# Slicing-Based Software-Defined Mobile Edge Computing in the Air

| | |
|---|---|
| Journal: | *IEEE Wireless Communications Magazine* |
| Manuscript ID | WCM-21-00303 |
| Topic or Series: | February 2022/Smart, Optimal, and Explainable Orchestration of Network Slices in 5G and Beyond Networks |
| Date Submitted by the Author: | 14-May-2021 |
| Complete List of Authors: | Tang, Jianhang; Yanshan University<br>Nie, Jiangtian; Nanyang Technological University, School of Computer Science and Engnieering<br>Zhao, Jun; Nanyang Technological University, School of Computer Science and Engineering; Nanyang Technological University<br>Zhou, Yi; Wuhan University of Technology,<br>Xiong, Zehui; Singapore University of Technology and Design, School of Computer Science and Engineering (SCSE)<br>Guizani, Mohsen; Qatar University, CSE |
| Key Words: | |
| | |

# Slicing-Based Software-Defined Mobile Edge Computing in the Air

Jianhang Tang, Jiangtian Nie, Jun Zhao, Yi Zhou, Zehui Xiong, and Mohsen Guizani

**Abstract**

Unmanned aerial vehicles (UAV)-enabled mobile edge computing (MEC) is promising to fulfill the growing resource demands of the emerging Internet of Things (IoT) applications, where flying UAVs act as complementary components for enhancing MEC systems. A UAV-enabled MEC system consists of aerial and ground segments, both including various hardware devices and facilities, e.g., UAVs, edge servers, and network devices. Such a hierarchical structure requires an efficient framework to manage and control all the devices and system resources while considering specific characteristics of the UAVs in the system. In this work, we propose a new systematic architecture termed slicing-based software-defined MEC in the air, where the concept of software-defined control is integrated into UAV-enabled MEC systems. Meanwhile, to provide customized UAV-enabled MEC services, resource slices are utilized to offer customized multi-dimensional resources by integrating virtual storage and computation resource blocks provided by both UAVs and edge devices. Based on the proposed slicing-based software-defined scheme, we discuss two potential application scenarios and some open issues. Particularly, we develop a Lyapunov optimization-based offloading algorithm for the efficient selection of resource slices. Finally, we present the simulation results to depict the effectiveness of the proposed scheme.

**Index Terms**

Unmanned aerial vehicle, mobile edge computing, Internet of Things, resource slicing, computation offloading

## I. Introduction

Recently, the unprecedented high requirements of the emerging IoT applications impose a noticeable challenge on mobile terminals with limited battery capacities and computation resources. Cloud computing

J. Tang is with the School of Information Science and Engineering, Yanshan University, Qinghuangdao, China. J. Nie and J. Zhao are with the School of Computer Science and Engineering, Nanyang Technological University, Singapore. Y. Zhou is with the School of Energy and Power Engineering, Wuhan University of Technology, Wuhan, China. Z. Xiong is with the Pillar of Information Systems Technology and Design, Singapore University of Technology and Design, Singapore. M. Guizani is with the Department of Computer Science and College of Engineering, Qatar University, Doha, Qatar.

provides a shared pool of resources (e.g., storage and computing) to enhance the capacities of end terminal devices. Nevertheless, dispatching IoT applications to remote cloud services incurs significant costs, which is infeasible to IoT applications due to energy, storage, and computing constraints. *Mobile edge computing* (MEC) [1] has emerged as a solution to supply various system resources at the edge of the network, which can potentially reduce mobile energy consumption and communication overhead. With a myriad of end devices popping up, MEC is encountering new challenges, e.g., higher bandwidth requirement, ultra-low latency, and massive connectivity. A novel paradigm termed UAV-enabled MEC has been proposed [2], which integrates unmanned aerial vehicles (UAVs) into MEC. With the features of flexibility, fast deployment, vast coverage, and adaptive mobility, UAVs constructed by various modules [3] (e.g., power systems, communication modules, and on-board computers) are considered as important complements of MEC networks. UAVs in MEC systems can act as relays to assist users in transmitting data [4], or MEC servers in processing data [5]. Instead of utilizing a single scheme, UAV-enabled MEC can provide seamless connectivity and adequate resources for end devices by fusing the benefits of both aerial and ground segments.

To reduce the overhead in network management and enhance the network flexibility, *software-defined networking* (SDN) technology is introduced in the UAV-enabled MEC systems [6]. SDN separates the control plane from the underlying data plane, and offers centralized SDN controllers with standard services via application programming interfaces (APIs). The conceptual idea behind SDN is to handle the management and control of network devices all by standard software. With SDN, the management of UAVs with high mobility can be merged into the conventional network (namely ground network) seamlessly. Flexibility is achieved, as new SDN applications can be adopted efficiently without the knowledge and control of underlying network facilities due to the network programmability introduced by SDN. With the usage of SDN as well as the network functions virtualization (NFV) functionality within, plenty of service-oriented network slices are provided on demand over a common aerial-ground network infrastructure [7]. Network slices are mutually isolated from each other, and equipped with independent management and control modules. By leveraging network slicing, UAV-enabled MEC systems can offer standardized and customized network services for various IoT applications with diversified requirements.

However, a UAV-enabled MEC system is composed of both aerial and ground layers, where the aerial layer includes multiple UAVs, and the ground layer consists of various devices, such as base stations (BSs), access points (APs), switches, as well as edge servers. Such a hierarchical structure requires sophisticated but efficient network management to comprehensively control the hierarchical network architecture and

utilize network resources. Meanwhile, conventional network slicing mainly optimizes resources only for networking functionalities. As such, a dedicated control scheme particularly for UAV-enabled MEC is required to improve the utilization of multi-dimensional and compound resources, e.g., storage, wireless communication channel, and computation resources. For example, we consider a scenario where smart vehicular users require ultra-high definition (UHD) video services for enhanced passenger entertainment experiences, and autonomous driving services, both involving a UAV-enabled MEC system simultaneously. The video services require more network communication resources, while the autonomous driving services need more real-time computational resources.

To tackle the above issues, we propose a new systematic architecture termed slicing-based software-defined UAV-enabled MEC, where software-defined technologies [8] are integrated into MEC systems with UAV devices. In the proposed scheme, resource slices are created by integrating virtual storage and computation resource blocks provided by both UAVs and ground edge devices into network slices. In Section II, we outline the architecture of slicing-based software-defined UAV-enabled MEC. In Section III, we identify some potential application scenarios and key research challenges. A Lyapunov optimization-based offloading algorithm with the selection of resource slices is also proposed in Section IV. We also demonstrate and evaluate the performance of the optimization algorithm in the proposed architecture. Finally, Section V concludes the article.

## II. Slicing-Based Software-Defined UAV-Enabled MEC

In this section, we develop a systematic architecture termed slicing-based software-defined UAV-enabled MEC, by integrateing the software-defined concept into UAV-enabled MEC systems and uses resource slicing to offer customized services for diverse IoT applications. Based on the system architecture, resource slicing patterns with UAVs are introduced.

### *A. Framework Design of Slicing-Based Software-Defined UAV-Enabled MEC*

As depicted in Fig. 1, a layered architecture of the slicing-based software-defined UAV-enabled MEC is proposed. Functionalities of the architectures are categorized into four logical planes, including data, control, application, and user planes. The design details of different planes are described as follows.

- **Data Plane** includes all underlying facilities (e.g., UAVs, BSs, APs, and edge servers). It is the cornerstone of the proposed framework. These hardware elements do not generate any control rules and operate according to the commands produced from the control plane. Resource slices are achieved by adding virtual storage and computation resource blocks into network slices composed of virtual

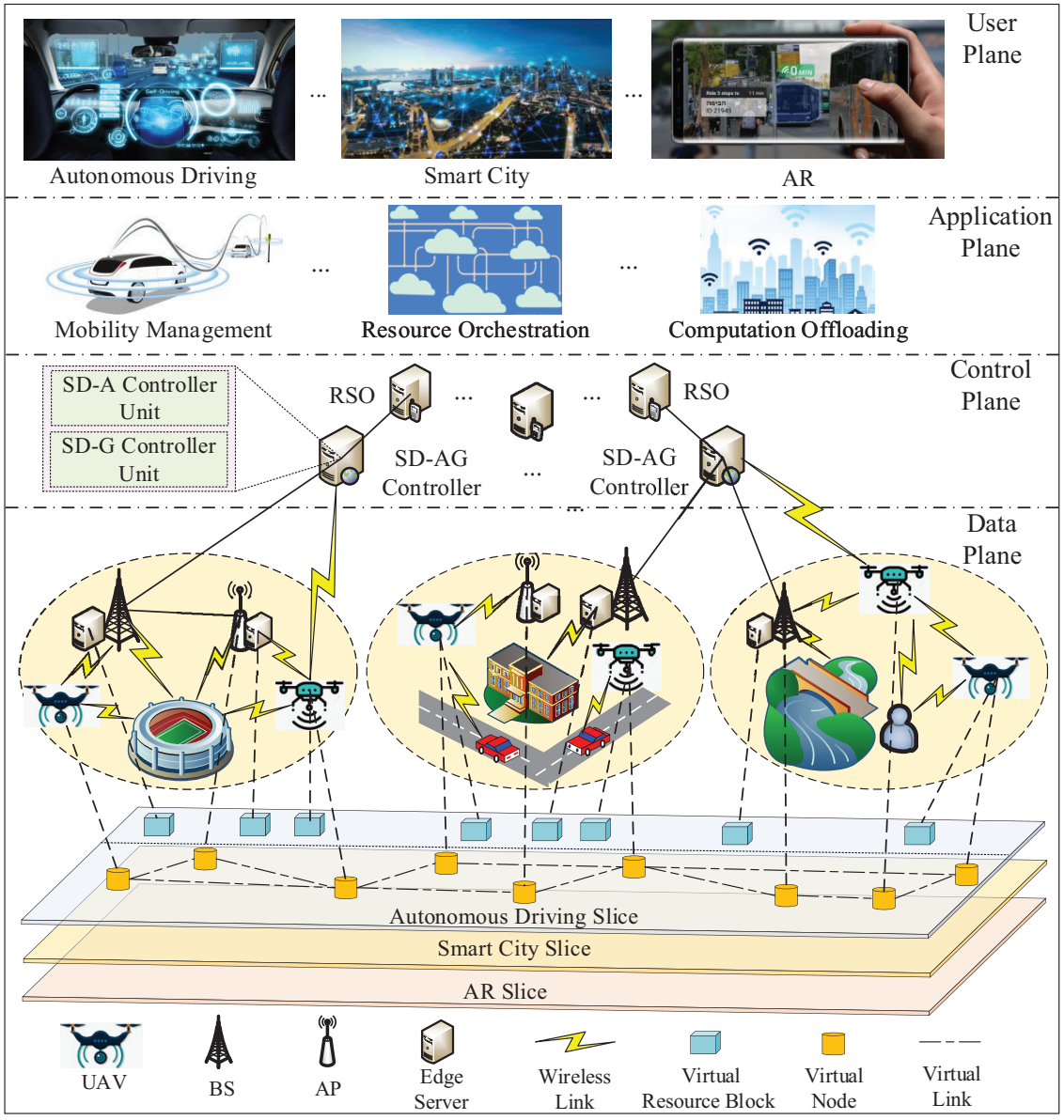



Fig. 1: The architecture of slicing-based software-defined UAV-enabled MEC.

nodes and links, where virtual resource blocks are mainly provided by edge servers and onboard micro-computers deployed on UAVs. A particular resource slice provides required services for IoT applications with similar resource demands.

- **Control Plane** is the kernel part of the architecture, which comprises distributed software-defined aerial and ground (SD-AG) controllers and resource slicing orchestrators (RSOs). To fully exploit the benefits of distinguishing characteristics of flying UAVs and terrestrial edge computing systems, an SD-AG controller is organized into software-defined air (SD-A) and ground (SD-G) controller units, which manages aerial and terrestrial facilities separately. SD-AG controllers communicate with each

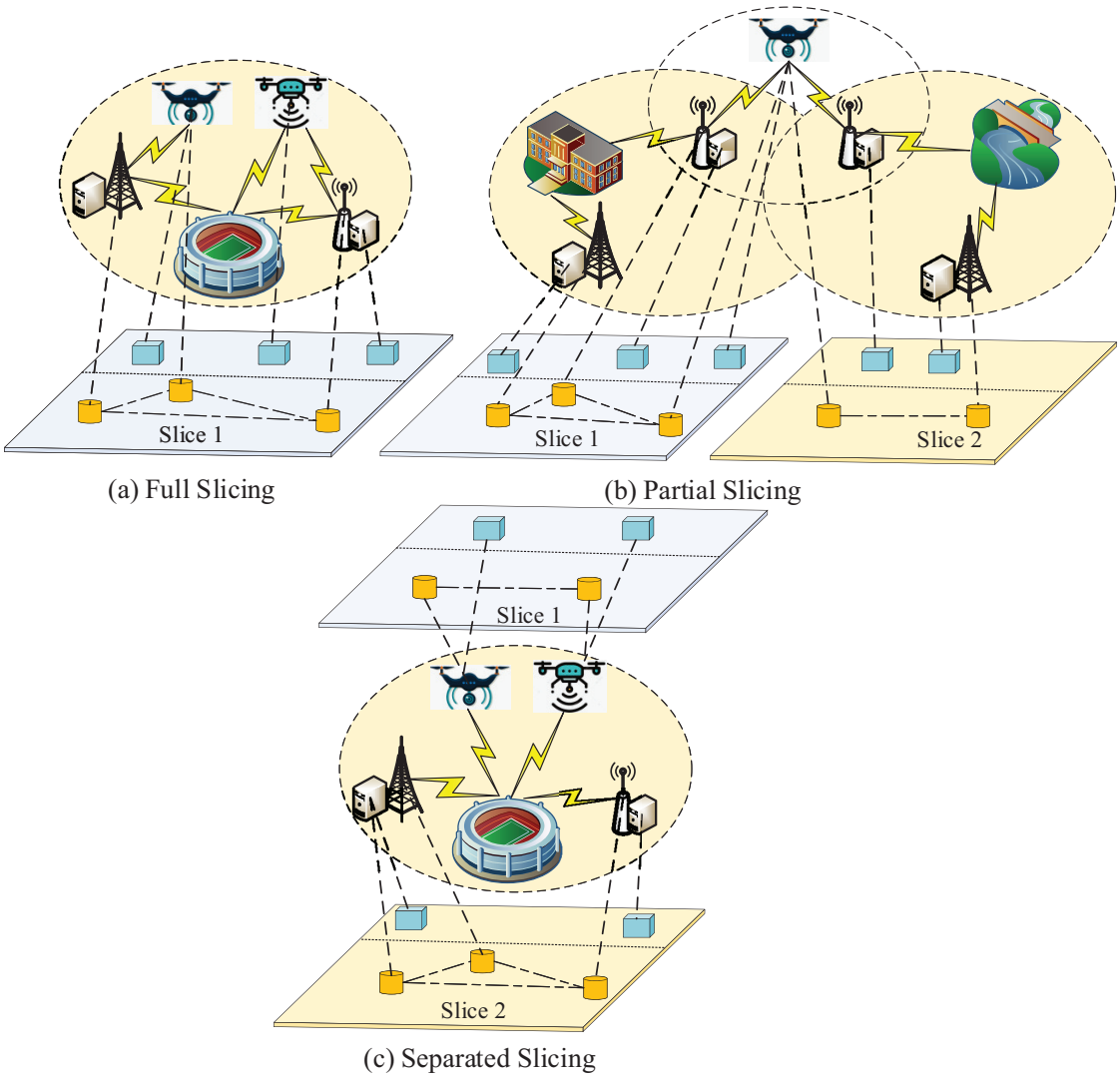
other through east-west bound APIs and deliver control commands by southbound APIs. RSOs are in charge of creating, maintaining, and terminating resource slices. With centralized SD-AG controllers, the RSOs can orchestrate resource slices flexibly and efficiently. Due to the mobility and geographic coverage of UAVs, the synergy of an SD-AG controller and an RSO is responsible for harmonizing the utilization of both aerial and terrestrial resources in a fixed geographical area.

- **Application Plane** implements various system applications (e.g., mobility management, resource orchestration, and computation offloading, etc.). These system applications are installed on SD-AG controllers through northbound APIs without any knowledge of hardware features, which makes the whole system flexible and programmable.
- **User Plane** is at the top layer of the framework, which includes multiple IoT applications with diverse requirements. The user plane communicates with the application plane through external APIs. When IoT applications arrive at the system, SD-AG controllers will assign related system applications to them.

With the hierarchical structure, the proposed framework can provide the following benefits. Firstly, the control plane is separated from the data plane to mitigate the impacts of the hardware diversity. The logically centralized but physically distributed SD-AG controllers are proposed to configure aerial and terrestrial devices separately, simplifying system management and evolution. As a result, SD-AG controllers are in charge of system control whereas the aerial and terrestrial devices are responsible for forwarding, storing, and processing data. Moreover, standard and general northbound APIs make the system programmable to support burgeoning IoT applications. Finally, resource slices are designed to offer customized services over a common UAV-enabled MEC infrastructure to improve the utilization of multi-dimensional resources.

### *B. Resource Slicing Patterns with UAVs*

In the slicing-based software-defined UAV-enabled MEC, resources from both aerial and terrestrial devices are shared among resource slices to enable customization of resources and services for the requirements of distinct use cases. Such a hierarchical framework explores the benefits of UAVs with high mobility and flexible geographic coverage, which differs from the regular MEC with fixed servers. To ensure that the resource slices match with the mobility and coverage of UAVs, we identify the resource slicing patterns implemented by organized embeddings of UAVs, including full slicing, partial slicing, and separated slicing, as shown in Fig. 2.




(a) Full Slicing (b) Partial Slicing

(c) Separated Slicing

Fig. 2: Resource slicing patterns with UAVs.

In the case of **full slicing**, UAVs provide resources for a single area and cooperate with terrestrial facilities in Fig. 2(a). The aerial and ground layers operate as an integrated system to provide vast resources for IoT applications in a fixed geographical range. Consequently, the full slicing scheme can improve resource utilization. It is easy to realize full slicing with a local SD-AG controller and a local RSO.

In the **partial slicing** case, a few UAVs are located at boundaries of geographical areas and controlled by different SD-AG controllers to offer services for multiple regions simultaneously, which is depicted in Fig. 2(b). In this situation, resources provided by shared UAVs are partitioned and assigned to resource slices as supplementary parts to provide additional resources quickly when the demands burst. To reach consensus among several SD-AG controllers for partial slicing, blockchain technology [9] is a promising solution. Blockchain is a decentralized digital ledger maintained by participants who may not trust each other in a

peer-to-peer network, where consensus algorithms (e.g., Proof of Stake (PoS) and Proof of Work (PoW)) are required among peer nodes. UAVs and edge servers can provide storage and computation resources for computation tasks like PoW puzzles in blockchain systems.

UAVs may fly frequently from one region to another, the resource slices related to the leaving ones may stop working, thereby reducing the quality of service (QoS). It is hard to provide survivable slices by full slicing and partial slicing. In Fig. 2(c), we illustrate the **separated slicing** case where aerial and terrestrial resources are allocated to distinct slices separately. The terrestrial slices are the main components to provide reliable resources, while aerial slices are considered as supplementary to these terrestrial slices to offer resources for some IoT applications which are not sensitive to delay. In this case, the mobility and failure of UAVs have little impact on system performance. Instead of migrating resource slices by backbone networks across regions, the resource slices supplied by UAVs can be migrated with their mobility, which relieves the communication burden.

## III. Applications and Challenges

### *A. Application Scenarios*

In the slicing-based software-defined UAV-enabled MEC framework, UAVs act as a complement of MEC to provide adequate resources. To highlight the potential MEC performance improvements with the usage of UAVs, two typical application scenarios are presented.

*Smart Transportation:* Smart transportation [10] has been rapidly developing to deal with the shortcomings of current transportation systems and prospective requirements for urban mobility, which mainly include intelligent traffic control and self-driving vehicles. The goal of intelligent traffic control is to enhance driving experiences and security by a comprehensive system in which traffic control units and vehicles share traffic information. To obtain real-time and detailed road information, more network resources are needed. Self-driving vehicles are mainly composed of perception and control modules. To implement swift control actions (e.g., brake and throttle) over vehicles, ultra-reliable and low-latency services are required based on the road situation provided by the perception module. Without restriction to practical road topologies, UAVs are free to move across streets, acting as flying roadside units (RSUs) and speed cameras. Moreover, two dedicated resource slices are generated for intelligent traffic control and self-driving vehicles, where one is filled with communication resources and the other one can provide more computation resources.

*Disaster Management:* Natural catastrophes and and man-made disasters (e.g., earthquake, floods, and fire) resulting in loss of lives and properties bring our attention to conducting search and rescue operations

efficiently with a fast response system [11]. The disasters may result in surface rupture and damage to ground communication infrastructure. It is time-consumed to fully recover the ground communication system, which may miss the first 72 hours after disasters hit. With the usage of UAVs, the situation of affected regions is assessed by first responders as early as possible. Various resource slices are provided for different government departments according to their roles and responsibilities. Moreover, new system applications, such as UAV deployment and connection algorithms, can be installed on SD-AG controllers by northbound APIs efficiently.

As discussed in this section, the flying UAVs are considered as core parts of the proposed architecture to support a wide range of services. However, it is hard to allocate the control resources offered by SD-AG controllers due to the high mobility of UAVs. Meanwhile, the inefficient configuration of multi-dimensional resources provided by UAVs will result in low-quality and unreliable services. Thus, how to allocate such various resources is a challenging issue. Next, we will discuss a series of open issues of resource management for further improvements.

### *B. Controller Placement*

The logically centralized SD-AG controllers are the most important components of the proposed architecture, controlling flying UAVs, edge servers, and underlying network devices. With multiple physically distributed controllers, one key issue is to place controllers, which refers to how to deploy these controllers on the ground and how to select associated hardware facilities, especially UAVs, to SD-AG controllers to improve performance metrics including latency and reliability. To achieve a global view of the system and implement various functions, SD-AG controllers exchange messages with other controllers and with underlying devices frequently. As a result, the controller response time is a critical index, including transmission and processing latencies. The transmission latency, including inter-controller and device-controller latencies, is primarily affected by network distances. The processing latency is mainly determined by controller workloads. Meanwhile, the failures of both network links and controller instances can result in disconnected devices, which will increase the latency drastically. Thus, it is significant to design controller placement schemes by reducing latency and improving reliability. Besides, the mobility of flying UAVs leads to time-varying network topology, which makes the controller placement problem more complex.

### C. Mobility Management

In the proposed framework, mobility management involves the following two major research directions: UAV trajectory optimization [12] and resource slice migration. To make the best of the resources carried by UAVs and improve the performance of resource slices, it is desirable to design trajectory optimization strategies corresponding to various optimization objectives. Since the capacity of the onboard battery is often limited. The energy efficiency of flying UAVs should be increased by optimizing their trajectories. UAVs are required to move frequently to supply multi-dimensional resources for sparsely distributed end devices, while mechanical actions for flying generate a large part of energy consumption. Meanwhile, the other part of energy consumption mainly stems from the service provision. Thus, how to reduce the total energy of resource slices with UAV trajectory optimization is a key issue in the proposed architecture.

When users move with their end devices through adjacent or overlapped geographical regions, the ongoing services running on resource slices may drop due to the disconnection. Besides, these ongoing services have to be served by a new resource slice with the same resource types in the destination area, which leads to resource slices that have to support the migration. The resource slice migration can be classified into the following two main categories: full slice migration and partial slice migration. When a group of end devices moves to a different region, a full slice migration is required to migrate all resources and the services of a resource slice from the original area to a new one. It is costly to migrate a full resource slice all the time. With partial slice migration, only identified services that are not provided by the resource slice in the destination location are migrated. Then, a slice merging scheme is required to offer service seamlessly.

### D. Resource Management

Due to the limited resources provided by underlying hardware facilities, resource management is a critical issue, where the resource slice embedding problem and resource orchestration problem are two major challenges. The problem of embedding resource slices is to deal with the efficient mapping of virtual network elements and virtual resource blocks onto substrate system resources. There are three sub-problems in this respect: virtual node mapping, virtual link mapping, and virtual resource block mapping. UAVs can host several virtual nodes and supply a collection of virtual resource blocks. Since virtual links are generated with a set of ordered virtual nodes, the flying UAVs can affect the overall performance of the embedding. If a UAV moves outside the region, the missing virtual nodes, links, and resource blocks concerning this UAV may degrade the slice performance. Fallback resources are required to enable the

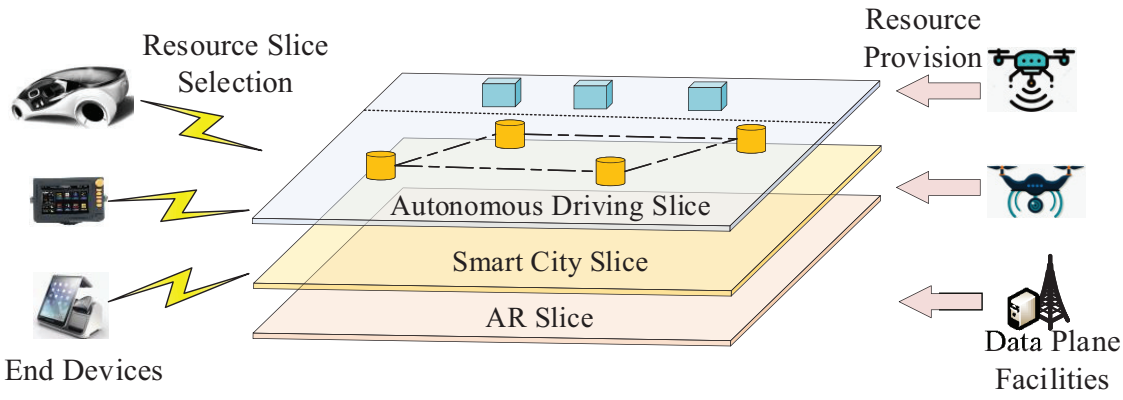



Fig. 3: An illustrative example with three end devices, three data plane facilities, and three resource slices.

realization of resilience, where backup nodes/links/resource blocks are generated for UAVs that move frequently. However, it is inefficient and impractical to set up a complete backup for all virtual resources. Thus, a fast re-embedding method is also required after the mobility of UAVs. Usually, a combination of backup and re-embedding approaches may obtain reliable resource slices.

The resource orchestration problem is to allocate multi-dimensional resources for resource slices, which is divided into two sub-problems: inter-slice orchestration where resources are assigned among different slices, and intra-slice orchestration where resources are allotted to users in a slice. The resource orchestration can be implemented by static orchestration or elastically dynamic orchestration. Owing to the stochastic resource demands, the dynamic orchestration makes the usage of multi-dimensional resources more efficient. With centralized SD-AG controllers, enough system observation information is collected for data-driven approaches (e.g., machine learning methods) to obtain a deep system observation and adaptive system control. Since the total number of resources allocated to users is achieved by the inter-slice orchestration result, the coupling between inter-slice and intra-slice orchestrations increases the complexity of the resource orchestration problem.

## IV. Efficient Resource Allocation in UAV-Enabled MEC

### A. *Computation Offloading*

To augment the capabilities of end devices, computation tasks are offloaded to resourceful edge servers and UAVs. Under the proposed framework, logically centralized SD-AG controllers select resource slices for tasks according to the information of end devices, such as the input size of the task, the remaining battery capacity, and the transmit power of the end device. After that, both wireless and computation resources should be allocated to different users in each slice, where a channel bandwidth is split into

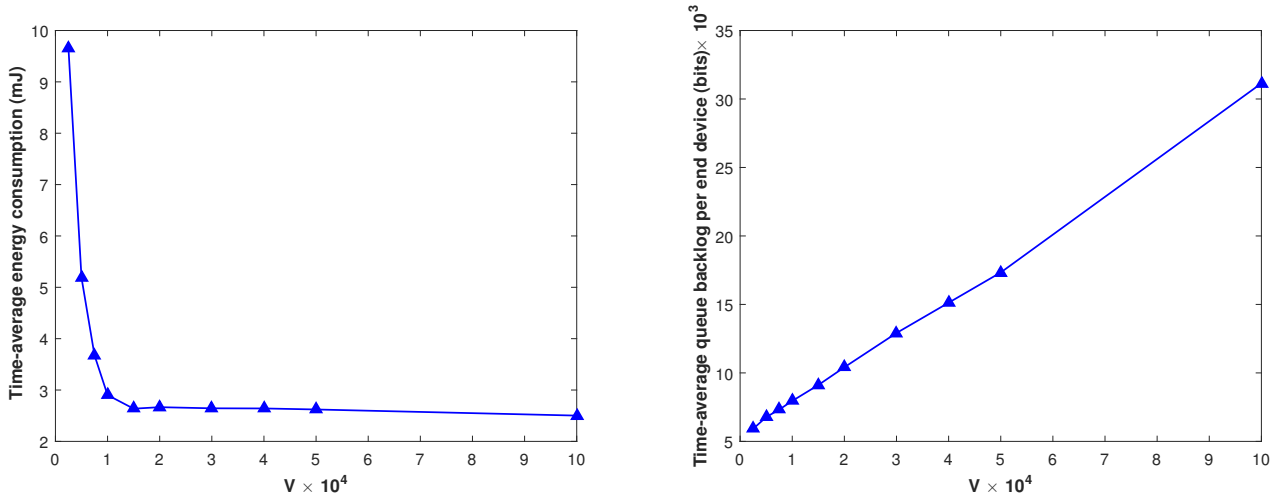
multiple physical resource blocks (PRBs). Due to the isolation of resource slices, only the end devices associating with the same BS or AP in a resource slice can affect each other. In this work, a sub-channel is assigned to end devices as a PRB and only provides services for a resource slice. Since each sub-channel can be used by multiple end devices in a resource slice, we follow the key idea underlying non-orthogonal multiple access (NOMA) to mitigate co-channel interferences by utilizing the successive interference cancellation. The achievable data rate experienced by an end device is presented in [13].

Figure 3 illustrates an example of computation offloading with the selection of resource slices. There are three resource slices where both communication and computation resources are provided by data plane facilities. Based on diverse resource requirements of tasks, three end devices select the corresponding resource slices to achieve the required services. However, the wireless links and the UAV states are time-varying and randomly uncertain, which makes the offloading problem more complicated and hard to model. Lyapunov optimization has been widely used to generate online offloading decisions [14]. With this framework, we do not need to achieve the task arrival distribution and prior wireless channel knowledge and quantify the system states and feasible control operations. In contrast, the task queue and channel states are observed at the beginning of each time slot. Then, by using drift-plus-penalty with control parameter $V$, a time-average energy minimization problem is transformed into several deterministic problems that can be solved with low complexity in each time slot. For any given $V > 0$, an online offloading algorithm can be presented to show that the time-average energy consumption differs from the optimal value by no more than a constant. Moreover, a tradeoff of $[O(1/V), O(V)]$ between the time-average energy consumption and task queue backlog is obtained.

### *B. Demonstration and Simulation Results*

As aforementioned, with slicing-based computation offloading, end devices select resource slices supplied by UAVs and edge servers, whereby end devices are associated with resource slices to receive customized services. Thus, computation offloading is the first step related to other key issues and plays a vital role in the proposed architecture. Due to the varying wireless links and UAV states, Lyapunov optimization is preferred for online offloading decision making. To investigate the performance of the Lyapunov-based offloading method in slicing-based software-defined UAV-enabled MEC, we present some simulation results in comparison with two benchmark schemes by setting different system parameters.

We employ a workstation with Intel Core i7-8550U CPU @1.80GHz to run these simulations. The details on simulation parameters are shown as follows. We consider a system including ten end devices operating at 1GHz. Two resource slices are equipped with 15vCPUs and 30vCPUs provided by edge




(a) (b)

Fig. 4: Time-average energy consumption/task queue backlog per end device vs. control parameter $V$.

servers and UAVs, where a vCPU operates at 2.3GHz. Meanwhile, these two resource slices are equipped with 100 PRBs and 50 PRBs provided by an AP and a UAV respectively, where the bandwidth of a PRB is set to 100kHz. Similar to [15], we ignore the transmission delay for the feedback of computed results from resource slices to end devices. The details on the channel gains are described in [13]. At the beginning of a time slot, an end device generate a computation task $T_i\left(L_i, C_i\right)$, where $L_i$ denotes the input size and $C_i$ is the number of CPU cycles to process one-bit input for local computing. The length of each time slot is set to 0.2s. $L_i$ is uniformly distributed within $[0, 15]$ kbits and $C_i$ is distributed within $[1000, 2000]$ cycles/bit. The results are achieved over 2000 time slots.

To compare with the proposed Lyapunov optimization-based offloading method, we consider the following two benchmark schemes, including 1) a local computing scheme only using end devices to process tasks, 2) an offloading scheme without local computing.

Figure 4 plots the relationship between the time-average energy consumption/task queue backlog per end devices and control parameter $V$. Fig. 4(a) depicts that the time-average energy consumption obtained by the proposed offloading scheme reduces inversely proportional to control parameter $V$ and it converges to the minimum energy consumption eventually. Meanwhile, Fig. 4(b) shows that the time-average queue backlog increases linearly with control parameter $V$. As a result, Fig. 4 verifies the tradeoff of $[O(1/V), O(V)]$ between the time-average energy consumption and task queue backlog, which is proved by the Lyapunov optimization theory. Then, we can balance the achievable system performance and energy consumption by setting the different values of $V$.

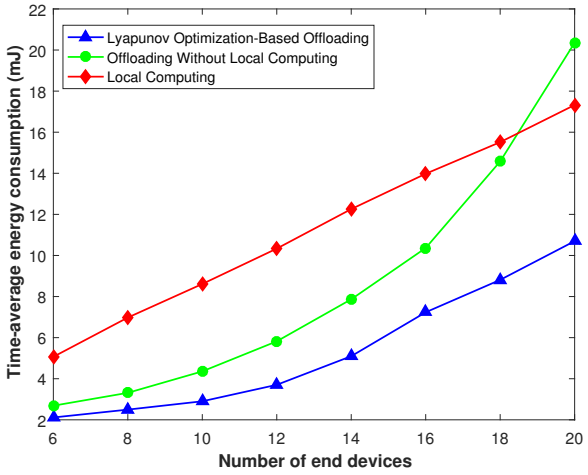



Fig. 5: Time-average energy consumption vs. number of end devices.

As observed from Fig. 5, the time-average energy consumption increases with the increased number of end devices in the proposed architecture. This is because more computation tasks require more communication and computation resources, and then the energy consumed by new end devices will add to the total energy consumption. By increasing the number of end devices from 6 to 18, the local computing scheme leads to higher energy consumption by comparing with the offloading scheme without local computing. The reason is that end devices can explore the advantage of offloading in terms of energy saving with enough communication resources. When the number of end devices is greater than 18, the offloading scheme without local computing leads to higher energy consumption. This is due to the fact that limited communication resources cannot satisfy the requirements of more end devices. Among the three benchmark schemes, the proposed Lyapunov optimization-based offloading scheme reach the best performance all the time. The reason is that the proposed scheme can improve the energy efficiency for both local execution and task offloading.

## V. Conclusion

In this article, we have proposed a new systematic architecture termed slicing-based software-defined UAV-enabled MEC, where resource slices are proposed to offer customized multi-dimensional resources for diverse IoT applications. With the flexibility and mobility of UAVs, we have identified various resource slicing schemes and key research challenges. Then, we have presented a Lyapunov optimization-based offloading algorithm with the selection of resource slices. Finally, we have presented simulation results to evaluate the performance of the proposed scheme. In future studies, a practical SD-AG controller and RSO will be designed with open-source tools, such as Kubernetes and OpenDaylight.

# REFERENCES

[1] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2322–2358, 2017.

[2] Y. Zhou, C. Pan, P. L. Yeoh, K. Wang, M. Elkashlan, B. Vucetic, and Y. Li, "Secure communications for uav-enabled mobile edge computing systems," *IEEE Trans. Commun.*, vol. 68, no. 1, pp. 376–388, 2019.

[3] W. Zhang, L. Li, N. Zhang, T. Han, and S. Wang, "Air-ground integrated mobile edge networks: A survey," *IEEE Access*, vol. 8, pp. 125 998–126 018, 2020.

[4] X. Lu, D. Xu, L. Xiao, L. Wang, and W. Zhuang, "Anti-jamming communication game for uav-aided vanets," in *Proceedings of the 2017 IEEE Global Communications Conference (GLOBECOM)*. IEEE, Dec. 2017, pp. 1–6.

[5] F. Zhou, Y. Wu, R. Q. Hu, and Y. Qian, "Computation rate maximization in uav-enabled wireless-powered mobile-edge computing systems," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 9, pp. 1927–1941, 2018.

[6] Z. Zhou, J. Feng, L. Tan, Y. He, and J. Gong, "An air-ground integration approach for mobile edge computing in iot," *IEEE Commun. Mag.*, vol. 56, no. 8, pp. 40–47, 2018.

[7] G. Faraci, C. Grasso, and G. Schembra, "Design of a 5g network slice extension with mec uavs managed with reinforcement learning," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 10, pp. 2356–2371, 2020.

[8] Y. Jararweh, M. Al-Ayyoub, E. Benkhelifa, M. Vouk, A. Rindos *et al.*, "Software defined cloud: Survey, system and evaluation," *Futur. Gener. Comp. Syst.*, vol. 58, pp. 56–74, 2016.

[9] Y. Xiao, N. Zhang, W. Lou, and Y. T. Hou, "A survey of distributed consensus protocols for blockchain networks," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 2, pp. 1432–1465, 2020.

[10] Z. Ding, B. Yang, Y. Chi, and L. Guo, "Enabling smart transportation systems: A parallel spatio-temporal database approach," *IEEE Trans. Comput.*, vol. 65, no. 5, pp. 1377–1391, 2015.

[11] M. Erdelj, E. Natalizio, K. R. Chowdhury, and I. F. Akyildiz, "Help from the sky: Leveraging uavs for disaster management," *IEEE Pervasive Comput.*, vol. 16, no. 1, pp. 24–32, 2017.

[12] X. Cao, J. Xu, and R. Zhang, "Mobile edge computing for cellular-connected uav: Computation offloading and trajectory optimization," in *Proceedings of the 2018 IEEE 19th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*. IEEE, Jun. 2018, pp. 1–5.

[13] B. Di, L. Song, and Y. Li, "Sub-channel assignment, power allocation, and user scheduling for non-orthogonal multiple access networks," *IEEE Trans. Wireless Commun.*, vol. 15, no. 11, pp. 7686–7698, 2016.

[14] C. Qiu, Y. Hu, Y. Chen, and B. Zeng, "Lyapunov optimization for energy harvesting wireless sensor communications," *IEEE Internet Things J.*, vol. 5, no. 3, pp. 1947–1956, 2018.

[15] Y. Mao, J. Zhang, and K. B. Letaief, "Dynamic computation offloading for mobile-edge computing with energy harvesting devices," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 12, pp. 3590–3605, 2016.