructures: Clouds, datacenters, HPC Exascale, Big Data infrastructures, networks, etc. It contains a large variety of powerful resources, network connectivity, and storage access, grouped in homogeneous clusters spread in 10 sites in France. We have Software stacks "OpenStack" [130] which is a set of open source software tools for deploying Cloud Computing infrastructures (Resource reservation, disk image deployment, monitoring tools, data collection and storage). We can identify also Cumulus [56] platform for computational offloading at the Edge, or platforms like FED4Fire [2]: largest federation of Next Generation Internet testbeds in Europe, PlanetLab [1]: a testbed for computer networking and distributed systems research, etc.

4.5 Summary

This section describes the SPP and summarizes the most common formulations, resolution strategies, and evaluation tools used in the literature to address this issue. Next, we propose to categorize the surveyed works based on a new classification scheme that allows to simplifying access to references in the category of interest and identifying more easily the challenges and emerging research directions.

5 A classification of service placement approaches

In this section, we use the taxonomy developed in Section 4.2 to provide a new classification scheme and categorize the works provided on the SPP by the research community. First, we propose to categorize the problem on two main scenarios according to the problem description. Then, we provide the classification we perform. The idea is to group the works addressing the same issues to better understand the needs of each problem, and facilitate the user's access to references.

5.1 Identified scenarios

When surveying the literature, we found that depending on the problem's features, we can identify two main scenarios. Scenario 1 that aims at deploying services in Fog while satisfying the QoS requirements and Scenario 2, slightly different from the first one, that in order to ensure minimum latency and satisfactory quality of service must disseminate and deploy services (replicate some services and place them) over Fog infrastructure. The description of these scenarios is given hereafter.

Table 5: Classification of surveyed works according to identified scenarios.

Service model		Papers
Scenario 1	Scenario 1.1	[35, 36, 54, 69, 81, 96, 115, 127, 132, 135–137, 139, 141]
	Scenario 1.2	[7, 10, 14, 16, 26, 43–45, 50, 55, 63, 77, 78, 86–88, 101, 107, 108, 129, 143, 145, 149, 150, 161–164]
	Scenario 1.3	[72, 93, 102, 112, 122, 126, 133, 134, 147, 148]
	Scenario 1.4	[9, 12, 15, 17, 19, 20, 29, 31, 34, 46, 57, 58, 65, 71, 75, 79, 89, 94, 97, 98, 103, 114, 131, 140, 149, 154, 157]
Scenario 2		[11, 13, 52, 100, 128, 166, 167]

[7]https://simpy.readthedocs.io/en/latest/

i) Scenario 1: Assigning services while satisfying requirements and optimizing a given objective (if any) Most of the surveyed works are part of this first scenario, and try to provide answers and address the following question: *"where should the service be deployed and executed to best fit with the objectives"*.

In this scenario, we remark that according to how we characterize the services, we can identify the following sub-scenarios: assignment of set of monolithic applications, assignment of continuous request sent from IoT node to the Fog/Cloud layer, assignment of set of applications each composed by a set of inter-dependant components, assignment of set of applications each having a DAG topology.

- *Scenario 1.1: Deploy a set of monolithic applications.* This scenario addresses the problem of defining the best storage/process location for set of services that are assumed to be monolithic.
- *Scenario 1.2: Deploy applications that receives continuous request from a data source.*
  This scenario deals with the assignment of service flows between service producer (i.e., sensors) and service consumers (i.e., Fog resources). Application placement here involves both determining the host devices and routing path that satisfies requirements and optimize objective (if any).
- *Scenario 1.3: Deploy applications each abstracted as a set of interdependent services.* This scenario assumes that each application is composed of a set of independent components, i.e., without networking dependencies (requirements in terms of latency, bandwidth, etc., between services are not considered).
- *Scenario 1.4: Deploy applications each abstracted as a Directed Acyclic Graph (DAG).* The application here is defined through a DAG topology, where each node represents an operational service of the application and the links describe the networking requirements between components (refer to Section 4.1.2.b for more details).

ii) Scenario 2: Ensure minimum latency and a satisfactory QoS when deploying services Fog Computing allows bringing computational power to the edge of the network, reduce latency and overheads. However, when applications need access to data that are centrally stored (which is the case of many services like video streaming, video on demand, gaming, etc.), the benefits of the Fog can be quickly affected (deterioration of latency, presence of bottlenecks). To avoid these situations, one solution consists to disseminate data in a Fog environment. Offloading data to the Edge involves the use of data replication and place the replicas on critical network nodes to provide a more cost-effective solution.

This scenario tackles the service placement problem in a slightly different context compared to scenario 1. Indeed, where to place the service replicas involves satisfying additional and specific requirements and constraints such as: do not place an identical service in the same place or region, which service replica to select, and so on. Moreover, addressing the SPP here is connect to others issues such as: "Which application components to replicate?", "How many replicas for each service should we create?", "When to create and destroy a copy?", etc. So many factors that make this problem and scenario a very challenging task.

5.2 A classification of SPP according to service placement taxonomy

Based on the identified scenarios and service placement taxonomy presented in Section 4.2, we propose to categorize some approaches elaborated in the literature. The provided classification are depicted in Table 6.

Scenario	Reference	Service placement taxonomy			
		Control plane	Online	Dynamic	Mobility
Scenario 1.1	[69, 141]	C			
	[81, 96, 127, 139]	C	✓		
	[35, 136]	C	✓	✓	

	[36, 115]	C	✓	✓	✓
	[137]	Di			
	[54, 132, 135]	Di	✓		
Scenario 1.2	[14, 26, 43–45, 50, 55, 63, 77, 88, 101, 107, 108, 129, 164]	C	✓		
	[16, 163]	C	✓	✓	
	[145, 149]	C	✓	✓	✓
	[7, 10, 78, 86, 87, 143, 161, 162]	Di	✓		
	[73, 150]	Di	✓	✓	✓
Scenario 1.3	[72]	C			
	[93, 112]	C	✓		
	[122, 126, 147]	C	✓	✓	✓
	[102, 133, 134, 148]	Di	✓		
Scenario 1.4	[9, 20, 29, 31, 57, 71, 75, 79, 98, 103, 118, 140, 154]	C			
	[12, 19, 34, 46, 65, 89, 94, 151, 157]	C	✓		
	[15, 58, 97]	C	✓	✓	
	[17, 114]	C	✓	✓	✓
	[131]	Di	✓		
Scenario 2	[11, 167]	C			
	[52, 100, 166]	C	✓		
	[13, 128]	Di	✓	✓	

Table 6: A classification of SPP according to service placement taxonomy. The ✓ mark means that the criterion is met; otherwise the criterion is not met or not considered.

A more detailed classification is depicted in Tables 7, 8, 9, 10, and 11. In these tables, we propose to outline the placement requirements considered (based on those mentioned in Section 4.1.3), and describe the individual contributions of each work. Table 7 (resp. Table 8, Table 9, and Table 10, 11) is dedicated to approaches dealing with Scenario 1.1 (resp. Scenario 1.2, Scenario 1.3, and Scenario 1.4, and Scenario 2). The following syntax is introduced to associate each work to the related taxonomy:

[C|Di] / [Off|On] / [S|Dy] / [nM|M]

The first parameter denotes whether the considered control plan is **C**entralized or **Di**stributed. The second parameter denotes whether the scheduling is performed in **Off**line or **On**line manner. The third one denotes whether the placement is **S**tatic (i.e., considers unchanging infrastructure and applications topologies and information), or **Dy**namic (i.e., handles the dynamicity of the system). Finally, the fourth parameter denotes whether the provided strategy supports the mobility (**M**) of end-users and/or Fog devices or not (**nM**). So, an approach denoted as C/On/Dy/nM will be a centralized, online, dynamic, and do not support mobility. This description allows categorizing quickly any given proposals and proper compare with similar approaches. We note that each of these categories is mutually independent.

Category	Reference	Requirements	Contributions
C/Off/S/nM	[69]	C_R	Provides data placement policy to help mobile applications by leveraging the edge resources offered by Fog Computing.
	[141]	C_R, C_N	Proposes a framework based on network functions virtualization to deploy services provided by the Cloud onto the FNs.
C/On/S/nM	[81]	C_R, C_D	Provides task scheduling algorithm in Fog based system.
	[96]	C_R, C_N, C_D	Proposes a latency-aware application management policy to achieve improvements in network conditions and service QoS.
	[127]	C_R, C_N	Designs a score-based scheduling framework for latency-sensitive applications that maximize the end-users service quality experienced.

	[139]	C_R, C_N	Introduces an online optimization scheme for the task distribution under uncertainties in Fog network.
C/On/Dy/nM	[35]	C_R, C_N, C_D	Provides an online placement approach that attempts to fairly satisfy all web applications.
C/On/Dy/M	[115]	C_R, C_N, C_L	Designs a dynamic and mobility-aware service placement framework to provide a desirable performance-cost trade-off in Mobile Edge Computing.
	[36]	C_R, C_N	Defines a mobility-aware dynamic service-migration mechanism based on the behavioral cognition of a mobile user in Edge Cognitive Computing.
Di/Off/S/nM	[137]	C_R, C_N, C_D	Provide a new formulation for the QoS-aware service allocation problem for Combined Fog-Cloud architectures.
Di/On/S/nM	[54]	C_R, C_N	Describes an efficient placement of Fog applications modules either on the Edge or in the Cloud.
	[135]	C_R, C_N	Presents a conceptual framework for Fog resource provisioning.
	[132]	C_R, C_D	Proposes a security and deadline aware tasks scheduling algorithm.

Table 7: Taxonomy dedicated to Scenarios 1.1.

Category	**Reference**	**Requirements**	**Contributions**
C/On/S/nM	[14]	C_R, C_N, C_D	Proposes a cost efficient resource provisioning strategy in Fog environment.
	[26]	C_R, C_N, C_D	Provides latency-Aware deployment policy combined with anycast strategies for Fog and Cloud service provisioning.
	[43]	C_R, C_N C_D	Minimizes power costs when distributing workload in Cloud/Fog systems.
	[44]	C_R, C_N C_D	Defines a workload allocation policy for the Fog-Cloud Computing services and investigates the tradeoff between power consumption and delay.
	[45]	C_R	Investigates the problem of service deployment while minimizing carbon footprint for video streaming service in Fog architecture.
	[50]	C_R, C_N, C_L, C_D	Provides a proactive tasks placement in Fog networks under latency and reliability constraints.
	[55]	C_R, C_N, C_D	Provides data placement policy in context of incidental disasters.
	[62]	C_R, C_N, C_D	Proposes a heuristic algorithm to address the task distribution, and virtual machine placement problem toward cost-efficient Fog computation and medical cyber-physical systems.
	[77]	C_R, C_D	Provides a mathematical service placement model in Fog architecture.
	[88]	C_R	Elaborates a data placement policy in Fog architectures that aims to reduce network usage.
	[101]	C_R, C_N, C_D	Proposes a balanced energy-delay solution for IoT applications placement and energy consumption problem in Fog Computing.
	[107, 108]	C_R, C_N	Provides a framework called iFogStor for IoT data placement in a Fog infrastructure.
	[129]	C_R	Provides a placement mechanism that models the competition between IoT users and the efficient service deployment over a hierarchical Fog-Cloud computing system.
	[164]	C_R, C_N, C_D	Proposes provisioning schemes for real-time processing applications in Fog Computing.
C/On/Dy/nM	[16]	C_R, C_N, C_D	Proposes a set of strategies for service placement in Edge-Cloud environment.
	[163]	C_R, C_D	Introduces a dynamic Fog service provisioning policy that meets QoS constraints.
C/On/Dy/M	[145]	C_R, C_N	Designs an online control algorithm that provides where and when services should be migrated according to demand variation and user mobility in Edge-Clouds networks.
	[149]	C_R, C_L	Proposes a strategy that dynamically route data to proper fog nodes in the context of real-time surveillance applications.

Di/On/S/nM	[7]	C_R, C_N, C_L	Proposes an algorithm to distribute workload in Fog Computing environment that minimize the response time and costs.
	[10]	C_R, C_N, C_D	Elaborate an approach for service mapping and service delegation between Fog and Cloud Computing.
	[78]	C_R, C_D	Proposes a QoS-aware service deployment technique in Fog Computing to reduce latency and network congestion.
	[86]	C_R, C_N	Proposes an online optimization framework to perform efficient task distribution over hybrid Fog-Cloud architecture.
	[87]	C_R, C_N	Provides an online dispatching and scheduling mechanism of tasks to distributed Edge-Clouds system.
	[143]	C_R	Develops workload placement algorithms to efficiently execute mobile programs in the Edge-Cloud network.
	[162]	C_R, C_D	Proposes a delay-minimizing policy for IoT applications placement over IoT-Fog-Cloud network.
	[161]	C_R	Proposes a delay-minimizing tasks offloading scheme for IoT applications in Fog Computing.
Di/On/Dy/M	[150]	C_R, C_N, C_D	Proposes a new model to coordinate service deployment and migration that includes latency, location awareness, and mobility support in the smart grid.

Table 8: Taxonomy dedicated to Scenarios 1.2.

Category	**Reference**	**Requirements**	**Contributions**
C/Off/S/nM	[72]	C_R	Proposes an heuristic algorithm to solve service deployment problem in Fog computing.
C/On/S/nM	[93]	C_R, C_N	Designs an algorithm for traffic-aware VM placement and determine the maximum number of accepted VMs in the cloudlet mesh.
	[112]	C_R	Proposes a dynamic service mapping strategy that optimize resource utilization and satisfy the users' QoS requirements in Fog Computing.
C/On/Dy/M	[122]	C_R, C_N, C_D	Evaluates resource provisioning in Smart City scenarios by providing an IoT application service placement mechanism.
	[126]	C_R, C_N, C_L, C_D	Proposes an application deployment and migration mechanism for geo-distributed systems.
	[147]	C_R	Proposes a service placement architecture for the IoT that continuously adapt (migrate) services according to the network changing conditions and users status.
Di/On/S/nM	[102]	C_R, C_N, C_D	Provides a service placement policy that leveraging context-aware information (location, time, and QoS) in Fog landscapes.
	[133, 134]	C_R, C_N, C_D	Proposes provisioning and service placement approach to enable the exploitation of Fog-based computational resources.
	[148]	C_R, C_D	Provides a mapping strategy of IoT applications that optimizes resource utilization and satisfies QoS services requirements.

Table 9: Taxonomy dedicated to Scenario 1.3.

Category	**Reference**	**Requirements**	**Contributions**
C/Off/S/nM	[20]	C_R, C_N	Provides a latency-aware service placement heuristic in Cloud-Fog environment.
	[29, 31]	C_R, C_N, C_L	Proposes a QoS-aware deployment strategy for multi-components IoT applications in Fog infrastructure.
	[98]	C_R, C_N, C_L	Designs a tasks deployment framework for Mobile Edge Cloud Offloading that achieves a trade-off between applications' runtime, mobile device battery lifetime and cost for the user.
	[57]	C_R, C_N	Provides a set of heuristics to address the service placement problem in Cloud-Fog network.

	[71]	C_R, C_N, C_L	Designs, implements, and evaluates a Fog Computing platform that runs analytics on multiple devices.
	[75]	C_R, C_N	Evaluates a set of heuristic algorithms for solving the service placement problem in the context of next-generation network architectures.
	[79]	C_R, C_N	Addresses the task assignment problem for the virtual Fog access point.
	[103]	C_R, C_N	Proposes a tasks deployment policy that assigns the processing tasks to nodes which provide the optimal processing time and near optimal networking costs in Edge-Fog Cloud system.
	[118]	C_R, C_N	Provides a scheduling algorithm that guarantees application performance and reduces the cost of using Cloud resources.
	[140]	C_R	Proposes a service mapping solution for efficient resources utilization in the Fog infrastructure.
	[154]	C_R, C_N, C_L	Proposes a mechanism for placing distributed IoT applications in Fog infrastructure while minimizing the services response time.
C/On/S/nM	[12]	C_R, C_N, C_L	Provides an optimization placement framework of data stream processing applications that minimise end-to-end latency in Edge-Cloud Computing.
	[19]	C_R, C_N, C_D	Designs an IoT service placement strategy for IoT-Cloud infrastructure that satisfies end user demands and minimizes overall operational cost.
	[34]	C_R, C_N	Proposes a solution for the distributed data stream application placement in a geographically distributed environment.
	[46]	C_R, C_N	Proposes an orchestration mechanisms for service provisioning in Fog network that minimizes the provisioning cost of IoT applications.
	[64]	C_R, C_N	Proposes a transfer-time aware service scheduling policy for IoT-Fog-Cloud Computing environments.
	[89]	C_R, C_N, C_D	Proposes a policy for service placement in Fog devices based on communities and transitive closures notions.
	[94]	C_R	Proposes an energy-aware algorithm for mapping application components on Fog devices.
	[151]	C_R, C_N, C_L	Addresses the multi-Component application placement problem in Edge environments and develop algorithms with provable performance bounds.
	[157]	C_R, C_N	Builts the Latency- Aware Video Edge Analytics system to investigate task placement schemes.
C/On/Dy/nM	[15]	C_R, C_L	Proposes a decision-making strategy for virtual machine placement considering multiple optimization objectives.
	[58]	C_R	Proposes adaptive scheduling strategies for dynamic event analytic dataflows placement in Edge-Cloud devices that support Smart City applications emerging needs.
	[97]	C_R, C_D	Refines the service provisioning algorithm to guarantee the application deadlines and optimize Edge resource exploitation.
C/On/Dy/M	[17]	C_R, C_N	Proposes an online mobile application placement algorithm that minimizes the services computational cost.
	[114]	C_R, C_N, C_D	Proposes a service placement and migration policy for mobile event processing applications in Cloud and Fog resources.
Di/On/S/nM	[131]	C_R, C_N	Introduces a collaborative approach of executing applications components in a distributed manner between fog devices.

Table 10: Taxonomy dedicated to Scenarios 1.4.

Category	**Reference**	**Requirements**	**Contributions**
C/Off/S/nM	[11]	C_R, C_N	Proposes an efficient cache placement strategy based on content popularity to reduce energy consumption in Fog networks.
	[167]	C_R,C_N	Proposes a placement algorithm for virtual machine replica in Mobile Edge Computing that minimizes the average data traffic in the network.

C/On/S/nM	[52]	C_R, C_N, C_L	Provides two request-routing strategies to tackle the problem of energy-efficient and latency-aware data placement in geo-distributed Cloud data centers.
	[100]	C_R	Provides a Fog-aware replica placement algorithm based on the definition of failure groups.
	[166]	C_R, C_N	Designs a task scheduling strategy that minimizes tasks completion time for promoting the user experience.
Di/On/Dy/nM	[13]	C_R, C_N, C_L	Elaborates a dynamic placement (creation/replacement/removal) of data replicas across IaaS providers.
	[128]	C_R, C_N, C_L	Provides a bandwidth and availability-aware policy for service deployment on Micro-Cloud infrastructures that maximizes user QoS and QoE.

Table 11: Taxonomy dedicated to Scenarios 2.

5.3 A classification of SPP according to resolution approaches

We propose now to group similar works based on the used resolution approach. Some solutions along with their objectives are detailed in Tables 12, 13, 14, 15, and 16. Each table refers to the aforementioned scenarios (see Section 5.1).

Category	**Solutions**	**References**	**Objective and a brief description of the resolution technique.**
C/Off/S/nM	Exact	[141]	*Minimizes the number of Fog nodes.* Exact resolution of the ILP using CPLEX solver.
		[69]	*Minimizes the overall communication cost.* Uses linear programming solver.
C/On/S/nM	Approximation	[139]	*Minimizes the total response time over all the deployed tasks.* Uses a Dual-fitting algorithm to find a feasible solution.
	Heuristic	[81]	*Maximizes the utilization of residual computing capabilities of terminals.* Provides two heuristics: 1) Prioritize the task with the earliest deadline. 2) Choose the device with the minimum remaining computation capacity.
		[127]	*Maximizes the QoE.* Computes a quality score that combines connectivity, bandwidth, and resource scores to deploy a service on the most suitable VM.
	Meta-heuristic	[96]	*Maximizes QoE-gain of the user.* Uses a Fuzzy logic based reasoning.
C/On/Dy/nM	Heuristic	[35]	*Equally satisfies the applications.* Uses utility-driven application placement policy.
C/On/Dy/M	Heuristic	[115]	*Minimizes the average service latency under cost budget constraints.* Uses Lyapunov optimization to decompose the problem into a set of problems that do not require a priory knowledge of user mobility.
	Machine Learning	[36]	*Minimizes the service costs, meantime and improve the QoE.* Uses Q-learning method [76] to determine for each service request, the host (node) that provides the optimal migration.
Di/Off/S/nM	Exact	[137]	*Minimizes the service latencies in a Fog, while satisfying the QoS requirements.* Uses Gurobi optimizer [66] to solve the ILP provided model.
Di/On/S/nM	Heuristic	[132]	*Minimizes the cost of the user job.* Sorts the jobs in increasing order of deadlines.
		[135]	*Maximizes the utilization of Fog resources.* Uses a fog colonies notion and simulation approach to perform provisioning plan of requested services.
		[54]	*Maximizes the application deployment revenue.* Operates an iterative application deployment by region.

Table 12: Classification according to resolution approaches dedicated to Scenarios 1.1.

Category	Solutions	References	Objective and a brief description of the resolution technique.
C/Off/S/nM	Approximation	[164]	*Finds the minimum congestion ratio.* Uses *fully polynomial-time approximation* that in order to reduce the resolution complexity of IoT application provisioning proposes to decompose the initial problem into two sub-problems to be solved separately.
		[129]	*Each user maximize its own QoE.* Uses ϵ-*Nash equilibrium* and offloading game to model the competition between end-users and provides near-optimal service mapping.
	Heuristic	[43, 44]	*Minimizes the power consumption in the Cloud-Fog Computing.* Decomposes the initial problem into three sub-problems that can be independently solved, and uses convex optimization techniques, Generalized Benders' Decomposition, and Hungarian method to find feasible solution.
		[55]	*Minimizes service delay and expensive resource over provisioning.* Computes the shortest path that satisfies application QoS constraints.
		[77]	*Minimizes the blocking probability* (ratio between a number of rejected workloads and the total number of workloads). Proposes three resolution approaches: Random, Lowest latency, and Maximum available capacity policies.
		[88]	*Optimizes the network usage.* Determines the Fog device with the closest and most evenly distance to the data sources. The placement decision considers the FN with the highest centrality value.
		[107, 108]	*Minimizes the overall latency of storing and retrieving data in a Fog.* Provides a geographical partition to decreases the problem-solving time.
		[14, 26, 45, 50, 62]	
	Meta-heuristic	[101]	*Minimizes the total energy consumption of mobile applications.* Uses a modified genetic algorithm.
C/On/Dy/nM	Heuristic	[16]	*Minimizes failed requests.* Proposes two heuristics: strictest deadline first, and first in first out policies.
		[163]	*Minimizes overall cost (processing, storage, and communication).* Proposes Two heuristics: 1) Min-Viol: aims at minimizing the deadline violations. 2) Min-Cost: aims at minimizing the total cost.
C/On/Dy/M	Heuristic	[145]	*Minimizes the cost of execution, delays, and location constraints.* Decouples the initial Markov Decision Process (MDP) into two independent MDPs and solves the problems using a Lyapunov optimization.
		[149]	*Minimizes the average cost over time.* When emergency event happens, computes reconfiguration and reallocation only on the impacted zone.
Di/On/S/nM	Heuristic	[7]	*Minimizes the response time and maximize the throughput.* Schedules jobs on VMs based on service level agreement.
		[10]	*Meets SLA and QoS.* Prioritizes the mapping based on a linearized decision composed by services size, completion time, and VMs capacity. Component with higher priority are mapped first and the one with lower priority are deployed last.
		[143]	*Minimizes the cumulative delay of executing mobile services.* Solves the MNIP problem as follows: first transforms the mixed nonlinear integer program to a convex optimization problem, and then solves the problem that only contains the integer variables.
		[162]	*Reduces the service delay for IoT applications.* Compares the estimated waiting time of task at a given FN with their deadline. If it is smaller, accept the task, if not, the FN offloads the service to one of its neighbors, or to the Cloud.
		[78, 86, 87, 161]	
Di/On/Dy/M	Heuristic	[150]	*Reduces the application delay of IoT applications in the Smart Grid.* Chooses the best nodes from the set of nodes that satisfy QoS of end-nodes and having the nearest deadline.

Table 13: Resolution approaches dedicated to Scenarios 1.2.

Category	Solutions	References	Objective and a brief description of the resolution technique.
C/Off/S/nM	Heuristic	[72]	*Maximizes the number of satisfied services.* Considers the applications with fewer components first (if the same, takes those who have the least feasible resources).
C/On/S/nM	Heuristic	[93]	*Minimizes the total inter-cloudlet communication traffic in cloudlet mesh.* Iterates on jobs and provides placement that minimize inter-cloudlet communication and satisfies the communication demands and resource constraints.
C/On/Dy/M	Heuristic	[122]	*Optimizes multiple objectives: maximize number of accepted IoT application requests, maximize service bandwidth, minimize service migrations between iterations, minimize number of active computational nodes, minimize the number of active gateways, minimize hop count between computational nodes and end devices, and minimize of path loss.* Computes iteratively the solution so that in every iteration refines the previous obtained solution by improving the model with an additional optimization objective.
		[126]	*Optimizes tasks mapping by saving bandwidth and reducing latency.* Determines incrementally the fog nodes that match the capacity constraints and handle the application dynamism.
		[147]	*Minimizes: the hop count between end-nodes and hosting nodes, the hop count between communication nodes, and the number of service migrations.* Deploys the services in random locations, and later adapts to the network and applications constraints and requirements.
Di/On/S/nM	Exact	[102]	*Maximizes the number of deployed applications.* Uses ILP solver.
		[134]	*Maximizes the number of services deployed on Fog landscape.* Uses ILP solver.
	Heuristic	[134]	*Maximizes the number of deployed services to Fog infrastructure.* Prioritizes the applications having the minimum value between their deadline and their deployment time.
	Meta-heuristic	[133]	*Maximizes the number of deployed services to Fog devices rather than to Cloud ones.* Uses a genetic Algorithm.

Table 14: Classification according to resolution approaches dedicated to Scenarios 1.3.

Category	Solutions	References	Objective and a brief description of the resolution techniques.
C/Off/S/nM	Exact	[9]	*Provides feasible (resp. optimal) service placement solutions in Fog environment.* Uses the constraint programming Choco-solver.
		[20]	*Minimizes the overall latency and ensures the QoS requirements.* Uses ILP-solver CPLEX.
	Heuristic	[29]	*Determines eligible deployments of composite applications.* Performs pre-processing plus backtracking to determine an eligible deployment.
		[31]	*Determines eligible deployments of composite applications.* Exploits Monte Carlo simulations [48] to handle the communication links variations, and performs pre-processing plus backtracking to determine the final eligible deployment.
		[98]	*Optimizes the following objectives: minimize runtime and user cost, and maximize battery lifetime.* Computes the local optimum solution for each objective, and then, selects the one with the lowest score (calculated according to a given equation).
		[57]	*Minimizes the overall cost (placement and link costs).* Provides six heuristics: 1) Limits the deployment of an application component to a subset of nodes; 2) Restricts the placement of a component to one particular node; 3) Applies the co-location of some components to one node; Accelerate the previous heuristics by 4) Combines heuristics 2 and 1; 5) Combine heuristics 2 and 3; and 6) Combines heuristics 1 and 3.
		[71]	*Maximizes the number of satisfied IoT analytics.* Prioritizes the scarcest resource first and the closer to source device next.

		[75]	*Minimizes end-to-end delay.* Elaborates a layered graph placement algorithm that proposes to find a lowest cost path that includes communication cost and processing cost.
		[79]	*Minimizes maximum cost service node.* Selects the mapping with the minimum total cost by iterating on the set of all possible mappings.
		[103]	*Minimizes network cost.* Minimizes processing cost first and then optimizes the network cost.
		[140]	*Optimizes utilization of network resources.* Prioritizes the components placement based on the resource expectation.
		[154]	*Minimises the average response time.* Proposes three heuristics based on backtracking solution and the notion of anchor.
C/On/S/nM	Exact	[19]	*Minimizes overall operational cost.* Uses the linear programming solver Xpress-MP.
		[34]	*Minimizes the application end-to-end latency.* Uses the ILP solver.
	Appro.	[151]	*Minimizes the maximum weighted cost on network nodes and links.* Provides polynomial-logarithmic worst-case optimality bound. Splits the application graph into simple branches, and solve the problem recursively.
	Heuristic	[12]	*Minimizes end-to-end latency.* Breakdowns the latency calculation into computational latency and network transfer latency, and minimizes the sum.
		[46]	*Minimizes the provisioning cost.* Adopts a divide and conquer approach and incrementally computes the best solution.
		[65]	*Determines an eligible application placement.* Prioritizes interdependent components based on the computation cost and communication time.
		[89]	*Minimizes the network delays between interrelated services while optimizing the QoS and the service availability for the users.* Uses a first fit decreasing approach to place applications in device communities, and then prioritizes the applications with the shortest deadlines.
		[94]	*Minimizes energy consumption.* Deploys the incoming application components to Fog resources based on the remaining CPU, energy consumption, and related deadline.
		[157]	*Minimizes the response time.* Performs sequential quadratic programming and divides the problem into two sub-problems to minimize computational complexity.
C/On/Dy/nM	Heuristic	[97]	*Minimizes the network cost during the task assignment.* Proposes to minimize first the processing cost and then optimizes the network cost.
	Meta-heuristic	[15]	*Optimizes a multi-objectives function: Minimize cost, maximize user support, minimize latency, and maximize user footprint.* Converts the multi-objective optimization problem to a single objective problem by using a scalarization method. And provides Pareto optimal solution.
		[58]	*Minimizes the total makespan while meeting energy and QoS constraints.* Proposes two prior GA meta-heuristics (GA-Incremental and GA-Global) to support dynamically the multiple dataflows arriving and departing.
C/On/Dy/M	Heuristic	[17]	*Minimizes the cost to run the application.* Performs an iterative matching process and local search phase to compute the best solution.
		[114]	*Minimizes the costs of migration and placement of a single component.* Creates time-graph model and considers the shortest path from data source to identify possible migration nodes.
Di/On/S/nM	Heuristic	[131]	*Minimizes the cost to run the application.* Prioritizes the placement according to dependencies between the components and computational power of edge devices.

Table 15: Resolution approaches dedicated to Scenario 1.4.

Category	Solutions	References	Objective and a brief description of the resolution technique.
C/Off/S/nM	Exact	[167]	*Minimizes the average data traffic in the Edge environment.* Exhaustive research is performed, i.e., enumerates all placements of service replica and selects the solution that minimizes the objective among all computed placement solutions.
	Heuristic	[167]	*Minimizes the overall latency of storing and retrieving data in a Fog.* Splits the original service placement problem into set of sub-problems each performing an optimal placement solution for one components.
		[11]	*Maximizes the energy efficiency while maintaining the successful delivery.* The provided algorithm proposes to categorize the services into three popularity levels and strategically cache them in Fog resources.
C/On/S/nM	Heuristic	[166]	*Minimizes the maximum average task completion time.* First, partition the problem on two sub-problems and optimize each of them. Then, re-couple the two solutions in order to optimize the main objective.
		[100]	*Achieves minimal latency in between the replicas and between the replicas and the data sources and sink.* Exploits end-users and service locality in Fog network when deploying.
Di/On/Dy/nM	Heuristic	[13]	*Defines a trade-off between cost and latency.* Evaluates cost of storing replicas and expected latency improvement, to make a migration or duplication decision.
		[128]	*Maximizes the end-to-end performance.* Provides bandwidth and availability-aware service placement policy.

Table 16: Classification according to resolution approaches dedicated to Scenarios 2.

Based on the provided tables, we propose in the next section to discuss the open research directions and the challenges related to SPP in Fog Computing.

6 Emerging research directions

This section discusses the current limitations and highlights the challenges related to service placement problem in the Fog Computing. Three main directions that need attention in the near future are identified: challenges related to the problem statement, optimization strategies, and evaluation environment.

6.1 Challenges related to problem statement

Despite the recent research efforts outlined on SPP, many challenges still remain open, and one of them concerns the "problem statement" i.e., which problematic we try to address (fits which scenario), which important information needs to be considered (infrastructure information, application description, mapping requirements), and under which aspects to address it (taxonomy). Here we present some of the identified open problems.

6.1.1 Scenario that considers dependencies at the data level.

Based on the surveyed papers, we identified two main scenarios considered by the research community (see section 5.1), however, we noticed that for the most works that fit into the aforementioned scenarios, an important use-case was not really (and marginally) considered in SPP. It corresponds to "computation placement with data dependencies" that represents one of the problems that motivate the Fog since the goal of the Fog Computing is to keep the services close to the devices that produce and act on the data. Thus, when deploying services, the mapping must consider the dependencies at the data level, either in terms of locality (e.g., if a service is deployed in a particular zone, the related components must also be deployed in the same area for security reasons for instance) or in term of minimizing the flow of data exchanged, etc. Marginally addressed in the literature (to the best of our knowledge), the

application placement with data dependencies represents real challenges that need to be more considered and studied.

6.1.2 Distributed service deployment

As mentioned earlier in the paper, distributing the making decisions to multiple substrate nodes instead of relying the mapping on a single central node helps to spread the load and possibly increasing scalability. With infrastructure such as Fog Computing, there is a tendency to believe that most works are based on this process. However, according to Tables 6, 7, 8, 9, 10, and 11 we observe that there is a lack of solutions proposed in the literature that address the SPP in distributed way. This is mainly due to that distributed algorithms are difficult to construct and their implementations are often non-trivial to achieve in a real environment due to the complexity of the inter-process communication and synchronization. Moreover, the lack of knowledge of the global state makes difficult to compute optimal solutions and even to reach near-optimal placement. We remark also that only a few approaches perform SPP in a distributed and dynamic way [13, 73, 128, 150]. And still less that considers distributed solutions that handle dynamicity of the system and support mobility of end-users/FNs ([73, 150] for scenario 1.2). Thus, research in these directions is still open.

6.2 Challenges related to optimization strategies

We detail hereafter some of the challenges and opportunities worth further research in terms of optimization strategies point of view.

6.2.1 Optimized objective.

Depending on the studied problem and considered use-case, different metrics can be observed: Latency/bandwidth between end-user, robustness in case of failures, energy consumption/battery life cycle of the sensors, cost, application-specific metric (frame/packet loss, end-to-end delay in processing some data), etc. According to Table 1, we can easily observe that the performance metrics are usually taken as individual objectives (minimize latency, maximize the number of satisfied applications, minimize energy consumption, etc.). However, to improve QoS and QoE, these metrics should be simultaneously considered in the objective function, instead of integrating them into the model as constraints functions. Considering multiple objectives at the same time has not received significant attention so far. Due to the complexity of such problems, only a few works have attempted to address such issues [15, 36, 45, 98, 118, 122, 166]. Indeed, conduct multi-objective optimization and derive effective solutions requires a huge computational effort [37]. Multiobjective metaheuristics can be explored to solve such a problem, like MOPSO [68] and NSGA-II [42], or scheduling approaches like [53].

6.2.2 Solution that support mobility

Due to the high mobility of some end-users and Fog devices, the elaborated solutions must ensure the continuity of the services and that the users always receive the desired requests. For this, the services must follow this mobility and perform some migrations across different Fog instances. When reviewing the literature, we observed that only a few works propose to provide a solution that supports the mobility of end-users/FNs (see Table 6). Moreover, most of the dynamic application migration approaches elaborated consider a restricted application topology (monolithic, line or tree graph), support only the mobility of end-users (not those of FNs), and developed dynamic solutions based on complex techniques and algorithms. In view of that, it is clear that the definition of standard algorithmic techniques appropriate for implementation in real Fog systems is a real need for the community. Moreover, we have observed that currently, most of the placement techniques that support mobility are reactive. Thus, addressing the problem from a proactive point of view by predicting the mobility pattern of users' and devices' could be more suitable in the Fog context. Indeed, understanding the movement behavior of end devices (or FN) may be helpful for an efficient service placement and service management in Fog Computing.

6.2.3 Energy-efficiency.

Regarding the energy consumption in Fog service management, we found that research is less prevalent in some aspects of energy consumption in the Fog environment. We identify the following factors: a model that considers the carbon footprint, uses renewable energy sources like wind turbines or solar panels, etc., take into account the energy-constrained of end devices (sensors) in terms of residual battery lifetime, the energy of communication links; etc. The use of Follow the Sun and Follow the Moon strategies in these cases could be helpful to manage efficiently the energy consumption, for instance by controlling the devices in terms of intelligent lighting, ventilation, and air conditioning, etc.

6.2.4 Generic and effective mapping

It is clearly seen in Section 4.3 that many formulation and solutions are developed to address the SPP for a specific application. These placement policies, given in the literature (mainly heuristics), consider different assumptions (infrastructure information, application topology, QoS attributes and metrics, ...), different objectives, that make them not easily comparable. Indeed, given the diversity of criteria, handling all these parameters is practically impossible. So, the questions arising today are: *which criteria are most significant and should be considered in order to develop an effective solution? according to these criteria can we compare the existing proposals and identify the relevant approaches? or should we make a clean sweep and propose a new generic and easy to upgrade methodology?*. Many open issues that deserve to be deepened.

6.3 Challenges related to evaluation environments

We propose to describe here one of the challenges that in our opinion is the most important regarding the evaluation environments' point of view.

6.3.1 Uniform environment

Through Table 4, we can easily observe that different tools are used by the research community to test and perform their experiments. Each tool has its own specificities and is adapted to a given problem (use-case) as described in Section 4.4. While there are a number of works that have addressed the SPP challenge, we notice that there does not yet a generic development environment that handles a large range of standardized IoT applications and allows considering various network topologies. So, there is a requirement of making a uniform platform involving most of the concepts, easy to take in hand and that favors the realization of extensive experiences. Based on the classification scheme and the scenarios provided in Section 5.1, our contribution through this paper consists to bring some basics on which we can rely to design a generic and extensible model that will take over the different Fog Computing use cases.

7 Conclusion

This paper focuses on the Service Placement Problem (SPP) in a Fog environment, which is currently an open issue that calls for extensive discussions and solutions. This paper gives a survey of current works. A description of the SPP was provided. Five scenarios related to this issue were identified. A categorization of solutions along four distinct dimensions (centralized vs. distributed control plan, offline vs. online scheduling, static vs. dynamic system, not support vs. support mobility of end-users/fog nodes) was elaborated. A number of algorithmic proposals to the SPP were discussed. Finally, these aspects were used to create a classification of SPP solutions elaborated in the literature based on placement taxonomy. More precisely and compared to existing surveys, our work highlights a new classification scheme which aims to simplify the user's access to references in a specific context and identify more easily the placement-related challenges.

References

[1] [n. d.]. "PlanetLab". https://www.planet-lab.org

[2] 2014. "Fed4fire project". http://www.fed4fire.eu/

[3] M. Abderrahim, M. Ouzzif, K. Guillouard, J. François, A. Lebre, C. Prud'homme, and X. Lorca. 2019. "Efficient Resource Allocation for Multi-Tenant Monitoring of Edge Infrastructures". In *PDP*. 158–165. https://doi.org/10.1109/EMPDP.2019.8671621

[4] P. A. Abdulla and R. Mayr. 2013. "Priced Timed Petri Nets". *Logical Methods in Computer Science* 9, 4 (2013). https://doi.org/10.2168/LMCS-9(4:10)2013

[5] C. Adjih, E. Baccelli, E. Fleury, G. Harter, N. Mitton, T. Noel, R. Pissard-Gibollet, F. Saint-Marcel, G. Schreiner, J. Vandaele, and T. Watteyne. 2015. FIT IoT-LAB: A large scale open experimental IoT testbed. In *IEEE WF-IoT*. 459–464.

[6] S. Agarwal, J. Dunagan, N. Jain, S. Saroiu, and A. Wolman. 2010. "Volley: Automated Data Placement for Geo-Distributed Cloud Services". In *7th USENIX Symposium on NSDI*. 17–32.

[7] Swati Agarwal, Shashank Yadav, and Arun Yadav. 2016. "An Efficient Architecture and Algorithm for Resource Provisioning in Fog Computing". *IJIEEB* 8 (01 2016), 48–61. https://doi.org/10.5815/ijieeb.2016.01.06

[8] Y. Ai, M. Peng, and K. Zhang. 2017. "Edge Computing Technologies for Internet of Things: A Primer". *Digital Communications and Networks* (2017). http://www.sciencedirect.com/science/article/pii/S2352864817301335

[9] Farah Aït-Salaht, Frédéric Desprez, Adrien Lebre, Charles Prud'homme, and Mohamed Abderrahim. 2019. Service Placement in Fog Computing Using Constraint Programming. In *2019 IEEE International Conference on Services Computing, SCC 2019, Milan, Italy, July 8-13, 2019*. 19–27. https://doi.org/10.1109/SCC.2019.00017

[10] A. Abdullah Alsaffar, P. Phuoc Hung, C. Seon Hong, E.-N. Huh, and M. Aazam. 2016. "An Architecture of IoT Service Delegation and Resource Allocation Based on Collaboration between Fog and Cloud Computing". *Mobile Information Systems* 2016 (2016), 6123234:1–6123234:15. https://doi.org/10.1155/2016/6123234

[11] I. Althamary, C.-W. Huang, P. Lin, S.-R. Yang, and C.-W. Cheng. 2018. "Popularity-Based Cache Placement for Fog Networks". In *14th IWCMC*. 800–804. https://doi.org/10.1109/IWCMC.2018.8450495

[12] G. Amarasinghe, M. D. de Assuncao, A. Harwood, and S. Karunasekera. 2018. "A Data Stream Processing Optimisation Framework for Edge Computing Applications". In *IEEE 21st ISORC*, Vol. 00. 91–98. https://doi.org/10.1109/ISORC.2018.00020

[13] A. Aral and T. Ovatman. 2018. "A Decentralized Replica Placement Algorithm for Edge Computing". *IEEE Transactions on Network and Service Management* 15, 2 (June 2018), 516–529. https://doi.org/10.1109/TNSM.2017.2788945

[14] H. Reza Arkian, A. Diyanat, and A. Pourkhalili. 2017. "MIST: Fog-Based Data Analytics Scheme with Cost-Efficient Resource Provisioning for IoT Crowdsensing Applications". *Journal of Network and Computer Applications* 82 (2017), 152–165. https://doi.org/10.1016/j.jnca.2017.01.012

[15] R. G. Aryal and J. Altmann. 2018. "Dynamic Application Deployment in Federations of Clouds and Edge Resources Using a Multiobjective Optimization AI Algorithm". In *FMEC*. 147–154. https://doi.org/10.1109/FMEC.2018.8364057

[16] O. Ascigil, T. K. Phan, A. G. Tasiopoulos, V. Sourlas, I. Psaras, and G. Pavlou. 2017. "On Uncoordinated Service Placement in Edge-Clouds". In *IEEE CloudCom*. 41–48.

[17] T. Bahreini and D. Grosu. 2017. "Efficient Placement of Multi-component Applications in Edge Computing Systems". In *ACM/IEEE Symposium on Edge Computing (SEC '17)*. ACM, New York, NY, USA, Article 5, 11 pages. https://doi.org/10.1145/3132211.3134454

[18] D. Balouek, A.Carpen-Amarie, G. Charrier, F. Desprez, E. Jeannot, E. Jeanvoine, A. Lebre, D. Margery, N. Niclausse, L. Nussbaum, O. Richard, C. Pérez, F. Quesnel, C. Rohr, and L. Sarzyniec. 2013. Adding Virtualization Capabilities to the Grid'5000 Testbed. In *Cloud Computing and Services Science*. Communications in Computer and Information Science, Vol. 367. Springer International Publishing, 3–20. https://doi.org/10.1007/978-3-319-04519-1_1

[19] M. Barcelo, A. Correa, J. Llorca, A. M. Tulino, J. L. Vicario, and A. Morell. 2016. "IoT-Cloud Service Optimization in Next Generation Smart Environments". *IEEE Journal on Selected Areas in Communications* 34, 12 (Dec 2016), 4077–4090. https://doi.org/10.1109/JSAC.2016.2621398

[20] A.R. Benamer, H. Teyeb, and N. Ben Hadj-Alouane. 2018. "Latency-Aware Placement Heuristic in Fog Computing Environment". In *OTM*. Springer International Publishing, Cham, 241–257.

[21] Kashif Bilal, Osman Khalid, Aiman Erbad, and Samee U. Khan. 2018. Potentials, trends, and prospects in edge technologies: Fog, cloudlet, mobile edge, and micro data centers. *Computer Networks* 130 (2018), 94 – 120. https://doi.org/10.1016/j.comnet.2017.10.002

[22] L. F. Bittencourt, J. Diaz-Montes, R. Buyya, O. F. Rana, and M. Parashar. 2017. "Mobility-Aware Application Scheduling in Fog Computing". *IEEE Cloud Computing* 4, 2 (March 2017), 26–35. https://doi.org/10.1109/MCC.2017.27

[23] S.H. Bokhari. 1981. On the Mapping Problem. *IEEE Trans. Comput.* C-30, 3 (March 1981), 207–214. https://doi.org/10.1109/TC.1981.1675756

[24] F. Bonomi, R. Milito, P. Natarajan, and J. Zhu. 2014. *"Fog Computing: A Platform for Internet of Things and Analytics"*. Springer International Publishing, Cham, 169–186.

[25] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli. 2012. "Fog Computing and Its Role in the Internet of Things". In *MCC*. ACM, New York, NY, USA, 13–16.

[26] P. Borylo, A. Lason, J. Rzasa, A. Szymanski, and A. Jajszczyk. 2016. "Energy-Aware Fog and Cloud Interplay Supported by Wide Area Software Defined Networking". In *IEEE ICC*. 1–7.

[27] S. Boyd and L. Vandenberghe. 2004. *"Convex Optimization"*. Cambridge University Press, New York, NY, USA.

[28] D. Breitgand and A. Epstein. 2011. "SLA-Aware Placement of Multi-Virtual Machine Elastic Services in Compute Clouds". In *IFIP/IEEE IM*. 161–168.

[29] A. Brogi and S. Forti. 2017. "QoS-Aware Deployment of IoT Applications Through the Fog". *IEEE Internet of Things Journal* 4, 5 (Oct 2017), 1185–1192.

[30] A. Brogi, S. Forti, C. Guerrero, and I. Lera. 2019. "How to Place Your Apps in the Fog - State of the Art and Open Challenges". *CoRR* abs/1901.05717 (2019).

[31] A. Brogi, S. Forti, and A. Ibrahim. 2017. "How to Best Deploy Your Fog Applications, Probably". In *IEEE ICFEC*. 105–114.

[32] R. E. Burkard, E. Çela, P. M. Pardalos, and L. S. Pitsoulis. 1998. "The Quadratic Assignment Problem".

[33] R.-N. Calheiros, R. Ranjan, A. Beloglazov, C.A.F. De Rose, and R. Buyya. 2011. "CloudSim: A Toolkit for Modeling and Simulation of Cloud Computing Environments and Evaluation of Resource Provisioning Algorithms". *Softw. Pract. Exper.* 41, 1 (Jan. 2011), 23–50.

[34] V. Cardellini, V. Grassi, F. Lo Presti, and M. Nardelli. 2016. "Optimal Operator Placement for Distributed Stream Processing Applications". In *ACM DEBS*. ACM, New York, NY, USA, 69–80.

[35] D. Carrera, M. Steinder, I. Whalley, J. Torres, and E. Ayguade. 2008. "Utility-Based Placement of Dynamic Web Applications with Fairness Goals". In *IEEE NOMS*. 9–16. https://doi.org/10.1109/NOMS.2008.4575111

[36] M. Chen, W. Li, G. Fortino, Y. Hao, L. Hu, and I. Humar. 2018. "A Dynamic Service-Migration Mechanism in Edge Cognitive Computing". *CoRR* abs/1808.07198 (2018).

[37] G. Chiandussi, M. Codegone, S. Ferrero, and F.E. Varesio. 2012. "Comparison of Multi-Objective Optimization Methodologies for Engineering Applications". *Computers & Mathematics with Applications* 63, 5 (2012), 912 – 942. https://doi.org/10.1016/j.camwa.2011.11.057

[38] M. Chiang and T. Zhang. 2016. "Fog and IoT: An Overview of Research Opportunities". *IEEE Internet of Things Journal* 3, 6 (Dec 2016), 854–864. https://doi.org/10.1109/JIOT.2016.2584538

[39] Cisco. [n. d.]. "Fog Computing and the Internet of Things: Extend the Cloud to Where the Things Are. White Paper. 2016. Available online:". http://www.cisco.com/c/dam/en_us/solutions/trends/iot/docs/computing-overview.pdf

[40] A. V. Dastjerdi and R. Buyya. 2016. "Fog Computing: Helping the Internet of Things Realize Its Potential". *Computer* 49, 8 (Aug 2016), 112–116. https://doi.org/10.1109/MC.2016.245

[41] A. Vahid Dastjerdi, H. Gupta, R.N. Calheiros, S.K. Ghosh, and R. Buyya. 2016. "Fog Computing: Principles, Architectures, and Applications". *CoRR* abs/1601.02752 (2016). http://arxiv.org/abs/1601.02752

[42] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation* 6, 2 (April 2002), 182–197. https://doi.org/10.1109/4235.996017

[43] R. Deng, R. Lu, C. Lai, and T. H. Luan. 2015. "Towards Power Consumption-Delay Tradeoff by Workload Allocation in Cloud-Fog Computing". In *IEEE ICC*. 3909–3914. https://doi.org/10.1109/ICC.2015.7248934

[44] R. Deng, R. Lu, C. Lai, T. H. Luan, and H. Liang. 2016. "Optimal Workload Allocation in Fog-Cloud Computing Toward Balanced Delay and Power Consumption". *IEEE Internet of Things Journal* 3, 6 (Dec 2016), 1171–1181. https://doi.org/10.1109/JIOT.2016.2565516

[45] C.T. Do, N.H. Tran, C. Pham, M.G.R. Alam, J.H. Son, and C. S. Hong. 2015. "Approximal Algorithm for Joint Resource Allocation and Minimizing Carbon Footprint in Geo-Distributed Fog Computing". In *ICOIN*. 324–329. https://doi.org/10.1109/ICOIN.2015.7057905

[46] B. Donassolo, I. Fajjari, A. Legrand, and P. Mertikopoulos. 2019. "Fog Based Framework for IoT Service Provisioning". In *IEEE CCNC*. Las Vegas, United States.

[47] M. Dorigo. 1992. *"Optimization, Learning and Natural Algorithms"*. Ph.D. Dissertation. Politecnico di Milano.

[48] W.L. Dunn and J.K. Shultis. 2011. *"Exploring Monte Carlo Methods"*. Elsevier Science & Technology.

[49] R. Eidenbenz and T. Locher. 2016. "Task Allocation for Distributed Stream Processing". *CoRR* abs/1601.06060 (2016).

[50] M. Saad ElBamby, M. Bennis, and W. Saad. 2017. "Proactive Edge Computing in Latency-Constrained Fog Networks". *CoRR* abs/1704.06749 (2017).

[51] E. Elmroth, P. Leitner, S. Schulte, and S. Venugopal. 2017. "Connecting Fog and Cloud Computing". *IEEE Cloud Computing* 4, 2 (March 2017), 22–25. https://doi.org/10.1109/MCC.2017.29

[52] Y. Fan, J. Chen, L. Wang, and Z. Cao. 2018. "Energy-Efficient and Latency-Aware Data Placement for Geo-Distributed Cloud Data Centers". In *Communications and Networking*. Springer International Publishing, Cham, 465–474.

[53] H. M. Fard, R. Prodan, and T. Fahringer. 2014. Multi-objective list scheduling of workflow applications in distributed computing infrastructures. *JPDC* 74, 3 (2014), 2152 – 2165. https://doi.org/10.1016/j.jpdc.2013.12.004

[54] F. Faticanti, F. De Pellegrini, D. Siracusa, D. Santoro, and S. Cretti. 2018. "Cutting Throughput on the Edge: App-Aware Placement in Fog Computing". *CoRR* abs/1810.04442 (2018).

[55] R. Gargees, B. Morago, R. Pelapur, D. Chemodanov, P. Calyam, Z. Oraibi, Y. Duan, G. Seetharaman, and K. Palaniappan. 2017. "Incident-Supporting Visual Cloud Computing Utilizing Software-Defined Networking". *IEEE TCSVT* 27, 1 (Jan 2017), 182–197. https://doi.org/10.1109/TCSVT.2016.2564898

[56] H. Gedawy, S. Tariq, A. Mtibaa, and K. Harras. 2016. "Cumulus: A distributed and flexible computing testbed for edge cloud computational offloading". In *CIoT*. 1–6.

[57] J. Gedeon, M. Stein, L. Wang, and M. Mühlhäuser. 2018. "On Scalable In-Network Operator Placement for Edge Computing". In *ICCCN*. 1–9. https://doi.org/10.1109/ICCCN.2018.8487419

[58] R. Ghosh, S. Prakash Reddy Komma, and Y. Simmhan. 2018. "Adaptive Energy-aware Scheduling of Dynamic Event Analytics across Edge and Cloud Resources". *CoRR* abs/1801.01087 (2018).

[59] T. N. Gia, M. Jiang, A. M. Rahmani, T. Westerlund, P. Liljeberg, and H. Tenhunen. 2015. "Fog Computing in Healthcare Internet of Things: A Case Study on ECG Feature Extraction". In *IEEE CIT/IUCC/DASC/PICom*. 356–363. https://doi.org/10.1109/CIT/IUCC/DASC/PICOM.2015.51

[60] N. K. Giang, M. Blackstock, R. Lea, and V. C. M. Leung. 2015. "Developing IoT applications in the Fog: A Distributed Dataflow approach". In *International Conference on the IOT*. 155–162.

[61] F. Glover. 1986. "Future Paths for Integer Programming and Links to Artificial Intelligence". *Comput. Oper. Res.* 13, 5 (May 1986), 533–549. https://doi.org/10.1016/0305-0548(86)90048-1

[62] L. Gu, D. Zeng, S. Guo, A. Barnawi, and Y. Xiang. 2017. "Cost Efficient Resource Management in Fog Computing Supported Medical Cyber-Physical System". *IEEE Transactions on Emerging Topics in Computing* 5, 1 (Jan 2017), 108–119.

[63] Y. Gu, W. Saad, M. Bennis, M. Debbah, and Z. Han. 2015. "Matching Theory for Future Wireless Networks: Fundamentals and Applications". *IEEE Communications Magazine* 53, 5 (May 2015), 52–59. https://doi.org/10.1109/MCOM.2015.7105641

[64] H. Gupta, A. Vahid Dastjerdi, S.-K. Ghosh, and R. Buyya. 2016. "iFogSim: A Toolkit for Modeling and Simulation of Resource Management Techniques in Internet of Things, Edge and Fog Computing Environments". *CoRR* abs/1606.02007 (2016).

[65] H. Gupta, S. Brata Nath, S. Chakraborty, and S.K. Ghosh. 2016. "SDFog: A Software Defined Computing Architecture for QoS Aware Service Orchestration over Edge Devices". *CoRR* abs/1609.01190 (2016). arXiv:1609.01190 http://arxiv.org/abs/1609.01190

[66] LLC Gurobi Optimization. 2018. Gurobi Optimizer Reference Manual. http://www.gurobi.com

[67] K. Ha, Z. Chen, W. Hu, W. Richter, P. Pillai, and M. Satyanarayanan. 2014. "Towards Wearable Cognitive Assistance". In *MobiSys*. ACM, New York, NY, USA, 68–81. https://doi.org/10.1145/2594368.2594383

[68] J. Hao, Z. Jin-hua, and C. liang jun. 2019. Multi-Objective Particle Swarm Optimization Algorithm Based on Enhanced -Dominance. 1–5. https://doi.org/10.1109/ICEIS.2006.1703200

[69] M. A. Hassan, M. Xiao, Q. Wei, and S. Chen. 2015. "Help Your Mobile Applications with Fog Computing". In *IEEE SECON Workshops*. 1–6.

[70] J.H. Holland. 1992. *"Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence"*. MIT Press, Cambridge, MA, USA.

[71] H. J. Hong, P. H. Tsai, A. C. Cheng, M. Y. S. Uddin, N. Venkatasubramanian, and C. H. Hsu. 2017. "Supporting Internet-of-Things Analytics in a Fog Computing Platform". In *IEEE CloudCom*. 138–145. https://doi.org/10.1109/CloudCom.2017.45

[72] H. J. Hong, P. H. Tsai, and C. H. Hsu. 2016. "Dynamic Module Deployment in a Fog Computing Platform". In *APNOMS*. 1–6.

[73] K. Hong, D. Lillethun, U. Ramachandran, B. Ottenwälder, and B. Koldehofe. 2013. "Mobile Fog: A Programming Model for Large-scale Applications on the Internet of Things". In *ACM MCC*. ACM, New York, NY, USA, 15–20.

[74] P. Hu, S. Dhelim, H. Ning, and T. Qiu. 2017. "Survey on Fog Computing: Architecture, Key Technologies, Applications and Open Issues". *Journal of Network and Computer Applications* 98 (2017), 27 – 42. https://doi.org/10.1016/j.jnca.2017.09.002

[75] X. Huang, S. Ganapathy, and T. Wolf. 2009. "Evaluating Algorithms for Composable Service Placement in Computer Networks". In *IEEE International Conference on Communications*. 1–6. https://doi.org/10.1109/ICC.2009.5199007

[76] K. Hwang and M. Chen. 2017. *"Big-Data Analytics for Cloud, IoT and Cognitive Computing"* (1st ed.). Wiley Publishing.

[77] K. Intharawijitr, K. Iida, and H. Koga. 2016. "Analysis of Fog Model Considering Computing and Communication Latency in 5G Cellular Networks". In *IEEE PerCom Workshops*. 1–4. https://doi.org/10.1109/PERCOMW.2016.7457059

[78] G. C. Jana and S. Banerjee. 2017. "Enhancement of QoS for Fog Computing Model Aspect of Robust Resource Management". In *ICICICT*. 1462–1466. https://doi.org/10.1109/ICICICT1.2017.8342785

[79] J. Jijin, B. C. Seet, P. H. J. Chong, and H. Jarrah. 2017. "Service Load Balancing in Fog-Based 5G Radio Access Networks". In *IEEE PIMRC*. 1–5.

[80] J. Kennedy and R.C. Eberhart. 1995. "Particle Swarm Optimization". In *IEEE International Conference on Neural Networks*. 1942–1948.

[81] V. Kochar and A. Sarkar. 2016. "Real Time Resource Allocation on a Dynamic Two Level Symbiotic Fog Architecture". In *ISED*. 49–55. https://doi.org/10.1109/ISED.2016.7977053

[82] K.p.saharan and Anuj Kumar. 2015. "Fog in Comparison to Cloud: A Survey". *International Journal of Computer Applications* 122, 3 (July 2015), 10–12. Full text available.

[83] Y.-K. Kwok and I. Ahmad. 1999. "Static Scheduling Algorithms for Allocating Directed Task Graphs to Multiprocessors". *ACM Comput. Surv.* 31, 4 (Dec. 1999), 406–471. https://doi.org/10.1145/344588.344618

[84] K. U. R. Laghari, N. Crespi, B. Molina, and C. E. Palau. 2011. "QoE Aware Service Delivery in Distributed Environment". In *IEEE AINA*. 837–842. https://doi.org/10.1109/WAINA.2011.58

[85] G. T. Lakshmanan, Y. Li, and R. Strom. 2008. "Placement Strategies for Internet-Scale Data Stream Systems". *IEEE Internet Computing* 12, 6 (Nov 2008), 50–60.

[86] G. Lee, W. Saad, and M. Bennis. 2017. "An Online Optimization Framework for Distributed Fog Network Formation with Minimal Latency". *CoRR* abs/1710.05239 (2017).

[87] G. Lee, W. Saad, and M. Bennis. 2017. "An Online Secretary Framework for Fog Network Formation with Minimal Latency". In *IEEE ICC*. 1–6.

[88] I. Lera, C. Guerrero, and C. Juiz. 2018. "Comparing Centrality Indices for Network Usage Optimization of Data Placement Policies in Fog Devices". In *FMEC*. 115–122. https://doi.org/10.1109/FMEC.2018.8364053

[89] I. Lera, C. Guerrero, and C. Juiz. 2019. Availability-Aware Service Placement Policy in Fog Computing Based on Graph Partitions. *IEEE Internet of Things Journal* 6, 2 (April 2019), 3641–3651. https://doi.org/10.1109/JIOT.2018.2889511

[90] I. Lera, C. Guerrero, and C. Juiz. 2019. YAFS: A simulator for IoT scenarios in fog computing. *CoRR* abs/1902.01091 (2019). arXiv:1902.01091 http://arxiv.org/abs/1902.01091

[91] C. Li, Y. Xue, J. Wang, W. Zhang, and T. Li. 2018. "Edge-Oriented Computing Paradigms: A Survey on Architecture Design and System Management". *ACM Comput. Surv.* 51, 2, Article 39 (April 2018), 34 pages. https://doi.org/10.1145/3154815

[92] F. Li, M. Voegler, M. Claessens, and S. Dustdar. 2013. Efficient and Scalable IoT Service Delivery on Cloud. In *IEEE CLOUD*. 740–747. https://doi.org/10.1109/CLOUD.2013.64

[93] K. Li and J. Nabrzyski. 2017. "Traffic-Aware Virtual Machine Placement in Cloudlet Mesh with Adaptive Bandwidth". In *IEEE CloudCom*. 49–56. https://doi.org/10.1109/CloudCom.2017.47

[94] M.M.E. Mahmoud, J.J.P.C. Rodrigues, K. Saleem, J. Al-Muhtadi, N. Kumar, and V. Korotaev. 2018. "Towards Energy-Aware Fog-Enabled Cloud of Things for Healthcare". *Computers & Electrical Engineering* 67 (2018), 58 – 69. https://doi.org/10.1016/j.compeleceng.2018.02.047

[95] R. Mahmud, R. Kotagiri, and R. Buyya. 2018. *"Fog Computing: A Taxonomy, Survey and Future Directions"*. Springer Singapore, Singapore, 103–130. https://doi.org/10.1007/978-981-10-5861-5_5

[96] R. Mahmud, K. Ramamohanarao, and R. Buyya. 2018. "Quality of Experience (QoE)-Aware Placement of Applications in Fog Computing Environments ". *ACM Transactions on Internet Technology* (2018), –.

[97] R. Mahmud, S.N. Srirama, K. Ramamohanarao, and R. Buyya. 2018. "Latency-aware Application Module Management for Fog Computing Environments". *J. Parallel and Distrib. Comput.* (2018), –.

[98] V. De Maio and I. Brandic. 2018. "First Hop Mobile Offloading of DAG Computations". In *IEEE/ACM CCGRID*. 83–92. https://doi.org/10.1109/CCGRID.2018.00023

[99] R.T. Marler and J.S. Arora. 2004. "Survey of Multi-Objective Optimization Methods for Engineering". *Structural and Multidisciplinary Optimization* 26, 6 (01 Apr 2004), 369–395. https://doi.org/10.1007/s00158-003-0368-6

[100] R. Mayer, H. Gupta, E. Saurez, and U. Ramachandran. 2017. "FogStore: Toward a Distributed Data Store for Fog Computing". *CoRR* abs/1709.07558 (2017).

[101] A. Mebrek, L. Merghem-Boulahia, and M. Esseghir. 2017. "Efficient Green Solution for a Balanced Energy Consumption and Delay in the IoT-Fog-Cloud Computing". In *IEEE NCA*. 1–4.

[102] SQ. T. Minh, D. T. Nguyen, A. Van Le, H. D. Nguyen, and A. Truong. 2017. "Toward Service Placement on Fog Computing Landscape". In *4th NAFOSTED Conference on Information and Computer Science*. 291–296.

[103] N. Mohan, P. Zhou, K. Govindaraj, and J. Kangasharju. 2017. "Managing Data in Computational Edge Clouds". In *MECOMM*. ACM, New York, NY, USA, 19–24. https://doi.org/10.1145/3098208.3098212

[104] D. Monderer and L. Shapley. 1996. "Potential Games". *Games and Economic Behavior* 14, 1 (1996), 124–143. https://EconPapers.repec.org/RePEc:eee:gamebe:v:14:y:1996:i:1:p:124-143

[105] C. Mouradian, D. Naboulsi, S. Yangui, R. H. Glitho, M. J. Morrow, and P. A. Polakos. 2018. "A Comprehensive Survey on Fog Computing: State-of-the-Art and Research Challenges". *IEEE Communications Surveys Tutorials* 20, 1 (Firstquarter 2018), 416–464.

[106] M. Mukherjee, L. Shu, and D. Wang. 2018. "Survey of Fog Computing: Fundamental, Network Applications, and Research Challenges". *IEEE Communications Surveys Tutorials* (2018), 1–1. https://doi.org/10.1109/COMST.2018.2814571

[107] M.I. Naas, L. Lemarchand, J. Boukhobza, and P. Raipin. 2018. "A Graph Partitioning-based Heuristic for Runtime IoT Data Placement Strategies in a Fog Infrastructure". In *SAC*. ACM, New York, NY, USA, 767–774. https://doi.org/10.1145/3167132.3167217

[108] M.I. Naas, P. Raipin Parvedy, J. Boukhobza, and L. Lemarchand. 2017. "iFogStor: An IoT Data Placement Strategy for Fog Infrastructure". In *IEEE ICFEC*. 97–104.

[109] Shubha Brata Nath, Harshit Gupta, Sandip Chakraborty, and Soumya K. Ghosh. 2018. A Survey of Fog Computing and Communication: Current Researches and Future Directions. *CoRR* abs/1804.04365 (2018). arXiv:1804.04365 http://arxiv.org/abs/1804.04365

[110] A. Nazari, D. Thiruvady, A. Aleti, and I. Moser. 2016. "A Mixed Integer Linear Programming Model for Reliability Optimisation in the Component Deployment Problem". *JORS* 67, 8 (01 Aug 2016), 1050–1060. https://doi.org/10.1057/jors.2015.119

[111] M.J. Neely. 2010. *"Stochastic Network Optimization with Application to Communication and Queueing Systems"*. Morgan and Claypool Publishers.

[112] L. Ni, J. Zhang, C. Jiang, C. Yan, and K. Yu. 2017. Resource Allocation Strategy in Fog Computing Based on Priced Timed Petri Nets. *IEEE Internet of Things Journal* 4, 5 (Oct 2017), 1216–1228. https://doi.org/10.1109/JIOT.2017.2709814

[113] T. Nishio, R. Shinkuma, T. Takahashi, and N.B. Mandayam. 2013. "Service-Oriented Heterogeneous Resource Sharing for Optimizing Service Latency in Mobile Cloud". In *MobileCloud*. ACM, New York, NY, USA, 19–26. http://doi.acm.org/10.1145/2492348.2492354

[114] B. Ottenwälder, B. Koldehofe, K. Rothermel, and U. Ramachandran. 2013. "MigCEP: Operator Migration for Mobility Driven Distributed Complex Event Processing". In *ACM DEBS*. ACM, New York, NY, USA, 183–194. https://doi.org/10.1145/2488222.2488265

[115] T. Ouyang, Z. Zhou, and X. Chen. 2018. "Follow Me at the Edge: Mobility-Aware Dynamic Service Placement for Mobile Edge Computing". *IEEE Journal on Selected Areas in Communications* (2018), 1–1. https://doi.org/10.1109/JSAC.2018.2869954

[116] S. S. N. Perala, I. Galanis, and I. Anagnostopoulos. 2018. "Fog Computing and Efficient Resource Management in the era of Internet-of-Video Things (IoVT)". In *IEEE ISCAS*. 1–5. https://doi.org/10.1109/ISCAS.2018.8351341

[117] C. Perera, Y. Qin, J.C. Estrella, S. Reiff-Marganiec, and A.V. Vasilakos. 2017. "Fog Computing for Sustainable Smart Cities: A Survey". *CoRR* abs/1703.07079 (2017). http://arxiv.org/abs/1703.07079

[118] X.-Q. Pham and E.-N. Huh. 2016. "Towards Task Scheduling in a Cloud-Fog Computing System". In *APNOMS*. 1–4. https://doi.org/10.1109/APNOMS.2016.7737240

[119] P. Pietzuch, J. Ledlie, J. Shneidman, M. Roussopoulos, M. Welsh, and M. Seltzer. 2006. "Network-Aware Operator Placement for Stream-Processing Systems". In *ICDE*. 49–49. https://doi.org/10.1109/ICDE.2006.105

[120] C. Puliafito, E. Mingozzi, and G. Anastasi. 2017. "Fog Computing for the Internet of Mobile Things: Issues and Challenges". In *IEEE SMARTCOMP*. 1–6. https://doi.org/10.1109/SMARTCOMP.2017.7947010

[121] M.L. Puterman. 1994. *"Markov Decision Processes: Discrete Stochastic Dynamic Programming"* (1st ed.). John Wiley & Sons, Inc., New York, NY, USA.

[122] J. Santos, T. Wauters, B. Volckaert, and F. De Turck. 2017. "Resource Provisioning for IoT Application Services in Smart Cities". In *CNSM*. 1–9. https://doi.org/10.23919/CNSM.2017.8255974

[123] S. Sarkar, S. Chatterjee, and S. Misra. 2018. "Assessment of the Suitability of Fog Computing in the Context of Internet of Things". *IEEE Transactions on Cloud Computing* 6, 1 (Jan 2018), 46–59.

[124] S. Sarkar and S. Misra. 2016. "Theoretical Modelling of Fog Computing: A Green Computing Paradigm to Support IoT Applications". *IET Networks* 5, 2 (2016), 23–29.

[125] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies. 2009. "The Case for VM-Based Cloudlets in Mobile Computing". *IEEE Pervasive Computing* 8, 4 (Oct 2009), 14–23. https://doi.org/10.1109/MPRV.2009.82

[126] E. Saurez, K. Hong, D. Lillethun, U. Ramachandran, and B. Ottenwälder. 2016. "Incremental Deployment and Migration of Geo-distributed Situation Awareness Applications in the Fog". In *ACM DEBS*. ACM, New York, NY, USA, 258–269.

[127] V. Scoca, A. Aral, I. Brandic, R. De Nicola, and R. Brundo Uriarte. 2018. "Scheduling Latency-Sensitive Applications in Edge Computing". In *CLOSER*.

[128] M. Selimi, D. Vega, F. Freitag, and L. Veiga. 2016. "Towards Network-Aware Service Placement in Community Network Micro-Clouds". In *Euro-Par - Volume 9833*. Springer-Verlag New York, Inc., New York, NY, USA, 376–388. https://doi.org/10.1007/978-3-319-43659-3_28

[129] H. Shah-Mansouri and V.W.S. Wong. 2017. "Hierarchical Fog-Cloud Computing for IoT Systems: A Computation Offloading Game". *CoRR* abs/1710.06089 (2017). arXiv:1710.06089 http://arxiv.org/abs/1710.06089

[130] A. Shrivastwa, S. Sarat, K. Jackson, C. Bunch, E. Sigler, and T. Campbell. 2016. *OpenStack: Building a Cloud Environment*. Packt Publishing.

[131] M. M. Shurman and M. K. Aljarah. 2017. "Collaborative Execution of Distributed Mobile and IoT Applications Running at the Edge". In *ICECTA*. 1–5. https://doi.org/10.1109/ICECTA.2017.8252057

[132] A. Singh, N. Auluck, O. Rana, A. Jones, and S. Nepal. 2017. "RT-SANE: Real Time Security Aware Scheduling on the Network Edge". In *UCC*. ACM, New York, NY, USA, 131–140.

[133] O. Skarlat, M. Nardelli, S. Schulte, M. Borkowski, and P. Leitner. 2017. "Optimized IoT Service Placement in the Fog". *Service Oriented Computing and Applications* 11, 4 (01 Dec 2017), 427–443.

[134] O. Skarlat, M. Nardelli, S. Schulte, and S. Dustdar. 2017. "Towards QoS-Aware Fog Service Placement". In *ICFEC*. IEEE Computer Society, 89–96.

[135] O. Skarlat, S. Schulte, M. Borkowski, and P. Leitner. 2016. "Resource Provisioning for IoT Services in the Fog". In *IEEE SOCA*. 32–39.

[136] V.B. Souza, X. Masip-Bruin, E. ópez, J. Garcia, G.J. Ren, A. Jukan, and A. Juan Ferrer. 2018. "Towards a proper service placement in combined Fog-to-Cloud (F2C) architectures". *Future Generation Computer Systems* 87 (2018), 1 – 15. https://doi.org/10.1016/j.future.2018.04.042

[137] V. B. C. Souza, W. Ramírez, X. Masip-Bruin, E. Marín-Tordera, G. Ren, and G. Tashakor. 2016. "Handling Service Allocation in Combined Fog-Cloud Scenarios". In *IEEE ICC*. 1–5.

[138] I. Stojmenovic. 2014. "Fog Computing: A Cloud to the Ground Support for Smart Things and Machine-to-Machine Networks". In *ATNAC*. 117–122.

[139] H. Tan, Z. Han, X. Y. Li, and F. C. M. Lau. 2017. "Online Job Dispatching and Scheduling in Edge-Clouds". In *IEEE INFOCOM*. 1–9. https://doi.org/10.1109/INFOCOM.2017.8057116

[140] M. Taneja and A. Davy. 2017. "Resource Aware Placement of IoT Application Modules in Fog-Cloud Computing Paradigm". In *IFIP/IEEE IM*. 1222–1228.

[141] R. I. Tinini, L. C. M. Reis, D. M. Batista, G. B. Figueiredo, M. Tornatore, and B. Mukherjee. 2017. "Optimal Placement of Virtualized BBU Processing in Hybrid Cloud-Fog RAN over TWDM-PON". In *GLOBECOM*. 1–6.

[142] K. Toczé and S. Nadjm-Tehrani. 2018. "A Taxonomy for Management and Optimization of Multiple Resources in Edge Computing". *CoRR* abs/1801.05610 (2018). arXiv:1801.05610 http://arxiv.org/abs/1801.05610

[143] L. Tong, Y. Li, and W. Gao. 2016. "A Hierarchical Edge Cloud Architecture for Mobile Computing". In *IEEE INFOCOM*. 1–9. https://doi.org/10.1109/INFOCOM.2016.7524340

[144] W. Tärneberg, A. Mehta, E. Wadbro, J. Tordsson, J. Eker, M. Kihl, and E. Elmroth. 2017. "Dynamic Application Placement in the Mobile Cloud Network". *Future Generation Computer Systems* 70 (2017), 163 – 177.

[145] R. Urgaonkar, S. Wang, T. He, M. Zafer, K. Chan, and K.K. Leung. 2015. "Dynamic Service Migration and Workload Scheduling in Edge-Clouds". *Performance Evaluation* 91 (2015), 205 – 228. Special Issue: Performance 2015.

[146] L. M. Vaquero and L. Rodero-Merino. 2014. "Finding Your Way in the Fog: Towards a Comprehensive Definition of Fog Computing". *SIGCOMM Comput. Commun. Rev.* 44, 5 (Oct. 2014), 27–32. https://doi.org/10.1145/2677046.2677052

[147] K. Velasquez, D.P. Abreu, M. Curado, and E. Monteiro. 2017. "Service Placement for Latency Reduction in the Internet of Things". *Annals of Telecommunications* 72, 1 (01 Feb 2017), 105–115. https://doi.org/10.1007/s12243-016-0524-9

[148] S. Venticinque and A. Amato. 2018. "A Methodology for Deployment of IoT Application in Fog". *Journal of Ambient Intelligence and Humanized Computing* (06 Apr 2018). https://doi.org/10.1007/s12652-018-0785-4

[149] J. Wang, J. Pan, and F. Esposito. 2017. "Elastic Urban Video Surveillance System Using Edge Computing". In *SmartIoT*. ACM, New York, NY, USA, Article 7, 6 pages. http://doi.acm.org/10.1145/3132479.3132490

[150] P. Wang, S. Liu, F. Ye, and X. Chen. 2018. "A Fog-Based Architecture and Programming Model for IoT Applications in the Smart Grid". *CoRR* abs/1804.01239 (2018). arXiv:1804.01239 http://arxiv.org/abs/1804.01239

[151] S. Wang, M. Zafer, and K. K. Leung. 2017. "Online Placement of Multi-Component Applications in Edge Computing Environments". *IEEE Access* 5 (2017), 2514–2533. https://doi.org/10.1109/ACCESS.2017.2665971

[152] Z. Wen, R. Yang, P. Garraghan, T. Lin, J. Xu, and M. Rovatsos. 2017. "Fog Orchestration for Internet of Things Services". *IEEE Internet Computing* 21, 2 (Mar 2017), 16–24. https://doi.org/10.1109/MIC.2017.36

[153] D. Whitley. 1994. "A Genetic Algorithm Tutorial". *Statistics and Computing* 4, 2 (01 Jun 1994), 65–85. https://doi.org/10.1007/BF00175354

[154] Y. Xia, X. Etchevers, L. Letondeur, T. Coupaye, and F. Desprez. 2018. "Combining Hardware Nodes and Software Components Ordering-based Heuristics for Optimizing the Placement of Distributed IoT Applications in the Fog". In *SAC*. ACM, New York, NY, USA, 751–760. https://doi.org/10.1145/3167132.3167215

[155] M. Yannuzzi, R. Milito, R. Serral-Gracia, D. Montero, and M. Nemirovsky. 2014. "Key Ingredients in an IoT Recipe: Fog Computing, Cloud computing, and More Fog Computing". In *IEEE CAMAD*. 325–329. https://doi.org/10.1109/CAMAD.2014.7033259

[156] D. Ye, M. Wu, S. Tang, and R. Yu. 2016. "Scalable Fog Computing with Service Offloading in Bus Networks". In *IEEE CSCloud*. 247–251.

[157] S. Yi, Z. Hao, Q. Zhang, Q. Zhang, W. Shi, and Q. Li. 2017. "LAVEA: Latency-aware Video Analytics on Edge Computing Platform". In *ACM/IEEE SEC*. ACM, New York, NY, USA, Article 15, 13 pages. https://doi.org/10.1145/3132211.3134459

[158] S. Yi, C. Li, and Q. Li. 2015. "A Survey of Fog Computing: Concepts, Applications and Issues". In *Mobidata*. ACM, New York, NY, USA, 37–42. http://doi.acm.org/10.1145/2757384.2757397

[159] S. Yi, Z. Qin, and Q. Li. 2015. Security and Privacy Issues of Fog Computing: A Survey. In *"Wireless Algorithms, Systems, and Applications"*, Kuai Xu and Haojin Zhu (Eds.). Springer International Publishing, Cham, 685–695.

[160] A. Yousefpour, C. Fung, T. Nguyen, K. Kadiyala, F. Jalali, A. Niakanlahiji, J. Kong, and J.P. Jue. 2018. "All One Needs to Know about Fog Computing and Related Edge Computing Paradigms: A Complete Survey". *CoRR* abs/1808.05283 (2018).

[161] A. Yousefpour, G. Ishigaki, R. Gour, and J. P. Jue. 2018. "On Reducing IoT Service Delay via Fog Offloading". *IEEE Internet of Things Journal* 5, 2 (April 2018), 998–1010. https://doi.org/10.1109/JIOT.2017.2788802

[162] A. Yousefpour, G. Ishigaki, and J. P. Jue. 2017. "Fog Computing: Towards Minimizing Delay in the Internet of Things". In *IEEE EDGE*. 17–24. https://doi.org/10.1109/IEEE.EDGE.2017.12

[163] A. Yousefpour, A. Patil, G. Ishigaki, J.P. Jue, I. Kim, X. Wang, H.C. Cankaya, Q. Zhang, and W. Xie. 2017. "QoS-aware Dynamic Fog Service Provisioning".

[164] R. Yu, G. Xue, and X. Zhang. 2018. "Application Provisioning in Fog Computing-enabled Internet-of-Things: A Network Perspective". In *IEEE INFOCOM*.

[165] J. K. Zao, T. T. Gan, C. K. You, S. J. R. MÃ©ndez, C. E. Chung, Y. T. Wang, T. Mullen, and T. P. Jung. 2014. "Augmented Brain Computer Interaction Based on Fog Computing and Linked Data". In *International Conference on Intelligent Environments*. 374–377. https://doi.org/10.1109/IE.2014.54

[166] D. Zeng, L. Gu, S. Guo, Z. Cheng, and S. Yu. 2016. "Joint Optimization of Task Scheduling and Image Placement in Fog Computing Supported Software-Defined Embedded System". *IEEE Trans. Comput.* 65, 12 (Dec 2016), 3702–3712.

[167] L. Zhao, J. Liu, Y. Shi, W. Sun, and H. Guo. 2017. "Optimal Placement of Virtual Machines in Mobile Edge Computing". In *GLOBECOM*. 1–6. https://doi.org/10.1109/GLOCOM.2017.8254084

[168] H. Zhu and C. Huang. 2017. "Availability-Aware Mobile Edge Application Placement in 5G Networks". In *GLOBECOM*. 1–6. https://doi.org/10.1109/GLOCOM.2017.8254591