A RESOURCE MANAGEMENT SURVEY FOR MISSION-CRITICAL AND TIME-CRITICAL APPLICATIONS IN MULTIACCESS EDGE COMPUTING

Nina Santi[1] and Nathalie Mitton[2]
[1]Inria, France, nina.santi@inria.fr, [2]Inria, France, nathalie.mitton@inria.fr

NOTE: Corresponding author: Nina Santi, nina.santi@inria.fr

Abstract – *Multiaccess Edge Computing (MEC) brings additional computing power in proximity of mobile users, reducing latency, saving energy and alleviating the network's bandwidth. This proximity is beneficial, especially for mission-critical applications where each second matters, such as disaster management or military operations. Moreover, it enables MEC resources embedded on mobile units like drones or robots that are flexible to be deployed for mission-critical applications. However, the MEC servers are capacity-limited and thus need an acute management of their resources. The mobile resources also need a smart deployment scheme to deliver their services efficiently. In this survey, we review mission-critical applications, resource allocation and deployment of mobile resources techniques in the context of the MEC. First, we introduce the technical specifics and uses of MEC in mission-critical applications to highlight their needs and requirements. Then, we discuss the resource allocation schemes for MEC and assess their fit depending on the application needs. In the same fashion, we finally review the deployment of MEC mobile resources. We believe this work could serve as a helping hand to design efficient MEC resource management schemes that respond to challenging environments such as mission-critical applications.*

Keywords – Disaster management, multiaccess computing, resource allocation, resource deployment, unmanned aerial vehicles

1. INTRODUCTION

Mission-critical applications require particular attention as they may imply life or important assets losses, which entails entail tremendous consequences in their failure. We can consider disaster management applications, where time is a precious resource: the first 72 hours, the *golden relief time*, is particularly critical to locate and res- cue people [1]. Military applications are also mission- critical as they defend citizens from external threats and defend the country. IoT technology combined with cloud computing has the potential to assist rescuers and agents to gain every precious second, by gathering information, analyzing the situation and providing support services. It will also help agents organize and coordinate their op- eration to handle the situation [2]. Cloud computing retains many advantages, like reduced costs [3] and is easily scalable [4]. However, cloud computing is constrained by its remote location from end users, leading to high latency and delays. This problem is increased by the heavy data generation from IoT devices that burden the network and possibly creating bottlenecks when tasks are not processed rapidly enough [5]. In addition, mission-critical applications operate in challenging environments with a damaged or scarce network, making the cloud difficult to reach. Hence, cloud computing may not fit all the mission-critical applications' challenges and requirements.

In recent years, a new trend has arisen, moving cloud computing capacities to the edge of the network. This paradigm is called edge computing where connected devices send their tasks to computing nodes located at the edge of the network, i.e., next to users or things producing data [5]. This proximity provides advantages over the cloud, namely: *i)* latency reduction [5, 6, 7] *ii)* energy saving [5, 6], *iii)* augmented privacy [6, 7] and *iv)* location and context awareness [5, 6]. These benefits represent the key to carrying out the strong requirements of real-time applications. It is especially the case for mission-critical and time-critical applications where time is an important resource [7]. Different edge computing paradigms exist, each more or less specific. Fog computing and edge com- puting both bring cloud services to the edge of the net- work, hence can be confused [8]. However, edge com- puting focuses more on things while fog computing focuses on the overall infrastructure from the edge to the cloud [5, 8]. Cloudlets are "data center in a box" close to users and accessible by WiFi. They take example of WiFi access point but with computing capacities to deliver cloud services close to users with little maintenance and low power [9]. However, they have been discarded because of their WiFi access that implies limited coverage, difficult mobility support between cloudlets and security concerns. Similar to cloudlets, micro-data-centers consist of 10 servers or less and are placed next to users [10]. Finally, Mobile Edge Computing (MEC) is defined by ETSI in 2015 as edge computing incorporated in Radio Access Networks (RANs) to serve mobile users [11] as shown in Figure 1. The term mobile edge computing has evolved to multiaccess computing [12], allowing heterogeneous Ra- dio Access Technologies (RATs), like 5G, LTE, Wi-Fi and so on, in the paradigm and so broadening its use cases. In this paper, we therefore employ MEC as multiaccess edge computing as a generic

term that also includes the mobile edge computing paradigm. MEC is the most promising candidate for mission-critical and time-critical applications, because of its proximity, good mobility support for mobile users and integration of multiple access technologies. In MEC networks, local servers are limited in resources and as a recent paradigm, it undertakes open challenges to manage these limited-capacity resources [5, 13]. Thus, MEC resources need to be properly managed to handle efficiently the users' requests. The resource management in MEC is divided in three aspects : *i),* offloading decision, *ii),* resource allocation and *iii),* users mobility management, i.e service migration.

In mission-critical scenarios, edge resources may be embarked on mobile units, such as drones and vehicles. Indeed, communication networks are often damaged by disaster or are nonexistent due to remote location [14]. Thus, drones and vehicles have the necessary mobility to be deployed rapidly in emergency areas, temporarily and are flexible enough to move to follow the demands' dynamic (which occurs when users are mobile or in situations where demands are highly dynamic in a single device) [15, 16, 17]. The resource management is then enlarged with a fourth aspect : *iv)* mobile resource management which includes the deployment of the resources, i.e., their number and location, their path planning and new costs such as deployment delays. In this survey, we start by presenting related surveys in Section 2. We then present two main use cases of mission-critical applications that may use edge computing in Section 3. As MEC is a recent field, we also include other edge com- puting paradigms like fog computing or cloudlets. We then review in Section 4 resource allocation methods for MEC, not only for mobile resources, as again there is not enough work on it. Finally, we review resource deployment schemes for MEC in Section 5 and provide open challenges and future research direction in Section 6. With that survey, we aim to provide tools and insight for the design of robust MEC resource management schemes that are befitting for real life hard-constrained use cases. To the best of our knowledge, it is the first survey that provides a review about MEC resource management through the scope of mission-critical applications.

2. RELATED WORKS

Several surveys about MEC exist in the literature. They are either general [6, 18, 19, 20] or focus on different aspects and methods [21, 22, 23]. Mao et al. [6] introduce MEC with the modelling of MEC communication and computation, mobile devices and edge server. They then review and classify resource management, and finally identify open research directions. Mach and Becvar [20] present a thorough survey about MEC offloading, resource allocation, user mobility management and its architecture. They highlight what to take into account when designing MEC computation offloading schemes and discuss previous work. Abbas et al. [18] provide a definition of MEC and its application. They also provide insight of MEC related research and technologies. Vhora and Gandhi [19] introduce a review on MEC architecture, related research and challenges, tools for simulation and finally MEC applications. Peng et al. [21] review service adoption and provision for MEC. They consider MEC service adoption, i.e task offloading, from the mobile users' perspective and MEC service provision, i.e resource allocation and server placement, from the edge server side. Wang et al. [22] review service migration in MEC that they define and compare with previous existing concepts. They discuss the state-of-art methods and technical service hosting solutions. Zamzam et al. [23] propose a resource management survey using machine learning methods. They organize the research by goals and classify machine learning methods. There also are surveys about public safety and mission-critical wireless network solutions [24, 25, 26, 27] or technology solutions [15, 2, 28]. Baldini et al. [24] survey public safety organization use cases, requirements and their wireless communications standard. Jarwan et al. [25] provide design requirements, architecture solutions and standards for public safety networks based on LTE. These works also provide a testing and evaluation framework for such networks using Network Simulator NS-3. Yu et al. [27] describe the layered architecture of public safety communication. Then they review communication technologies for device-to-device communications and dynamic wireless networks. They also discuss the integration of some technologies in public safety networks like 5G and edge computing. Kumbhar et al. [26] introduce public safety networks standards and challenges with a focus on LTE, Land Mobile Radio System (LMRS) and Software-Defined Radio (SDR). The white paper [15] reports technologies employed in public safety applications and highlights gaps and the technology that can fill them. The authors provide thorough use cases and their technologies' opportunities. Works [2, 28] study the application of IoT technologies for disaster management operations and future research directions. We can note that none of these surveys focus on MEC. This survey is complementary to these previous ones as it browses MEC resource management work and discusses them by their suitability depending on the applications, with a focus on mission-critical applications.

3. MISSION-CRITICAL APPLICATIONS USE CASES

MEC offer computing services independently from the Internet and at proximity to requesting users and devices [8]. This proximity allow new low-latency services such as object or speech recognition [10] or augmented/virtual reality [19, 11]. It also offers a new type of location-aware and context-aware services [6, 29]. Theses MEC services may be employed by mobile users but also by IoT devices [19, 11], such as security cameras [17, 30]. Mission-critical applications may profit from these

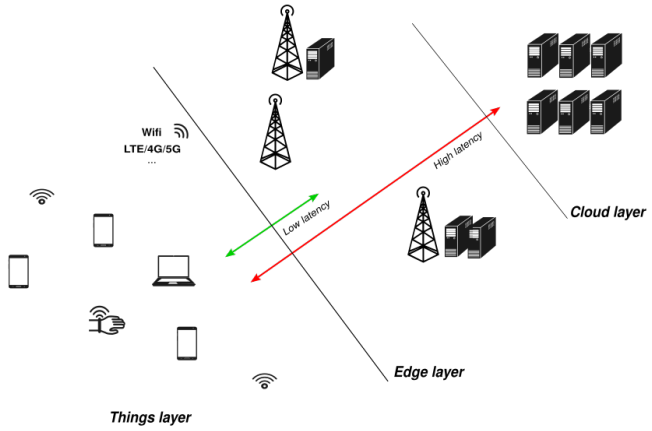

Fig. 1 – The multiacess edge computing paradigm

types of services to carry out their crucial objectives that involve life and properties. Thus, in the next section, we are going to review two main use cases, disaster management and military and what kind of edge services they may request. We will also review the technical specifics of these two uses cases.

3.1 Disaster management

With climate change, disaster occurences are bound to increase [31], having important social and economic impacts. The rescuers need to be prepared and supported efficiently to carry out their mission, saving many lives and recover from the situation. IoT is recognized as a relevant technology for providing useful support to rescue operations [2, 28]. But all the data produced by IoT needs low-latency processing to be useful for rescuers in real time. For this, MEC is a promising candidate due to its proximity [5] and its on-site rapidly deployable [32, 33] networking structure, even in difficult environments. In the next section, we review disaster management applications that use MEC but also other edge computing paradigms as currently there is little work only on mobile MEC.

3.1.1 Edge-enabled disaster management applications

Edge services enable low-latency applications that provide situation awareness to rescuers. These types of applications deliver important information about the situation, helping rescuers adapt and organize their mission. A common use of edge computing by disaster management applications is for video and image analytics [17, 16, 34, 17, 35, 30, 36, 37] as edge computing responds to the short response time requirements of these applications. It helps discover victims' locations, count and state and provide facial recognition for missing persons. It also helps analyzing the environment to detect dangerous paths clogged with fire or hazardous chemicals and paths obstructed by wreckage.

Wu. et al. found that video analytics done in the edge instead of on the cloud reduces the latency up to 61% using 4G [17], expressing the usefulness of edge computing in these situations. The analysis can be done entirely in the edge servers, but as they are capacity limited it may be done only partially, the rest being handled in the cloud. Some previous work leverages the edge computing power to filter images taken from a disaster context and send only relevant ones to the cloud for advanced processing. Indeed, several images and video are taken by smartphones [16], drones [34] or cameras [17], and by sending them all the cloud will burden the network, as they do not all contain useful information. The filtering will save precious bandwidth and act in real time without human intervention. COCO, proposed by Zhao et al. [35], is a MEC-based adaptive image sensor that uploads to the cloud only images with specific content. Liu et al. [16] propose Echo that is an edge-based face recognition frame- work, that also filters images and preprocesses them before sending them to the cloud. Chemodanov et al. [30] propose geospatial video analytics that analyze images from many devices in a broad area to deliver information to rescuers. They employ fog computing to preprocess images and manage human-computer interactions, that are trivial tasks with low latency expectations. edge computing allow other applications, like localization and path finding for autonomous agents (drones and boats) when searching for victims in the sea [36]. Avgeris et al. [37] propose SMOKE, a three-layered cyber-physical social system to detect forest fires and assist public authorities. They use the edge layer to process images captured by IoT nodes to detect fire at an early stage. They also propose the horizontal and vertical scaling of the edge resource to adjust the QoS. It can monitor rescuers' health in mission and alert about their state [17]. Finally, artificial intelligence which runs at the edge provides an action plan and helps decision-making [17].

3.1.2 Technical specificity

Architecture Several architectures are "three-layered" where the first layer is composed of field sensors, then the second edge layer and the third one that is the cloud. The cloud is kept to store historic data [17] and runs less time-sensitive or heavy tasks [35, 16, 30]. Edge servers are often in mobile units near the disaster scene like fire vehicles, public buses [38] or drones [36]. It highlights the need for edge resources mobility management to support the rescuers as they move on the field. Sensor devices that gather data are heterogeneous. Wireless wearable sensors monitor health or help for localization. Body cameras [17] and smartphones [35, 16, 30], from civilians or rescuers, capture the environment to analyze paths, recognize missing people or help evaluate their state and injuries. Surveillance cameras [30, 17] are also used to evaluate the environment on a larger scale.

Finally, autonomous agents, like drones, unnamed agent boats or robots, are able to go where humans cannot and cover rapidly an area to find missing people or assess the situation [36, 34]. Figure 2 represents an example of a disaster management architecture.

Network specifications In edge computing, there are two main channels of communication: on one part the communication between end devices and the edge, the other part is the connection between the edge and the cloud. The regular network can be damaged rendering the connection between the edge and the cloud disrupted or unstable [16, 34]. For the connection between the edge and end devices, 4G and 5G are the most common network access employed, especially with civil smart- phones [16] or drones [34, 36]. With the expansion of smart cities, the public WiFi hotspot is also a candidate. However, it is noticeable that 4G seems to induce less la- tency than WiFi in the case of video analytics [17]. Also, end devices and sensors may establish an ad hoc network to communicate together and with edge computing devices, without pre-existing structure [17, 30]. The satellite network is an other option especially when the re-gular network does not work, however the latency can be problematic [16].

3.2 Military

IoT for military is restricted because of unstable networks, limited bandwidth, power-limited devices and a highly dynamic environment [39]. Edge computing offers the low-latency and mobility required in the battlefield [40].

3.2.1 Edge-enabled military applications

The battlefield has a vast variety of heterogeneous sensors and devices that generate a lot of heterogeneous data [41, 42]. To ease the instability of the network, edge computing has the power to declutter all this information by filtering, preprocessing and add meaning to the mass of data. Singh et al. [40] introduce an edge-based system to monitor soldiers' health, weapons and location of those in command, the other soldiers and themselves. In addi- tion to filter, their framework merges and attaches mea-ning to data to bring situational awareness to agents on the field and those in command. Wang et al. [41] leverage fog computing to compute and store the mass of data near the field and so provide real-time responses. Moreover, they use it to filter and preprocess data to reduce information sent to the cloud, sparing bandwidth. They found that the latency is reduced to about 85% when 300 tasks are requested. Castiglione et al. [42] use edge computing to authenticate agents with their biometrics data when they access sensitive material like weapons or vehicles, in addition to monitoring their health. Lewis et al. [43] propose tactical cloudlets to compute intensive tasks, like video and audio recognition and also filter useless data to lighten the application.

3.2.2 Technical specificity

There are numerous sensors on the battlefield that can be on the ground sensors or wearable [42]. Sensors are worn by soldiers for health monitoring. They can also be on weapons to monitor their status [40]. Like disaster management application, drones [41] or robots [44] may be used. In battlefield health monitoring, the wireless devices worn by soldiers form a Body Area Network (BAN) [40]. Devices communicate with each other and with the edge with the LoWPAN wireless network. These devices send raw data to edge networking devices which transmit it to the semantic fog where data is processed meaningfully. Architecture of the combat cloud-fog consists of the combat resource, fog computing and cloud computing [41]. Combat resource is combat units which collect data and execute physical action, like radars or drones. They can communicate together. Networking devices near the field perform the fog computing. The computing tasks are distributed among them since networking devices have their own duty and low capacities.
Mission-critical applications have strong requirements and distinct specificities. We have seen they employ heterogeneous end devices, sensors, vehicles and autonomous agents, generating many data. All this data, the unstable network and the strong latency requirement make MEC a promising solution to deliver effective support to agents. Moreover, we have seen diverse network access used in these applications, which is consistent with the variety of end devices. By integrating heterogeneous network access, MEC is all the more consistent in this type of application. Finally, some applications employ vehicles to transport the edge server [16, 17], which stresses their high dynamic and need for resource mobility management.

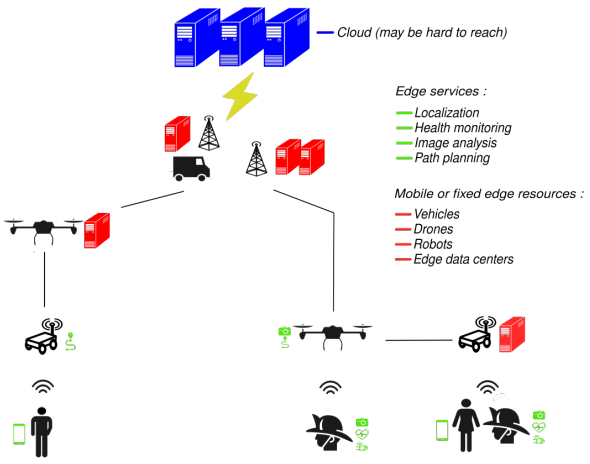

Fig. 2 – An example of an architecture of a MEC disaster management application

4. RESOURCE ALLOCATION

MEC resources are capacity limited, unlike the cloud, and can even work on batteries. The resource allocation scheme is so vital to manage these limited resources

Table 1 – Review of architectures, goals and resources used in resource allocation for MEC

Architecture			Cloud	Ref	Goals			Allocated resources			
					Energy	Latency	Other	Computing resources	Communication resources	Task placement	Other
Two Edge servers	One user	Two APs	x	[45]		x		x	x		
One Edge server	Two fixed users	One AP		[46]	x						Offloading time
				[47]	x			x	x		
	Multi-users			[48]	x			x			
				[49]	x	x		x	x		
			x	[50]		x		x	x		
				[51]	x			x	x		
				[52]		x		x	x		
				[53]	x	x		x			
				[54]	x	x		x			
				[55]		x	Revenue	x	x		
		Multi-APs		[56]		x		x			
				[57]	x				x		
				[58]	x		Costs	x	x		
				[59]	x	x		x	x		
Multi-Edge servers			x	[60]	x					x	
			x	[61]		x		x	x		
				[62]	x	x					Offloading time
				[63]			Min timeout probability	x			
				[64]	x		Reliability	x	x		
			x	[65]		x		x		x	
				[66]		x	Costs	x			
				[67]		x		x	x		
				[68]	x	x		x		x	
				[69]		x		x			
			x	[70]	x	x		x	x		
				[71]	x			x	x		
			x	[72]	x	x		x	x		
				[68]	x	x		x		x	
				[73]		x			x	x	
				[74]	x	x		x	x		
				[75]	x			x			
Multi-UAVs				[76]	x			x			
				[77]			Nb of served tasks	x	x		
One UAV	Multi-devices (fixed)			[78]	x						Bit offloading

Table 2.1 – Review of methods used in resource allocation for MEC

Methods	Ref	Main constrains				Offloading decision
		Communication capacities	Computing capacities	Devices battery	Tasks deadlines	
Cauchy-Schwarz inequality, Linear programming	[61]	x	x			x
KKT conditions, Sub-gradients method	[52]	x				
ADMM decomposition, Convex optimization	[55]	x	x			x
Regularization technique, Convex optimization	[66]		x			
Lagrange duality method, KKT conditions, Convex optimization	[47]	x	x			x
Dinkelbach, Lagrange duality and Sub-gradients methods	[46]			x		x
Majorization-minimization method	[57]	x				x
Benders decomposition	[60]				x	
Convex optimization, Heuristic algorithm	[62]	x	x			
Decomposition and iteration algorithm	[65]		x		x	
Genetic algorithm	[58]	x	x		x	x
Decomposition and iteration algorithm	[49]	x	x			x
Successive convex approximation, Matching theory	[74]		x		x	x
Decomposition and iteration algorithm based on genetic algorithms	[75]	x	x			x
Cauchy-Schwarz inequality, Convex optimization	[79]	x	x			
Decomposition and iteration algorithm	[76]		x			
Successive convex approximation, Decomposition and iteration algorithm	[78]		x	x		x
Decomposition and iteration algorithm	[68]		x	x		x
Reinforcement learning	[70]			x	x	
	[53]		x		x	x
	[69]	x	x			
	[56]				x	
	[73]	x				
	[51]	x	x		x	x
Decomposition and iteration algorithm	[59]			x	x	x
Deep Neural Network	[48]		x		x	x
Game theory	[67]	x	x			

Table 2.2 – Review of methods used in resource allocation for MEC (following)

Methods	Ref	Main constrains				Offloading decision
		Communication capacities	Computing capacities	Devices battery	Tasks deadlines	
Dynamic Programming algorithm	[54]		x		x	
Decomposition and iteration algorithm	[71]	x	x		x	x
Convex optimization and Heuristic	[50]	x	x	x	x	x
Heuristic	[72]	x	x		x	x
SCA-based iteration algorithm	[77]	x	x		x	x
Lyapunov theory	[63]		x			x
Interior-point (IPA) alogrithm	[45]	x	x			

efficiently and respond to users' requests. In this section, we first show in Section 4.1 the different aspects to take into account for modelling the requesting devices' tasks. Then in Section 4.2 we present the different types of MEC resources we can allocate, be it computing or communica- tion resources. We then discuss in Section 4.3 the diffe-rent goals pursued by resource allocation methods. We finally review in Section 4.4 the different methods that have their own trade-off between accuracy and speed. We discuss these methods according to the system scale and requirements.

The reviewed papers in this section are summarized in tables 1, 2.1 and 2.2. Table 1 summarizes architectures, goals and resources considered in each scheme. For the architecture, reviewed papers consider the number of edge servers, the number of users, the number of Access Points (APs) and may integrate the cloud into the system. Their goals are mainly to reduce the energy consumption of the system or the latency. There are also other goals like reducing the different costs of the network, ensuring its reliability or maximizing the tasks coverage, e.g., the number of served tasks.

The allocated resources are mainly computing resources, with or without communication resources, and the tasks placement, e.g., on which the server will process a task. Communication resources are rarely allocated alone. Tables 2.1 and 2.2 summarize which methods the reviewed papers employ. We discuss these methods in Section 4.4. Depending on the goals and the system, reviewed papers take into consideration different constraints, such as communication capacities of MEC servers, the devices' battery and so on. Finally, each reviewed paper may resolve along the resource allocation the offloading decision, that is the decision to process tasks locally on the devices or remotely on servers.

4.1 Task modelling

The system modelling is crucial in an efficient resource allocation scheme. It allows the algorithm to consider the system critical aspects and deliver adapted results. An important aspect of system modelling is the modelling of the tasks. A task is commonly modelled as (S, C, L), where S is the associated data (input data and code), C the required CPU cycles to achieve the task and L the task deadline, i.e., the task maximum tolerant latency [70, 68, 80]. S and C can be deduced through code profiling [81, 58, 62]. Some works do not include the task deadline in their modelling [74, 62]. But this parameter is central in latency sensitive applications for obvious reasons. It will allow the scheme to respect task deadlines, prioritize tasks with close deadlines and may employ it to drop some obsolete tasks that burden buffers. Also, work can assume dividing tasks to allocate parts with different resources to accelerate processing. However, if parts of the tasks have strong dependencies, this may burden the network as a resource needs to wait for the others to process its part. Some work does not consider dividing to simplify the scheme [58]. The task generation speed [79] or distribution [82] have impact on the workload over the network, and may help prevent bottlenecks.

4.2 Resources to allocate

Diverse types of resources can be allocated to users, mainly communication resource and computational resources. Moreover, we can consider them jointly, leading to more efficient schemes.

Communication resource Usually, the main communication resource allocated is the bandwidth. It is allocated to devices with a percentage of the total spectrum bandwidth available [80, 67] or the amount of radio bandwidth [70]. It can also depend on the channel access method considered in the system. For sys-

tems using TDMA, some work allocates time in each time slots for each device proportionally to the data they need to offload [79, 52]. For OFDMA, subcarriers are allocated [57, 83]. The promising Non-Orthogonal Multiple Access (NOMA) method, suitable for 5G, allows sharing subcarriers between multiple users instead to at most one, like in OFDMA. So if the system uses NOMA, the resource allocation scheme had to assign the subcarriers to multiple users [71, 50]. Some work also model communication resources abstractly to be applied on different types of systems. A point to consider when we allocate communication resources is the interference. The intra-cell interference is usually ignored to sub-channels assignment [49, 62]. Inter-cell interference makes the resource allocation more complex as it adds dependence between users' uploading rates [62]. Some work ignores this inter-cell interference as they postulate that cells are far enough from each other or have orthogonal bandwidth allocation [49]. However, it can be interesting to consider it for networks where these interferences are highly probable, like ultra-dense MEC networks. In addition, as MEC may possess heterogeneous Radio Access Technologies (RATs), it raises the interesting problem of choosing the right RAT to serve a device at a given time for a given task. For instance, Hsu et al. [72] consider the licensed 5G and the unlicensed NR-U. Indeed, each RAT has its own characteristic, like coverage, mobility support, data rate and so on. All of this may influence the delay, the energy consumption and the quality of service. They can also incur additional costs, like 5G instead of the generally free Wi-Fi. Finally, next generation emergency [84] and public safety [85] communications are challenges to incorporate in MEC resources allocations.

Computational resource Instead of the cloud, MEC systems have limited computational resources. Thus they are critical resources we need to allocate efficiently. It is even more the case with many mobile users or extremely limited edge resources, as it is often the case in mission-critical applications. If the computing resources are badly allocated, important devices' tasks may be unprocessed on time and the overall system can be congested. Computational resources may be CPU cycles per seconds [55, 80, 62] or CPU cores [56]. For UAVs-based MEC, some work allocates the number of offloaded bits to the UAV [86, 87]. However, recent work considers allocating CPU frequencies instead because it seems to reduce energy consumption further [78, 88, 89]. To reduce the latency further and help MEC servers, some works also consider the use of some spare computing power on certain devices that can use it to assist other devices. The devices with enough resources communicate directly with requesting devices, called device-to-device [90] or machine-to-machine [71] communication.

Joint communication and computation The communication and computation allocations affect each other. Regardless of how much a task is given a certain amount of communication resource, if it does not have enough computing resource, the task will not be processed more rapidly, and conversely [55]. So these two resources are involved in QoS requirements, such as delay and energy consumption, and jointly allocating them lead to more efficient results [62].

Server selection The server selection for a task can also be considered as a resource allocation. Matching MEC nodes with tasks is relevant because MEC nodes may possess heterogeneous capacities, in terms of quantity as well of quality, and fit more or less a task need [65, 70]. Plus, there is a trade-off to consider between computational time and network delay, depending on servers' workloads, their distance from devices and their channel quality. For example, it may be worthier to assign a task to a farther server but which is less busy [65].

4.3 Goals

The goals of the resource allocation scheme depend on the use cases and the applications considered by the work. They can be more adapted to latency sensitive applications by minimizing tasks' completion time, or fit MEC systems with battery-powered devices by minimizing their energy consumption. The goal can be tuned with weight in the objective function. It can aim to prioritize some devices [62] or some aspect of a multi-objective problem [49], like giving more weight to energy consumption rather than latency.

4.3.1 Energy-aware parameters

In mobile edge computing networks, mobile devices are battery powered. Thus minimizing their energy consumption is crucial to maintaining the user's quality of experience [62] and preserve autonomous devices' battery to let them complete their tasks. Some work considers the overall energy consumption, e.g., from the local computing to the offload computing [57]. Other work considers only the device's energy consumption as is it assumed that MEC servers have reliable power sources [47, 49]. However, in mission-critical applications, servers can be battery powered, like embedded on UAVs or in buses. They may have more energy at their disposal than end devices; nonetheless, their energy budget is limited. Moreover, the energy consumption of the overall system is always important to minimize the application's energy impact. The energy consumption for a task is often calculated as $E = \kappa F^{\beta} \cdot c$, where F is the computing capacities of the device as CPU cycles per seconds, c represent the number of CPU cycles required to finish the task, κ and β are constant that depends on the device's chip architecture [54, 62]. κ is often $10^{-}26$ or $10^{-}27$ and β is 2 [59, 70, 53, 49]. So the computing capacities influence the task's process time but also the energy consumption. Therefore, there is a trade-off between energy consumption and execution time to consider. This trade-off can be adjusted with a weight factor in the optimization goal to fit the application needs, having a low energy consumption or a reduced latency [91, 59]. It can

additionally include external factors like the device's residual battery [59] and so adjust to the devices' needs in real time. When MEC servers are UAVs, the hover time is to be included in the energy model under the form: $Eh = P \cdot T$, where P is the power to hover and T the hovering time [92]. Besides the hovering time, UAV consumes energy for flying, depending on its velocity and weight [86]. Its accelerations have equally significant impact on energy [78]. We can ignore some energy consumption points in the optimization whether they are idle energy and we cannot control it. This is the case for server idle energy consumption or energy consumption of links that are traffic independent [60, 58]. Plus, some actions are negligible in comparison to others in the system, like downloading energy consumption [53].

4.3.2 Latency

Latency is crucial in mission-critical applications where situations may be life or death, like in search and rescue. A task's latency comprises the processing time and necessary transmission time from the device to the edge and potentially to the cloud [79, 68, 54, 65]. Work [52] add to it the compression time, present in system with heavy tasks like video processing. We can also add the local or remote computational queuing delay [55, 64] because of the continue tasks generation, present even when other tasks are processed. The processing time depends on CPU cycles required to complete the task and the computing capacities, e.g., CPU cycles/seconds, allocated to the task [74, 69]. The latency is equally affected by the data generation speed. When the generation is superior to the system processing capacities, data accumulates in buffers and nodes don't process tasks in real time. Wang et al. [61] refer to it as a blocking state and propose to adapt the resource allocation scheme depending on whether the system is in a blocking state or in a nonblocking state. Furthermore, the data generation is usually non-uniform across the system. It leads vary- ing workloads between servers, and some may be over- loaded while others are free from tasks. It is so interest- ing to consider balancing in the resource allocation [69], as well as the trade-off between computing and transmis- sion time when moving a task to a less loaded but further node [65]. In addition, some devices can process critical tasks or occupy a pivotal role in the system, therefore they need priority in their processing. A solution proposed in [52] is minimizing a weighted-sum delay of all devices, the weight reflecting devices' importance in the system. Alternatively, [51] proposes to measure each tasks' pri- ority with delay and reliability requirements. Standardly, the downloading time from server to devices is ignored, since results data are smaller and downlinks have higher rates [62, 58]. What's more, the transmission time between a base station and its associated MEC server is ignored [59]. Finally, as seen in Section 4.1, the partition of tasks can greatly reduce the processing time by paralleling the processing. [45] shows that the dynamic placement of the tasks partitioning decision, i.e the decision to process on which nodes each part of the task, can reduce the latency. Indeed, if the decision is taken on the requesting node, it can take its much constrained resources. But if the decision is taken on a remote MEC server, it may take more time to reach other MEC servers, depending on their placement from the device.

4.3.3 Reliability

As seen in Section 3, some tasks of mission-critical application are of vital importance. Thus, the MEC must have a certain level of reliability to ensure that these tasks are processed. In wireless networks, the reliability is seen as the probability to successfully transfer data within a delay [93]. A first challenge in MEC networks is node failure. The redundancy of tasks is a relevant solution to mitigate this effect [94, 95]. However, it can burden the network if the redundancy takes more than the needed computing or communication resources. A node failure measurement helps ensure the minimum tasks' reliability, avoiding the resources' overuse [77]. Another challenge is extreme events in server and UE processing queues. When queues are overloaded they may drop some critical tasks, and assuring an average queuing delay is not sufficient to prevent that [96]. Thus, the work [64] uses the statistics of the extreme queue length to ensure reliability.

4.4 Methods

The chosen method for resource allocation has to propose a satisfactory compromise between precision, computational complexity and scalability depending on the problem and its context. Some methods may be unable to solve a problem [56] or fill the system requirements. In addition, the method has to fit the scale of the system, not being too complex for large-scale systems, and its needs, for example if suboptimal results are sufficient.

4.4.1 Optimization methods

Classic mathematical optimization methods aims to solve problems optimally. Cao et al. [47] solve optimally the resource allocation in a three-node network to minimize devices' energy consumption with the Lagrange duality method. Chen et al. [68] propose a scheme for resource allocation and task placement in ultra-dense networks for minimizing the task completion time. They resolve the computational resource allocation part of the problem with Karush–Kuhn–Tucker (KKT) conditions. Ren et al. [52] exploit the KKT conditions to allocate a MEC server's resources to users while minimizing the delay, where data is compressed locally by the user before sending. Even though classic mathematical optimization methods allow optimal outcomes they come with significant complexity. Thus they are adapted to small-scale systems with few parameters. They are unadapted to large-scale systems where the complexity is too high to handle and they will either not be able to solve the problem or demand an unfeasible amount of time.

4.4.2 Decomposition techniques

Decomposition techniques are used for challenging problems where optimal solutions are likely nonexistent. They decompose the initial problem into sub-problems, easier to tackle. They are in addition employed instead of optimal solutions to reduce complexity. A reduced complexity is important in MEC systems where resources are limited unlike in the cloud. They can be a good trade-off between efficiency and results, especially with systems where medium accuracy is sufficient. Plus, we may implement them easier and in a distributed manner. **Iterative algorithms** When considering several joint problems, we can decouple the sub-problems and solve them individually, like decoupling the offloading decision and the resource allocation. To retain the connection be- tween the sub-problems, we solve them in an iterative algorithm. Each iteration takes the output of the previous iteration in input to update the solution until convergence to the optimal solution. It allows a decreased complexity but at the cost of the solution's precision. In iterative algorithms, we have to pay attention to its convergence properties and its required iterations. Li et al. [75] propose a two-stage heuristic resolving iteratively the offloading decision and the CPU frequency allocation with the goal of minimizing the energy consumption of mobile devices. Pham et al. [49] propose the JOBCA iterative algorithm to solve the resource allocation and offloading problem for wireless back-haul networks. Networks with wireless back-haul may be utilized for rural areas or emergency services where wired back-haul is expensive and restrictive. Li et al. [58] introduce an offloading and resource allocation scheme for multiple wireless access points to minimize the monetary and energy costs. Tran and Pompili [62] propose a resource allocation scheme for multiservers in ultra-dense networks to minimize a weighted sum of task completion time with devices' energy consumption. For that they introduce an iterative heuristic algorithm to solve the initial problem in polynomial time. Fan and Ansari [65] address the workload allocation for cloudlets, considering the trade-off between sending tasks to a near cloudlet but overloaded or a far cloudlet but less busy. To simplify the initial problem, they propose an iterative algorithm solving task assignment and computing resource allocation. Zhang et al. [59] aim at finding the trade-off between latency and energy consumption. They investigate a scenario with one small cell and another with multiple small cells and propose an iterative search algorithm for the multiple cell scenarios. Zhu et al. [71] introduce a resource allocation scheme for 5G Industrial Internet of Thing (IIoT). In this scheme, they include devices with enough computing resource to help other devices with machine-to-machine communication. **Mathematical decomposition** Mathematical solutions exist to transform the problem into simpler sub-problems. Then, we can solve them with classic optimization methods. Ji and Guo [46] propose a resource allocation for two users, one far and one close to the MEC server. In the relay mode, where the nearest user serves as relay between the MEC server and the far user, they employ the Dinkelbach's method to transform the non-convex problem into a convex one. Next, they solve it with classical convex optimization methods. Wang et al. [55] propose a resource allocation strategy with a two-stage tandem queues for maximizing the revenue of the network. The first queue is for packet transmission through the base station and the second for computational processing at the MEC server. The initial NP-Hard problem is decomposed with an Alternating Direction Method of Multipliers (ADMM)-based algorithm into convex sub-problems. ADMM is an algorithm to solve problem with a splittable objective function. It is adapted to decentralized systems because of its decomposability and requires a few iterations to converge for modest accuracy [97]. However, it is slow to converge for high accuracy. Yang et al. [60] handle the task allocation problem for cloudlets with Bender decomposition to minimize the overall energy consumption. Zhang et al. [80] use a modified generalized Benders decomposition for latency-sensitive services with caching to minimize the overall latency. They also solve the problem with a branch and bound method that has an exponential computation complexity. Wang et al. [66] introduce MOERA, an online resource allocation algorithm to minimize arbitrary operational costs and costs that reduce quality of service (e.g., delay) and consider user's mobility without their prior knowledge. They use a regularization technique [98] to divide into sub-problems. Wang et al. [61] consider MEC systems having a nonblocking state and a blocking state when too much data has accumulated in a server's buffers. For the nonblocking state, they divide the problems into task assignment and resource allocation sub-problems with the Cauchy–Schwarz inequality. For the blocking state they aim to recover the nonblocking state rapidly by equalizing the transmitting and computing resource across layers. Lyu et al. [54] address task admission and resource allocation by minimizing the energy consumption with their EROS scheme. The initial problem is simplified to an integer programming problem by pre-admitting tasks that have to be offloaded to meet their deadlines. Then it is resolved with a quantified dynamic programming algorithm. Haber et al. [77] provide a resource allocation scheme for UAV-assisted MEC, taking into account the UAV positioning and reliability.

4.4.3 Game theory

Game theory methods are adapted to systems where each node has individuals' interests. For example, when there is a service provider aiming at maximizing its revenue and autonomous devices, each wanting to complete their tasks as quickly as possible. Theses methods can propose a consensus in such systems in a decentralized manner.

Josilo and Dán [67] provide a resource allocation model where edge services providers and devices interact as a Stackelberg game. The devices are the leaders and want to minimize their tasks completion time by choosing to which edge server they offload their tasks and through which access point. Sardellitti et al. [74] use matching theory to assign users to a MEC server and their communication and computational resources, according to the users' preferences.

4.4.4 Learning methods

Learning methods learn from the past and/or from the environment. They are more rapid than classic methods, but can be less precise. Each one possesses its own advantages or inconvenient.

Evolutionary Computation (EA) EA is inspired by biology. Many algorithms exist under EA and are more or less adapted to certain problems with their own pros and cons. For example, genetic algorithms tend to not be trapped in local optima [99, 100] while being hard tuning it to problems. Thus, Wan et al. [100] propose a different use of EA for task-driven resource assignment, including hybridization of different EA algorithms. Li et al. [99] use a genetic algorithm to minimize completion time for mo- bile devices and an edge server.

Reinforcement learning Allocation resource schemes can use a reinforcement learning method. More specifically a Q-learning method can be used. It has for advantage to be model-free and adapt itself to a stochastic environment. It is so a solution for dynamic context, that we retrieve in mission-critical MEC scenarios [70]. Also, we can tune it to take more or less long-term decisions. Wang et al. [73] propose a multi-stack reinforcement learning algorithm for resource allocation in mobile edge computing. They use multi-stack to take advantage of a historical resource allocation scheme and avoid learning the same scheme. However, a disadvantage of reinforcement learning is the Q table. It will be excessively large for large-scale systems due to many different possible states, rendering its storage and the Q value search complex [53, 69]. Alternatively, we can use a deep reinforcement learning method, with a deep neural network to estimate the Q value for an action and a state. But we lose the "model-free" properties of the Q-learning, and need to train a model. Chen et al. [70] propose a deep reinforcement learning for CoMEC network, where collaborative edge servers are connected. Li et al. [53] use deep reinforcement learning for allocating computational resources of a MEC server to mobile devices by minimizing execution delay and energy consumption. Wang et al. [69] introduce a deep reinforcement learning based resource allocation algorithm to minimize the computing and routing delay in edge networks. They also consider balancing the resource allocation to reduce local- ized pressure on the network and improve delays. Yang et al. [56] propose a deep reinforcement learning agent for the trade-off between downlink data reliability and delay by CPU allocation and data blocklength in ultra-reliable low latency communication networks. The Q-learning method is suitable when there is not much communication or interaction with other agents in the system, i.e., in MEC environment the mobile devices. However, if we assume that the mobile devices interact and are intelligent agents, Q-learning lacks an adaption mechanism to the other agents' (mobile devices) actions. Feng et al. [51] employ a WoLF-PHC reinforcement learning for resource allocation to reduce energy consumption and prioritize tasks in mission-critical applications. The WoLF-PHC algorithm adapts the learning rate by learning slower when we "take the ascendant" to let the other agent the time to adapt its strategy and reach a whole system equilibrium. Conversely, the learning rate will be faster when the other agent takes the ascendant to "catch them up" [101].

Deep neural network Li and and Lv [48] use a Deep Neural Network (DNN) for resource allocation to minimize the network energy consumption. They train DNNs to simulate the behavior of a sequential quadratic programming algorithm. They train a DNN with a fixed number of devices in the data set and the other with a random number of devices, rendering the latter one more flexible than the specialized one. Thus, the DNN will take less time to solve the problem with an approximation of the optimal result. However, the environment is highly dynamic and leads various uncertainties. A training set might be under-representative of the complex system and the trained DNN is not flexible enough to tackle some situations as it does not adapt on run [69]. Moreover, it can be difficult to find good data beforehand.

5. MOBILE RESOURCE DEPLOYMENT

When MEC servers are mounted on UAV or robots, they are suited to cover the needs of mobile users in temporary events or emergency responses. Indeed, fixed resources might instead be too costly, too inflexible to deploy or just needed for a limited time. Particularly in emergency responses and post-disaster management, deploying temporary additional computing resources can help rescuers, victims and wireless devices processing critical tasks with critical delay constraints. However, mobile edge resource deployment comes with many challenges.

For deploying one mobile resource, we have to optimize its trajectory between a starting and an ending point to serve the mobile devices by minimizing the delay [104] or the system energy consumption [83, 78, 86]. For deploying multiple mobile resources, we need to optimize their numbers, i.e minimizing their number while satisfying the goal, their locations and associate them with mobile users. Indeed, with multiple mobile resources, we do not have a starting and ending point so we cannot plan the entire trajectory but rather compute the next location point. Goals can be minimizing energy consumption [92], minimizing number of deployed nodes [82] or balance the workload between resources [108]. Also, the deployment scheme is often joint with another problem-

Table 3 – Review of architectures and goals used in mobile resource deployment for MEC

Architecture		Goals	Ref	Deployment type		
				Number of UAVs to deploy	UAVs locations	UAVs trajectory
One UAV	Fixed nodes	Min energy	[83]			x
			[78]			x
	Mobile users		[87]			x
			[102]			x
			[86]			x
			[103]			x
		Min Delay	[104]			x
		Max Computation Rate	[105]			x
		Max Offloaded Bits	[106]			x
Multi-UAVs	Fixed nodes	Max Coverage	[107]			x
		Load balancing	[108]		x	
		Min UAV number	[82]	x	x	
	Mobile users	Min Energy	[92]	x		
			[109]		x	
		Min Delay and Energy	[88]		x	
		Max Offloaded Tasks	[110]		x	
		Max Served Tasks	[111]		x	
		Max Computation Efficiency	[89]			x
		Nb of served resquests	[77]		x	

atic: tasks scheduling, offloading decision, CPU optimization, i.e what amount of CPU a task needs, resource or bits allocation. We summarize reviewed papers for this part in tables 3 and 4, classifying them depending on their main objectives and undertaken constraints.

5.1 System modelling

One UAV deployment Generally, when we consider one UAV deployment, we assume that it starts and finishes its trajectory at predefined locations. Like that, the UAV does cycles in which devices can offload their tasks [102, 83, 78, 87, 102, 106]. The problem is then to study the path planning in theses cycles. The cycle is separated in time slots, where the UAV is considered static, as well as devices when they are mobile [83, 78, 87]. In general, in these system the area covered is not large [86], and thus these works are convenient for short term deployment and low-scale applications or to help fixed servers in short areas. We name this deployment type as trajectory in table 3.

Multi-UAVs deployment Deployment of multiple UAVs can cover large areas and be used in large-scale applications. It is a complex challenge that is highly coupled with the resource allocation scheme as they depend on each other. Previous research considers different scenario for multiple UAVs deployment. [107] and [109] assume a three layer MEC system, with a device layer, a UAV layer and fixed ground MEC servers. Islambouli and Sharafeddine [82] study UAVs swarm deployment with some UAVs acting as relays for multi-hop offloading when the transmission power is too low [82]. Some other works consider UAVs deployment jointly with other problems like tasks scheduling [92], user association and resource allocation [89].

5.2 Deployment methods

Like resource allocation, the deployment of UAV is often associated with a joint problem. In these cases, the problem will be too complex to address directly. Thus, works tend to decompose the initial problem into sub-problems and resolve them iteratively, where the deployment part is resolved with the results of the previously resolved joint problems [89, 102, 78]. In the next subsections, when authors employ iterative algorithms, we will focus on the deployment part.

5.2.1 Convex optimization

Convex optimization allows finding an optimal solution to a relatively simple problem. It can be sufficient in a problem with one UAV, but not for a more complex problem, like with multiple UAVs. Xiong et al. [87] use the CVX solver to solve a UAV trajectory along with offloading and bits allocation. Li et al. [86] also propose a convex function solvable by a CVX solver in their two-stage alternating algorithm for UAV trajectory and bits allocation.

5.2.2 Successive Convex Optimization (SCA method

Non-convex optimization problems are frequent in UAVs-enabled MEC due to lots of constraints and parameters. Thus, the Successive Convex Optimization (SCA) method

Table 4 – Review of methods used in mobile resource deployment for MEC

Ref	Method	Joint problematic							Main constraints
		Bits allocation	Scheduling	Offloading	Power allocation	CPU optimi-zation	Resource allocation	User association	
[83]	Lagrangian duality, Successive convex approximation	x	x		x				Power and computation capacities
[78]	Successive convex approximation, Decomposition and iterative algorithm	x			x	x			Energy devices consumption's, Trajectory constraints
[87]	Decomposition and iterative algorithm	x		x					Task deadlines, Energy budget
[102]	Dinkelbach method, Successive convex approximation, Decomposition and iterative algorithm						x		Computing capacities, Mechanical constraints
[86]	Decomposition and iterative algorithm	x							Energy budget, Data Causality
[104]	Penalty dual-decomposition method		x	x					Residual battery, Energy budget
[105]	Decomposition and iterative algorithm						x		UAV speed
[109]	Decomposition and iterative algorithm			x					Devices coverage
[88]	Successive convex approximation						x		Computing capacities
[103]	Block successive upper-bound minimization algorithm			x			x		Latency constraints, Computation and power capacities
[110]	Greedy algorithm							x	Computation capacity
[111]	Greedy dispatching algorithm								Communication range, UAVs number
[89]	Decomposition and iterative algorithm						x	x	UAV Velocity, Obstacles
[106]	Decomposition and iterative algorithm				x			x	UAV battery and velocity
[107]	Deep Reinforcement Learning		x				x		-
[108]	Differential Evolution algorithm		x						Computation capacities
[92]			x	x			x		Computation capacities
[82]	Meta-Heuristic			x					Computation capacities, Time and power constraints
[77]	SCA-based iterative algorithm			x			x		UAV battery, Tasks reliabiltiy, Latency requirements

resolve these problems by approximating them into convex problems iteratively [112]. This method will produce a local optimal solution in a parallel and distributed manner. Some work employs the SCA method to resolve UAV trajectory [105, 83, 89, 78] and UAV position [88, 77] problems. However the resulting optimizer can have a high computational complexity and does not respond to the real-time requirement of the system [78].

5.2.3 Greedy algorithms

Greedy algorithms are known heuristics solutions for coverage problems [113], such as in UAV deployment. They propose a good estimation of the global optimal solution to complex problems. Chen et al. [110] use a greedy algorithm to deploy UAVs to locations and associate their devices' tasks to maximize offloaded tasks. Wang et al. [111] use a greedy algorithm to dispatch UAVs, considering users' hotspots, for maximizing the number of processed tasks.

5.2.4 Population-based meta-heuristics

Population-based meta-heuristics search for the best solutions in a set of candidate solutions. It starts with a random population of solutions, then merges, keeps or eliminates each one in each iteration to obtain the most suited. They have the advantage to avoid local optima [114] at the cost of a higher complexity than a classic optimization method. Thus, it can be hard to employ them for online solutions. Besides, each algorithm possesses its own advantage and inconvenience.

Evolutionary computation Wang et al. [92] use a Differential Evolution (DE) algorithm to decide UAV location. Their problem possesses a mixed decision variables and is a variable-length, posing problem to use efficiently a DE algorithm, so they propose a new encoding where each UAV in an individual and the population is a deployment solution. Yang et al. [108] also use a DE to deploy UAVs at a location to balance the workload among them to avoid bottleneck in the network.

Ions motion optimization Islambouli and Sharafeddine [82] use ions motion optimization [114] to choose the number of UAVs and their positions, along with device associations and computation allocations. The algorithm models the population of possible solutions that are anions and cations and choose an efficient solution iteratively. The work [114] shows that ions motions optimization tend to avoid local optimum and few tuning parameters, instead of other population-based algorithms.

6. OPEN ISSUES AND CHALLENGES

In this section, we discuss some still open issues and related challenges.

6.1 Real-time resource allocation and deployment

MEC application environments evolve and change rapidly, it is therefore essential to evaluate and predict diverse aspects of the application and network to respond appropriately. For the resource allocation scheme, the changing traffic load and channel conditions can hinder the network, creating bottlenecks and significantly impede the delay. For deployment of mobile resources, user mobility is important to take into account to position the resource at the proper place and taking account of the travel time to be sure they are available when needed. Many other aspects can impact the resource management scheme. Classic machine learning mechanisms may help predict these aspects, but they have to be meticulously modelled as historic data may not match the application due to the challenging and very fast changing environment.

6.2 Security and privacy

The security and privacy questions in MEC are sensitive because of the distributed and wireless nature of the paradigm. Also, in mission-critical applications, it is even more the case as the information can be sensitive and malicious attackers can take advantage of the situation or make it worse. The fixed and wearable sensors are prone to network attacks on their wireless communication. The attackers can jam the communication, rendering them unreliable or listen to the confidential data. The cloud is generally more secure than the other layers of MEC, but privacy is to be considered as we transmit sensitive data to the Internet. MEC needs proper security and privacy mechanisms to be reliable in sensitive situations.

6.3 Green MEC

Several pieces of work focus on reducing the energy consumption of the devices, as it is important to preserve their battery. However, they may not consider the energy consumption on the overall application, i.e., the energy consumption of the edge and cloud. It is indispensable to consider it globally to achieve green MEC, therefore minimizing pollution and reduce costs. Further, it is even more the case with mission-critical applications where resources may be on mobile units and so battery-constrained.

6.4 MEC experimentation and test beds

The majority of the reviewed work validate their work by simulation. Although there are good simulation tools, experiments are valuable to assess a scheme in real situations. The prevalence of simulations is undoubtedly due to the lack of tools, especially test beds for edge com- puting. The SILECS[1] platform proposes a large-scale distributed infrastructure from sensors to large data centers, thus making it a possible tool for MEC experimentation.

[1] https://www.silecs.net/

7. CONCLUSION

In this paper, we have reviewed mission-critical and time-critical applications using MEC. We present their architectures and the edge services they use. We then reviewed work on MEC resource allocation, highlighting their modelling, goals and methods and do the the same for mobile resource deployment schemes. We finish by providing some open challenges and research direction. With this work, we hope to assist the researcher designing MEC resource management schemes that fit highly dynamic applications, like mission-critical and time-critical applications, and fully leveraging MEC potential.

REFERENCES

[1] S. F. Ochoa and R. Santos. "Human-centric wireless sensor networks to improve information availability during urban search and rescue activ- ities". In: *Information Fusion* 22 (2015).

[2] A. Sinha, P. Kumar, N. P. Rana, R. Islam, and Y. K. Dwivedi. "Impact of internet of things (IoT) in disaster management: a task-technology fit perspective". In: *Annals of Operations Research* 283 (2019).

[3] A. Botta, W. de Donato, V. Persico, and A. Pescapé. "Integration of Cloud computing and Internet of Things: A survey". In: *Future Gener. Comput. Syst.* 56 (2016).

[4] P. M. Mell and T. Grance. "The NIST Definition of Cloud Computing". In: *NIST Special Publication 800-145*. 2011.

[5] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu. "Edge Com- puting: Vision and Challenges". In: *IEEE Internet of Things Journal* 3 (2016).

[6] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief. "A Survey on Mobile Edge Computing: The Communication Perspective". In: *IEEE Communi- cations Surveys Tutorials* 19 (2017).

[7] M. Satyanarayanan. "The Emergence of Edge Computing". In: *Computer* 50 (2017).

[8] A. Yousefpour, C. Fung, T. Nguyen, K. P. Kadiyala, F. Jalali, A. Niakanlahiji, J. Kong, and J. P. Jue. "All One Needs to Know about Fog Computing and Related Edge Computing Paradigms: A Complete Survey". In: *ArXiv* abs/1808.05283 (2019).

[9] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies. "The Case for VM-Based Cloudlets in Mobile Computing". In: *IEEE Pervasive Computing* 8 (2009).

[10] V. Bahl. *emergence of micro datacenter (cloudlets/edges) for mobile computing*. http://www.microsoft.com/en-us/research/wp-content/uploads/2016/11/Micro-Data-Centers-mDCs-for-Mobile-Computing-1.pdf. Mar. 13, 2015.

[11] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young. *Mobile Edge Computing - A key technology towards 5G*. Tech. rep. White Paper No. 11. ETSI (European Telecommunications Standards Institute), 2015.

[12] F. Giust, G. Verin, K. Antevski, J. Chou, Y. Fang, W. Featherstone, F. Fontes, D. Frydman, A. Li, A. Manzalini, D. Purkayastha, D. Sabella, C. Wehner, K-W Wen, and Z. Zhou. *MEC Deployments in 4G and Evolution Towards 5G*. Tech. rep. White Paper No. 24. ETSI (European Telecommunications Standards Institute), 2018.

[13] D. Dechouniotis, N. Athanasopoulos, A. Leivadeas, N. Mitton, R. Jungers, and S. Papavassiliou. "Edge Computing Resource Allocation for Dynamic Networks: The DRUID-NET Vision and Perspective". In: *Sensors (Basel, Switzerland)* 20 (2020).

[14] B.S. Manoj and A. H. Baker. "Communication Challenges in Emergency Response". In: *Commun. ACM* 50 (2007).

[15] M. Ulema, D. Zuckerman, P. Chatzimisios, F. Granelli, N. Mangra, E. Markakis, K. Namuduri, Y. Nikoloudakis, P. Rawat, M. Z. Shakir, and T. Zhang. *Public Safety Technology Gaps and Opportunities*. White Paper 1st ed. 2021.

[16] F. Liu, Y. Guo, Z. Cai, N. Xiao, and Z. Zhao. "Edge-Enabled Disaster Rescue: A Case Study of Searching for Missing People". In: *ACM Trans. Intell. Syst. Technol.* 10 (2019).

[17] X. Wu, R. Dunne, Q. Zhang, and W. Shi. "Edge Computing Enabled Smart Firefighting: Opportunities and Challenges". In: *Proceedings of the Fifth ACM/IEEE Workshop on Hot Topics in Web Systems and Technologies*. 2017.

[18] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie. "Mobile Edge Computing: A Survey". In: *IEEE Internet of Things Journal* 5 (2018).

[19] F. Vhora and J. Gandhi. "A Comprehensive Survey on Mobile Edge Computing: Challenges, Tools, Applications". In: *2020 Fourth International Conference on Computing Methodologies and Communication (ICCMC)*. 2020.

[20] P. Mach and Z. Becvar. "Mobile Edge Computing: A Survey on Architecture and Computation Offloading". In: *IEEE Communications Surveys Tutorials* 19 (2017).

[21] K. Peng, V. C. M. Leung, X. Xu, L. Zheng, J. Wang, and Q. Huang. "A Survey on Mobile Edge Computing: Focusing on Service Adoption and Provision". In: *Wireless Communications and Mobile Computing* 2018 (2018).

[22] S. Wang, J. Xu, N. Zhang, and Y. Liu. "A Survey on Service Migration in Mobile Edge Computing". In: *IEEE Access* 6 (2018).

[23] M. Zamzam, T. Elshabrawy, and M. Ashour. "Resource Management using Machine Learning in Mobile Edge Computing: A Survey". In: *2019 Ninth International Conference on Intelligent Computing and Information Systems (ICICIS)*. 2019.

[24] G. Baldini, S. Karanasios, D. Allen, and F. Vergari. "Survey of Wireless Communication Technologies for Public Safety". In: *IEEE Communications Surveys Tutorials* 16 (2014).

[25] A. Jarwan, A. Sabbah, M. Ibnkahla, and O. Issa. "LTE-Based Public Safety Networks: A Survey". In: *IEEE Communications Surveys Tutorials* 21 (2019).

[26] A. Kumbhar, F. Koohifar, İ. G venç, and B. Mueller. "A Survey on Legacy and Emerging Technologies for Public Safety Communications". In: *IEEE Com- munications Surveys Tutorials* 19 (2017).

[27] W. Yu, H. Xu, J. Nguyen, E. Blasch, A. Hematian, and W. Gao. "Survey of Public Safety Communications: User-Side and Network-Side Solutions and Future Directions". In: *IEEE Access* 6 (2018).

[28] P. P. Ray, M. Mukherjee, and L. Shu. "Internet of Things for Disaster Management: State-of-the-Art and Prospects". In: *IEEE Access* 5 (2017).

[29] S. Nunna, A. Kousaridas, M. Ibrahim, M. Dillinger, C. Thuemmler, H. Feussner, and A. Schneider. "Enabling Real-Time Context-Aware Collaboration through 5G and Mobile Edge Computing". In: *2015 12th International Conference on Information Technology - New Generations*. 2015.

[30] D. Chemodanov, P. Calyam, and K. Palaniappan. "Fog Computing to enable Geospatial Video Analytics for Disaster Situational Awareness". In: 2019.

[31] M. Berlemann and M. F. Steinhardt. "Climate Change, Natural Disasters, and Migration—a Survey of the Empirical Evidence". In: *CESifo Economic Studies* 63 (2017).

[32] W.-P. Chen, A.-H. Tsai, and C.-H. Tsai. "Smart Traffic Offloading with Mobile Edge Computing for Disaster-Resilient Communication Networks". In: *Journal of Network and Systems Management* 27 (2019).

[33] J. Xu, K. Ota, and M. Dong. "Big Data on the Fly: UAV-Mounted Mobile Edge Computing for Disaster Management". In: *IEEE Transactions on Network Science and Engineering* 7 (2020).

[34] J. Wang, Z. Feng, Z. Chen, S. George, M. Bala, P. Pillai, S. Yang, and M. Satyanarayanan. "Bandwidth-Efficient Live Video Analytics for Drones Via Edge Computing". In: *2018 IEEE/ACM Symposium on Edge Computing (SEC)*. 2018.

[35] Z. Zhao, F. Liu, Z. Cai, and N. Xiao. "Edge-Based Content-Aware Crowdsourcing Approach for Image Sensing in Disaster Environment". In: *Proceedings of the 14th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*. 2017.

[36] T. Yang, Z. Jiang, J. Dong, H. Feng, and C. Yang. "Multi Agents to Search and Rescue Based on Group Intelligent Algorithm and Edge Computing". In: *2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*. 2018.

[37] "Where There Is Fire There Is SMOKE: A Scalable Edge Computing Framework for Early Fire Detection". In: *Sensors* 19 (2019).

[38] M. Narang, W. Liu, J. Gutierrez, and L. Chiaraviglio. "A Cyber Physical Buses-and-Drones Mobile Edge Infrastructure for Large Scale Disaster Emergency Communications". In: *2017 IEEE 37th International Conference on Distributed Computing Systems Workshops (ICDCSW)*. 2017.

[39] N. Suri, M. Tortonesi, J. Michaelis, P. Budulas, G. Benincasa, S. Russell, C. Stefanelli, and R. Winkler. "Analyzing the applicability of Internet of Things to the battlefield environment". In: *2016 International Conference on Military Communications and Information Systems (ICMCIS)*. 2016.

[40] D. Singh, G. Tripathi, A. M. Alberti, and A. Jara. "Semantic edge computing and IoT architecture for military health services in battlefield". In: *2017 14th IEEE Annual Consumer Communications Networking Conference (CCNC)*. 2017.

[41] Y. Wang, Z. Ren, H. Zhang, X. Hou, and Y. Xiao. ""Combat Cloud-Fog" Network Architecture for Internet of Battlefield Things and Load Balan-cing Technology". In: *2018 IEEE International Conference on Smart Internet of Things (SmartIoT)*. 2018.

[42] A. Castiglione, K. R. Choo, M. Nappi, and S. Ricciardi. "Context Aware Ubiquitous Biometrics in Edge of Military Things". In: *IEEE Cloud Compu-ting* 4 (2017).

[43] G. Lewis, S. Echeverría, S. Simanta, B. Bradshaw, and J. Root. "Tactical Cloudlets: Moving Cloud Computing to the Edge". In: *2014 IEEE Military Communications Conference*. 2014.

[44] M. Ashokkumar and T. Thirumurugan. "Integrated IOT based design and Android operated Multi-purpose Field Surveillance Robot for Military Use". In: *Proceedings of the International Conference for Phoenixes on Emerging Current Trends in Engineering and Management (PECTEAM 2018)*. 2018.

[45] Z. Liu and Q. Zhang. "Adaptive Task Partitioning at Local Device or Remote Edge Server for Offloading in MEC". In: *2020 IEEE Wireless Communications and Networking Conference (WCNC)*. 2020.

[46] L. Ji and S. Guo. "Energy-Efficient Cooperative Resource Allocation in Wireless Powered Mobile Edge Computing". In: *IEEE Internet of Things Journal* 6 (2019).

[47] X. Cao, F. Wang, J. Xu, R. Zhang, and S. Cui. "Joint Computation and Communication Cooperation for Energy-Efficient Mobile Edge Computing". In: *IEEE Internet of Things Journal* 6 (2019).

[48] J. Li and T. Lv. "Deep Neural Network based Computational Resource Allocation for Mobile Edge Computing". In: *2018 IEEE Globecom Workshops (GC Wkshps)*. 2018.

[49] Q. Pham, L. B. Le, S. Chung, and W. Hwang. "Mobile Edge Computing With Wireless Backhaul: Joint Task Offloading and Resource Allocation". In: *IEEE Access* 7 (2019).

[50] M. A. Hossain and N. Ansari. "Energy Aware Latency Minimization for Network Slicing Enabled Edge Computing". In: *IEEE Transactions on Green Communications and Networking* (2021). Early Access.

[51] L. Feng, Y. Zhou, T. Liu, X. Que, P. Yu, T. Hong, and X. Qiu. "Energy-Efficient Offloading for Mission-Critical IoT Services Using EVT-Embedded Intelligent Learning". In: *IEEE Transactions on Green Communications and Networking* 5 (2021).

[52] J. Ren, G. Yu, Y. Cai, and Y. He. "Latency Optimization for Resource Allocation in Mobile-Edge Computation Offloading". In: *IEEE Transactions on Wireless Communications* 17 (2018).

[53] J. Li, H. Gao, T. Lv, and Y. Lu. "Deep reinforcement learning based computation offloading and resource allocation for MEC". In: *2018 IEEE Wireless Communications and Networking Conference (WCNC)*. 2018.

[54] X. Lyu, H. Tian, W. Ni, Y. Zhang, P. Zhang, and R. P. Liu. "Energy-Efficient Admission of Delay-Sensitive Tasks for Mobile Edge Computing". In: *IEEE Transactions on Communications* 66 (2018).

[55] Y. Wang, X. Tao, Y. T. Hou, and P. Zhang. "Effective Capacity-Based Resource Allocation in Mobile Edge Computing With Two-Stage Tandem Queues". In: *IEEE Transactions on Communications* 67 (2019).

[56] T. Yang, Y. Hu, M. C. Gursoy, A. Schmeink, and R. Mathar. "Deep Reinforcement Learning based Resource Allocation in Low Latency Edge Computing Networks". In: *2018 15th International Symposium on Wireless Communication Systems (ISWCS)*. 2018.

[57] A. Khalili, S. Zarandi, and M. Rasti. "Joint Resource Allocation and Offloading Decision in Mobile Edge Computing". In: *IEEE Communications Letters* 23 (2019).

[58] Q. Li, J. Zhao, and Y. Gong. "Computation offloading and resource allocation for mobile edge computing with multiple access points". In: *IET Communications* 13 (2019).

[59] J. Zhang, X. Hu, Z. Ning, E. C. -. Ngai, L. Zhou, J. Wei, J. Cheng, and B. Hu. "Energy-Latency Tradeoff for Energy-Aware Offloading in Mobile Edge Computing Networks". In: *IEEE Internet of Things Journal* 5 (2018).

[60] S. Yang, F. Li, M. Shen, X. Chen, X. Fu, and Y. Wang. "Cloudlet Placement and Task Allocation in Mobile Edge Computing". In: *IEEE Internet of Things Journal* 6 (2019).

[61] P. Wang, C. Yao, Z. Zheng, G. Sun, and L. Song. "Joint Task Assignment, Transmission, and Computing Resource Allocation in Multilayer Mobile Edge Computing Systems". In: *IEEE Internet of Things Journal* 6 (2019).

[62] T. X. Tran and D. Pompili. "Joint Task Offloading and Resource Allocation for Multi-Server Mobile-Edge Computing Networks". In: *IEEE Transactions on Vehicular Technology* 68 (2019).

[63] J. Liu and Q. Zhang. "Computation Resource Allocation for Heterogeneous Time-Critical IoT Services in MEC". In: *2020 IEEE Wireless Communications and Networking Conference (WCNC)*. 2020.

[64] C.-F. Liu, M. Bennis, M. Debbah, and H. V. Poor. "Dynamic Task Offloading and Resource Allocation for Ultra-Reliable Low-Latency Edge Computing". In: *IEEE Transactions on Communications* 67 (2019).

[65] Q. Fan and N. Ansari. "Application Aware Workload Allocation for Edge Computing-Based IoT". In: *IEEE Internet of Things Journal* 5 (2018).

[66] L. Wang, L. Jiao, J. Li, J. Gedeon, and M. M hlhäuser. "MOERA: Mobility-Agnostic Online Resource Allocation for Edge Computing". In: *IEEE Transactions on Mobile Computing* 18 (2019).

[67] S. Jošilo and G. Dán. "Joint Allocation of Computing and Wireless Resources to Autonomous Devices in Mobile Edge Computing". In: *Proceedings of the 2018 Workshop on Mobile Edge Communications*. 2018.

[68] M. Chen and Y. Hao. "Task Offloading for Mobile Edge Computing in Software Defined Ultra-Dense Network". In: *IEEE Journal on Selected Areas in Communications* 36 (2018).

[69] J. Wang, L. Zhao, J. Liu, and N. Kato. "Smart Resource Allocation for Mobile Edge Computing: A Deep Reinforcement Learning Approach". In: *IEEE Transactions on Emerging Topics in Computing* (2019).

[70] J. Chen, S. Chen, Q. Wang, B. Cao, G. Feng, and J. Hu. "iRAF: A Deep Reinforcement Learning Approach for Collaborative Mobile Edge Computing IoT Networks". In: *IEEE Internet of Things Journal* 6 (2019).

[71] N. Zhu, X. Xu, S. Han, and S. Lv. "Sleep-Scheduling and Joint Computation-Communication Resource Allocation in MEC Networks for 5G IIoT". In: *2021 IEEE Wireless Communications and Networking Conference (WCNC)*. 2021.

[72] C.-W. Hsu, Y.-L. Hsu, and H.-Y. Wei. "Energy-Efficient Edge Offloading in Heterogeneous Industrial IoT Networks for Factory of Future". In: *IEEE Access* 8 (2020).

[73] S. Wang, M. Chen, X. Liu, C. Yin, S. Cui, and H. V. Poor. "A Machine Learning Approach for Task and Resource Allocation in Mobile Edge Computing Based Networks". In: *IEEE Internet of Things Journal* (2020).

[74] S. Sardellitti, M. Merluzzi, and S. Barbarossa. "Optimal Association of Mobile Users to Multi-Access Edge Computing Resources". In: *2018 IEEE International Conference on Communications Workshops (ICC Workshops)*. 2018.

[75] H. Li, H. Xu, C. Zhou, X. Lü, and Z. Han. "Joint Optimization Strategy of Computation Offloading and Resource Allocation in Multi-Access Edge Computing Environment". In: *IEEE Transactions on Vehicular Technology* 69 (2020).

[76] Z. Yang, C. Pan, K. Wang, and M. Shikh-Bahaei. "Energy Efficient Resource Allocation in UAV-Enabled Mobile Edge Computing Networks". In: *IEEE Transactions on Wireless Communications* 18 (2019).

[77] E. El Haber, H. A. Alameddine, C. Assi, and S. Sharafeddine. "UAV-aided Ultra-Reliable Low-Latency Computation Offloading in Future IoT Networks". In: *IEEE Transactions on Communications* (2021). Early Access.

[78] Y. Liu, K. Xiong, Q. Ni, P. Fan, and K. B. Letaief. "UAV-Assisted Wireless Powered Cooperative Mobile Edge Computing: Joint Offloading, CPU Control, and Trajectory Optimization". In: *IEEE Internet of Things Journal* 7 (2020).

[79] P. Wang, Z. Zheng, B. Di, and L. Song. "HetMEC: Latency-Optimal Task Assignment and Resource Allocation for Heterogeneous Multi-Layer Mobile Edge Computing". In: *IEEE Transactions on Wireless Communications* 18 (2019).

[80] J. Zhang, X. Hu, Z. Ning, E. C. -. Ngai, L. Zhou, J. Wei, J. Cheng, B. Hu, and V. C. M. Leung. "Joint Resource Allocation for Latency-Sensitive Services Over Mobile Edge Computing Networks With Caching". In: *IEEE Internet of Things Journal* 6 (2019).

[81] L. Yang, J. Cao, H. Cheng, and Y. Ji. "Multi-User Computation Partitioning for Latency Sensitive Mobile Cloud Applications". In: *IEEE Transactions on Computers* 64 (2015).

[82] R. Islambouli and S. Sharafeddine. "Optimized 3D Deployment of UAV-Mounted Cloudlets to Support Latency-Sensitive Services in IoT Networks". In: *IEEE Access* (2019).

[83] T. Zhang, Y. Xu, J. Loo, D. Yang, and L. Xiao. "Joint Computation and Communication Design for UAV-Assisted Mobile Edge Computing in IoT". In: *IEEE Transactions on Industrial Informatics* (2020).

[84] E. K. Markakis, I. Politis, A. Lykourgiotis, Y. Rebahi, G. Mastorakis, C. X. Mavromoustakis, and E. Pallis. "Efficient Next Generation Emergency Communications over Multi-Access Edge Computing". In: *IEEE Communications Magazine* 55 (2017).

[85] M. Mezzavilla, M. Polese, A. Zanella, A. Dhananjay, S. Rangan, C. Kessler, T. S. Rappaport, and M. Zorzi. "Public Safety Communications above 6 GHz: Challenges and Opportunities". In: *IEEE Access* 6 (2018).

[86] L. Li, X. Wen, Z. Lu, Q. Pan, W. Jing, and Z. Hu. "Energy-Efficient UAV-Enabled MEC System: Bits Allocation Optimization and Trajectory Design". In: *Sensors* (2019).

[87] J. Xiong, H. Guo, and J. Liu. "Task Offloading in UAV-Aided Edge Computing: Bit Allocation and Trajectory Optimization". In: *IEEE Communications Letters* 23 (2019).

[88] Z. Yu, Y. Gong, S. Gong, and Y. Guo. "Joint Task Offloading and Resource Allocation in UAV-Enabled Mobile Edge Computing". In: *IEEE Internet of Things Journal* 7 (2020).

[89] J. Zhang, L. Zhou, F. Zhou, B. Seet, H. Zhang, Z. Cai, and J. Wei. "Computation-Efficient Offloading and Trajectory Scheduling for Multi-UAV Assisted Mobile Edge Computing". In: *IEEE Transactions on Vehicular Technology* 69 (2020).

[90] R. Chai, J. Lin, M. Chen, and Q. Chen. "Task Execution Cost Minimization-Based Joint Computation Offloading and Resource Allocation for Cellular D2D MEC Systems". In: *IEEE Systems Journal* 13 (2019).

[91] S. Guo, B. Xiao, Y. Yang, and Y. Yang. "Energy-efficient dynamic offloading and resource scheduling in mobile cloud computing". In: *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*. 2016.

[92] Y. Wang, Z. -Y. Ru, K. Wang, and P. -Q. Huang. "Joint Deployment and Task Scheduling Optimization for Large-Scale Mobile Users in Multi-UAV-Enabled Mobile Edge Computing". In: *IEEE Transactions on Cybernetics* (2020).

[93] M. S. Elbamby, C. Perfecto, C.-F. Liu, J. Park, S. Samarakoon, X. Chen, and M. Bennis. "Wireless Edge Computing With Latency and Reliability Guarantees". In: *Proceedings of the IEEE* 107 (2019).

[94] D. Öhmann, M. Simsek, and G. P. Fettweis. "Achieving high availability in wireless networks by an optimal number of Rayleigh-fading links". In: *2014 IEEE Globecom Workshops (GC Wkshps)*. 2014.

[95] M. A. Mahmood, W. K.G. Seah, and I. Welch. "Reliability in wireless sensor networks: A survey and challenges ahead". In: *Computer Networks* 79 (2015).

[96] M. Bennis, M. Debbah, and H. V. Poor. "Ultrareliable and Low-Latency Wireless Communication: Tail, Risk, and Scale". In: *Proceedings of the IEEE* 106 (2018).

[97] S. P. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. "Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers". In: *Found. Trends Mach. Learn.* 3 (2011).

[98] N. Buchbinder, S. Chen, and J. Naor. "Competitive Analysis via Regularization". In: *SODA*. 2014.

[99] Z. Li and Q. Zhu. "Genetic Algorithm-Based Optimization of Offloading and Resource Allocation in Mobile-Edge Computing". In: *Information* 11 (2020).

[100] L. Wan, L. Sun, X. Kong, Y. Yuan, K. Sun, and F. Xia. "Task-Driven Resource Assignment in Mobile Edge Computing Exploiting Evolutionary Computation". In: *IEEE Wireless Communications* 26 (2019).

[101] M. Bowling and M. Veloso. "Multiagent learning using a variable learning rate". In: *Artificial Intelligence* 136 (2002).

[102] M. Li, N. Cheng, J. Gao, Y. Wang, L. Zhao, and X. Shen. "Energy-Efficient UAV-Assisted Mobile Edge Computing: Resource Allocation and Trajectory Optimization". In: *IEEE Transactions on Vehicular Technology* 69 (2020).

[103] Y. K. Tun, Y. M. Park, N. H. Tran, W. Saad, S. R. Pandey, and C. S. Hong. "Energy-Efficient Resource Management in UAV-Assisted Mobile Edge Computing". In: *IEEE Communications Letters* 25 (2021).

[104] Q. Hu, Y. Cai, G. Yu, Z. Qin, M. Zhao, and G. Y. Li. "Joint Offloading and Trajectory Design for UAV-Enabled Mobile Edge Computing Systems". In: *IEEE Internet of Things Journal* (2019).

[105] F. Zhou, Y. Wu, R. Q. Hu, and Y. Qian. "Computation Rate Maximization in UAV-Enabled Wireless-Powered Mobile-Edge Computing Systems". In: *IEEE Journal on Selected Areas in Communications* 36 (2018).

[106] Y. Qian, F. Wang, J. Li, L. Shi, K. Cai, and F. Shu. "User Association and Path Planning for UAV-Aided Mobile Edge Computing With Energy Restriction". In: *IEEE Wireless Communications Letters* 8 (2019).

[107] S. Wan, J. Lu, P. Fan, and K. B. Letaief. "Toward Big Data Processing in IoT: Path Planning and Resource Management of UAV Base Stations in Mobile-Edge Computing System". In: *IEEE Internet of Things Journal* 7 (2020).

[108] L. Yang, H. Yao, J. Wang, C. Jiang, A. Benslimane, and Y. Liu. "Multi-UAV-Enabled Load-Balance Mobile-Edge Computing for IoT Networks". In: *IEEE Internet of Things Journal* (2020).

[109] G. Wu, Y. Miao, Y. Zhang, and A. Barnawi. "Energy efficient for UAV-enabled mobile edge computing networks: Intelligent task prediction and offloading". In: *Comput. Commun.* 150 (2020).

[110] Y. Chen and Z. Zheng. "Joint Deployment and Task Computation of UAVs in UAV-assisted Edge Computing Network". In: *2020 21st Asia-Pacific Network Operations and Management Symposium (APNOMS)*. 2020.

[111] J. Wang, K. Liu, and J. Pan. "Online UAV-Mounted Edge Server Dispatching for Mobile-to-Mobile Edge Computing". In: *IEEE Internet of Things Jour- nal* 7 (2020).

[112] M. Razaviyayn, M. Hong, Z. Luo, and J. Pang. "Parallel Successive Convex Approximation for Nonsmooth Nonconvex Optimization". In: *NIPS*. 2014.

[113] D. S. Hochbaum and A. Pathria. "Analysis of the greedy approach in problems of maximum k-coverage". In: *Naval Research Logistics (NRL)* 45 (1998).

[114] B. Javidy, A. Hatamlou, and S. Mirjalili. "Ions motion algorithm for solving optimization problems". In: *Applied Soft Computing* 32 (2015).

AUTHORS

Nina Santi is a PhD student under the supervision of Nathalie Mitton in the Inria FUN team. She has received an MSc degree in computer science from University of Lille, France, in 2020. Her research interests are wireless network architectures like edge computing, autonomous or self-managing systems and resource management.

Nathalie Mitton received MSc and PhD degrees in computer science from INSA Lyon in 2003 and 2006 respectively. She received her Habilitation à diriger des recherches (HDR) in 2011 from Université Lille 1. She is currently an Inria full researcher since 2006 and from 2012, she is the scientific head of the Inria FUN team. Her research interests focus on self-organization from PHY to routing for wireless constrained networks. She has published her research in more than 50 international journals and more than 120 international conferences. She is involved in the setup of the FIT IoT LAB platform (http://fit-equipex.fr, https://www.iot-lab.info), the H2020 Cyber- SANE and VESSEDIA projects and in several program and organization committees such as Infocom (since 2019), PerCom (since 2019), DCOSS (since 2019), Adhocnow (since 2015), ICC (since 2015), Globecom (since 2017), Pe-Wasun 2017, VTC (since 2016), etc. She also supervises several PhD students and engineers.