SkyCore: Moving Core to the Edge for Untethered and Reliable UAV-based LTE Networks

Mehrdad Moradi*
University of Michigan

Karthikeyan Sundaresan
NEC Labs America

Eugene Chai
NEC Labs America

Sampath Rangarajan
NEC Labs America

Z. Morley Mao
University of Michigan

ABSTRACT

The advances in unmanned aerial vehicle (UAV) technology have empowered mobile operators to deploy LTE base stations (BSs) on UAVs, and provide on-demand, adaptive connectivity to hotspot venues as well as emergency scenarios. However, today's evolved packet core (EPC) that orchestrates the LTE RAN faces fundamental limitations in catering to such a challenging, wireless and mobile UAV environment, particularly in the presence of multiple BSs (UAVs). In this work, we argue for and propose an alternate, radical *edge* EPC design, called SkyCore that pushes the EPC functionality to the extreme edge of the core network – collapses the EPC into a single, light-weight, self-contained entity that is co-located with each of the UAV BS. SkyCore incorporates elements that are designed to address the unique challenges facing such a distributed design in the UAV environment, namely the resource-constraints of UAV platforms, and the distributed management of pronounced UAV and UE mobility. We build and deploy a fully functional version of SkyCore on a two-UAV LTE network and showcase its (i) ability to interoperate with commercial LTE BSs as well as smartphones, (ii) support for both hotspot and standalone multi-UAV deployments, and (iii) superior control and data plane performance compared to other EPC variants in this environment.

KEYWORDS

5G; UAV; LTE; EPC; Edge EPC; UAV-based LTE networks; SDN; Drone; Resource-Challenged; Mobility Management

ACM Reference Format:
Mehrdad Moradi, Karthikeyan Sundaresan, Eugene Chai, Sampath Rangarajan, and Z. Morley Mao. 2018. SkyCore: Moving Core to the Edge for Untethered and Reliable UAV-based LTE Networks. In *The 24th Annual International Conference on Mobile Computing and Networking (MobiCom '18), October 29-November 2, 2018, New Delhi, India.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3241539.3241549

*The contact author: moradi@umich.edu

MobiCom '18, October 29-November 2, 2018, New Delhi, India

ACM ISBN 978-1-4503-5903-0/18/10…$15.00
https://doi.org/10.1145/3241539.3241549

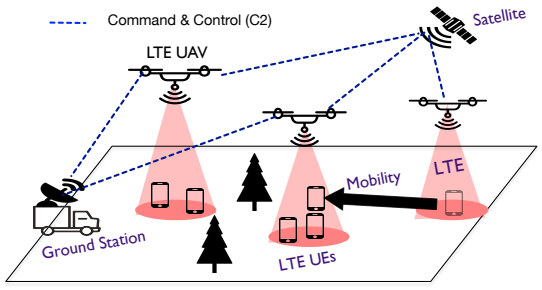


Fig. 1: UAV-based LTE networks.

1 INTRODUCTION

LTE networks that are ubiquitous today are deployed after sufficient RF planning in a region. However, the static nature of LTE base station (BS) deployments limits their ability to cater to certain key 5G use cases – surging traffic demands in hotspots (e.g., stadiums, event centers), as well as their availability in emergency situations (e.g., natural disasters), where the infrastructure could itself be compromised (section 2). Providing an additional degree of freedom for base stations, namely *mobility*, allows them to break away from such limitations.

UAV-driven LTE networks. In this regard, recent advances in unmanned aerial vehicle (UAV) technology have empowered operators to take on-demand, outdoor connectivity to another level, by allowing their base stations to be deployed aerially on UAVs (Fig. 1), thereby offering complete flexibility in their deployment and optimization. Mobile operators like AT&T and Verizon have both conducted trials with LTE base stations mounted on UAVs [15, 16] (helicopter andfi xed-wing aircraft respectively, Fig. 3). AT&T also provided LTE network services from its UAV in the aftermath of hurricane Maria in Puerto Rico last year [7]. Further, with the availability of shared access spectrum like CBRS [4] in 3.5 GHz, this also opens the door for smaller, greenfield operators to deploy and provide on-demand, private LTE connectivity services without the heavy cost associated with spectrum and deployment.

Limitations of the legacy EPC. A typical LTE network requires the deployment of two essential components: a radio access network (RAN) consisting of multiple base stations (BSs) that provide wide-area wireless connectivity to clients

(UEs), and a high-speed, wired core network of gateways (evolved packet core, EPC) that sits behind the RAN and is responsible for all the mobility, management and control functions, as well as routing user traffic to/from the Internet. Realizing a multi-UAV-driven RAN (BSs deployed on UAVs) with an *EPC on the ground or in the cloud* is one way to directly apply today's EPC architecture to the UAV environment (as shown in Fig. 5). Based on publicly available information [5, 7, 14–16], this has been the case with the current operator-driven UAV efforts. However, this faces significant limitations in delivering real value to this challenging environment. Specifically, while a **tethered setup** (EPC-UAV link being wired, Fig. 5a, and possibly in Fig. 3a) significantly *limits the UAV's mobility and ability to scale to multiple UAVs*, a **wireless setup** (EPC-UAV link being wireless/mobile, Figs. 3, 5b) incurs all the *vagaries of the wireless channel*. For the latter, the choice of the wireless technology becomes critical given that the EPC is responsible for setting up, routing, and tearing down all voice/data bearers. It is essential for the EPC to *reliably reach all the UAVs* wirelessly, including those that are potentially far away in the presence of non-line-of-sight conditions (e.g., buildings, foliage, etc.). Further, it must deliver sufficient capacity to support the traffic demands in the RAN. It is extremely challenging for a wireless technology, be it lower frequency (sub-6 GHz like LTE, WiFi, and etc.) or higher frequency (mmWave, satellite), to simultaneously satisfy the *needs of range, reliability/robustness, and capacity* that the UAV environment demands from the critical EPC-RAN link.

Core at the Edge. Given the fundamental limitations in deploying an EPC on the ground or in the cloud to support a multi-UAV RAN, we advocate for a radical, yet standards-compliant re-design of the EPC, namely the *Edge-EPC* architecture, to suit the UAV environment. As the name suggests, we aim to push the *entire* EPC functionality to the extreme edge of the core network, by collapsing and locating the EPC as a single, light-weight, self-contained entity on each of the UAVs (BSs) as shown in Fig. 7. Being completely distributed at the very edge of the network, such an architecture completely eliminates wireless on the critical EPC-RAN path and hence the crippling drawbacks faced by the legacy architecture in this environment.

While definitely promising at the outset, realizing this radical design is not without its own set of challenges that are unique to the UAV environment. In particular, (i) *Resource-challenged environment:* The compute resources consumed by the numerous network functions in the EPC is appreciable and becomes a concern when all the EPC functionality is placed into a single node, and deployed directly on a UAV platform – the latter being highly resource-challenged to begin with. This could significantly affect both the UAV's operational lifetime as well as the processing (control and data plane) latency of its traffic, thereby resulting in a reduced traffic capacity. (ii) *Mobility management:* The hierarchical nature of the legacy EPC architecture, gives a single network gateway (like the mobility management entity, the MME) a consolidated view of multiple BSs, thereby allowing it to efficiently manage handoffs during mobility of active UEs as well as tracking/paging mobile UEs that are in idle mode. Mobility of both active (handoffs) and idle UEs (tracking/-paging) becomes a critical challenge, when the entire EPC is located at each of the UAVs, thereby restricting their view of events to only those that are local to the UAV.

Our Proposal – SkyCore. Towards our vision of building *untethered yet reliable UAV-based LTE networks*, we present our novel EPC design, SkyCore. SkyCore embodies the Edge-EPC architecture while introducing two key pillars in its design to address the associated challenges – a complete *software refactoring of the EPC* for compute-efficient deployment on a UAV, and a new *inter-EPC communication interface* to enable fully functional operation in a multi-UAV environment. Through *software refactoring*, SkyCore eliminates the distributed EPC interfaces and collapses all distributed functionalities into a single logical entity (agent) by transforming the latter into a series of switchingfl ow tables and associated switching actions. It also reduces control plane signaling and latency by precomputing and storing (in-memory) several key attributes (security keys, QoS profile, etc.) for UEs that can be accessed quickly in real time without any computation. To ensure complete EPC functionality, SkyCore manages mobility right at the edge of the network – it enables a new control/data interface through *software-defined networking (SDN)* to realize efficient *inter-EPC signaling and communication* directly between UAVs. This allows the SkyCore agents on each UAV to *proactively* synchronize their states with each other, thereby avoiding the real-time impact of wireless (UAV-UAV) links on critical control functions – results in fast and seamless handoff of active-mode UEs as well as tracking of idle-mode UEs across multiple UAVs. Our design decisions are efficient and scalable for airborne LTE networks consisting of a few to at most tens of UAVs.

Real-world prototype: We have built a complete version of SkyCore on a single board server with a small compute and energy footprint, and deployed it on DJI Matrice 600 Pro rotary-wing drones to create a two-UAV LTE network. To the best of our knowledge, this is thefi rst realization of a self-contained Edge-EPC solution that can support a multi-UAV network and is a direct affirmation of SkyCore's design. SkyCore's feasibility and functionality are validated by seamless integration and operation with a commercial LTE RAN (BS) from ip.access and off-the-shelf UEs (Moto G and Nexus smartphones). We demonstrate SkyCore UAVs to operate both as LTE hotspots that allow for better UE connectivity to the Internet by extending coverage of a terrestrial LTE network, as well as standalone LTE networks for connectivity of geographically separated UEs through two different UAVs (e.g.,fi rst responders in emergency scenarios), while also allowing for handoffs. Our real-world evaluations of SkyCore and its comparison with a state-of-the-art software EPC (OpenEPC [12]) on UAV clearly showcases SkyCore's

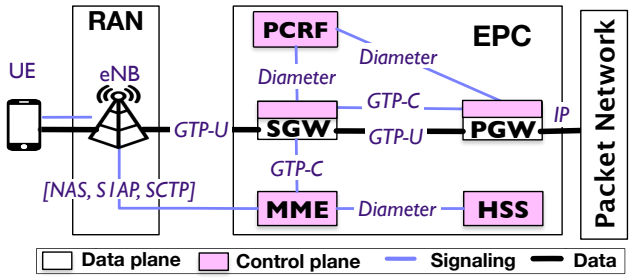


Fig. 2: Legacy EPC architecture.

superior performance and scalability – SkyCore provides an order of magnitude lower control plane latencies, incurs 5× lower CPU utilization, and provides data plane rates that currently scale up to a Gbps.

Our two key contributions in this work include,

- A novel Edge-EPC solution, SkyCore that can *reliably and scalably* support a multi-UAV LTE network deployment that was not possible earlier.
- A real-world implementation and evaluation that showcase both its feasibility and its superior performance.

Broader implications: SkyCore's underlying design is driven by the observation that when connectivity between core network functions, which are on the critical path, is unreliable (wireless and mobile), the merits of pushing functionality to the edge of the network significantly outweighs the associated drawbacks. Hence, although designed for a multi-UAV environment, SkyCore's design can also benefit other deployments, where distributed critical functions have to communicate over unreliable links (e.g., distributed enterprise RANs).

2 MOTIVATION

(a) (b)

Fig. 3: (a) AT&T's and (b) Verizon's Cell on Wings (CoW). Both adopt the legacy EPC architecture with wireless connectivity (e.g., using a satellite relay or directly based on other technologies) between UAVs (eNBs) and the EPC deployed on the ground (e.g., on trailers hosting the core [3]) or in their cloud network. AT&T used a tether to provide a fiber data connection (possibly to support a wired EPC on the ground if available) and power to the UAV for unlimitedfl ight time. In Verizon's tests, there was no tether to the ground but the EPC was still on/in the ground/cloud.

2.1 UAV-based LTE Networks

We consider low-altitude UAV networks, such as those considered by mobile operators [15, 16] for on-demand LTE network deployments. These are envisioned to provide temporary LTE coverage from the sky to a designated area on the ground. One use case is to enhance connectivity at big events (e.g., music festivals, football tournaments) by creating a *hotspot* UAV-based LTE network extending capacity/-coverage of local terrestrial LTE networks (macrocells). The second use case is to establish a *standalone* UAV-based LTE network forfi rst responders and the general public battling a disaster where no infrastructure is available. Such a network can keep them connected through local group communication and peer-to-peer (P2P) services, e.g., push-to-video.

The relevance of our work to high-altitude, long-endurance platforms like Google's Loon [8] and Facebook's Aquila [6] is discussed in section 9. In a UAV-based LTE network, an LTE BS (eNB) is directly deployed on each UAV, and multiple of them together provide wireless connectivity to UEs over a desired wide area as shown in Fig. 1. However, not much thought has been paid towards the deployment of an EPC to support such a RAN. Deploying and managing a traditional LTE EPC is a challenge in its own right. Designing one to support an LTE RAN on UAVs, which are highly restrictive in their compute capabilities, endurance and payload capacity, further amplifies the associated challenges.

To foster a better understanding, wefi rst provide a primer on EPC's key functionality, followed by the limitations of today's EPC for our target environment, and the benefits and drawbacks of an "alternate" Edge-EPC architecture.

2.2 EPC Primer

Fig. 2 shows the network architecture of the EPC, which is a distributed system of different nodes, each consisting diverse network functions (NFs) that are required to manage the LTE network. The EPC consists of data and control data planes: the data plane enforces operator policies (e.g., DPI, QoS classes, accounting) on data traffic to/from the user equipment (UE), while the control plane provides key control and management functions such as access control, mobility, and security management. eNodeBs or eNBs (RANs) are grouped into logical serving areas and connected to serving gateways (SGWs). The SGW is connected to an external packet network (e.g., the internet) via the packet data network gateway (PGW). The PGW enforces most of the data plane policies (e.g., NAT, DPI) and may connect the core to other IP network services (e.g., Web servers). The EPC forwards each UE's data traffic between the eNodeB and PGW using a separate GTP-U (GPRS tunneling protocol) tunnel. The mobility management entity (MME) is responsible for access control, security and mobility functions (e.g., attach/detach, paging/handoff) in conjunction with the home subscriber server (HSS) database.

2.3 Limitations of Legacy EPC Architecture

The straightforward way to apply the EPC to our UAV network would be to collapse all the EPC network functions into a single node (EPC-in-a-box) and deploy this EPC node on a resource-capable machine on/in the ground/cloud that can support multiple UAV BSs. Based on publicly available information [5, 7, 14–16], this is the approach adopted by operators like AT&T and Verizon in their recent trials (Fig. 3).

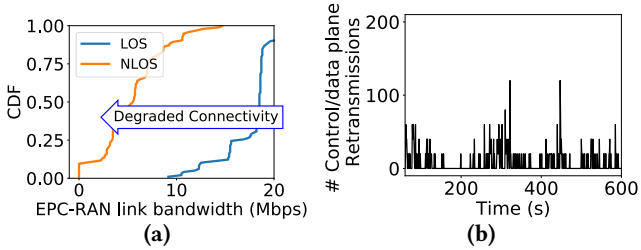


Fig. 4: (a) Degraded throughput on the EPC-RAN link (10 MHz LTE link) when an LTE UAVfl ies in LOS and NLOS (over a building) trajectory with a 3GPP-compliant EPC being on the ground (detail on the EPC and RAN in section 6). (b) # of SCTP/TCP (signaling/data) retransmissions in NLOS.

2.3.1 Tethered Deployment (Wired EPC-UAV link). In today's traditional LTE networks, the connectivity between the EPC and eNBs (RAN) is a reliable, wired network provisioned with sufficient bandwidth for catering to the UE traffic demands in both downlink and uplink. A similar approach can be adopted in our UAV network, where the RAN runs on the UAV, which is tethered by a wire to a ground station running the EPC (Fig. 5a and possibly in Fig. 3a). However, such an approach significantly limits the potential andfl exibility of the UAV tofl y and re-position itself to cater to network traffic requirements, not to mention the associated safety concerns and the infeasibility of scaling such a setup to support a network of UAVs. With UAV technology advancing at a rapid pace to provide longer operational times [1], such a tethered EPC-on-ground does not offer a viable, future-proof solution.

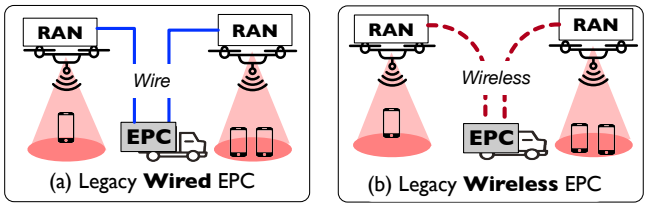


Fig. 5: Legacy EPC variants for UAV-based LTE networks.

2.3.2 Un-tethered Deployment (Wireless EPC-UAV link). The other alternative is where the connectivity between the EPC and eNBs (UAVs) is wireless (Figs. 3, 5b). In this design, the EPC could be placed on the ground (e.g., on trailers hosting the core [3]) or in the operator's cloud. The backhaul between UAVs and the EPC could be routed through satellite (in case of the cloud or ground EPC) or another wireless technology, e.g., mmWave, directly between UAV and EPC (in case of the ground EPC). While relying on the satellite backhaul is an expensive option that may not scale to large and particularly time-sensitive traffic demands, the ground-EPC option also has its share of drawbacks that we elaborate.

Reliability vs. range vs. capacity: The wireless channel is inherently an unreliable medium, and is subject to wireless artifacts such as shadowing (building, trees, obstacles), multipath fading, etc. that can significantly degrade signal quality (by as much as 70% in our experiments, Fig. 4a) and cause high packet retransmissions (more than 100 SCTP/TCP retransmissions, Fig. 4b) and potentially cause disconnections. The choice of the wireless technology also plays an important role. Using lower frequencies like 700 MHz, 1 GHz, etc. allows for better penetration and hence longer communication ranges and better reliability but significantly lesser bandwidth (capacity of few tens of MHz). In contrast, higher frequencies like mmWave (28 GHz, 60 GHz, etc.) offer significantly more bandwidth (hundreds of MHz to a GHz) but suffer from higher attenuation and hence lower range. While the latter can employ beamforming to cope with attenuation, they are limited by line-of-sight requirements and the need to constantly track the beam direction with respect to each UAV as they move – impediment for reliable operation in low-altitude deployments. Thus, it is extremely challenging to identify a wireless modality for the critical EPC-RAN (ground to UAV) link that can offer the simultaneous features of reliable connectivity, increased communication range, and capacity, that is warranted by this EPC deployment.

Single-point bottleneck: The EPC node on the ground becomes the routing focal point that ferries traffic not only between UEs and the Internet but also between UEs within the UAV network. Hence, even if the UAV backhaul (connectivity between UAVs) is well-provisioned, having a small set of ground EPC nodes, concentrates all the traffic on the UAV backhaul towards these ground nodes, which in turn become the bottleneck. This would significantly degrade the capacity of the network as a whole. For a low-altitude UAV network deployed to provide on-demand connectivity to a small geographic region, the bulk of the traffic *might be* local – e.g., between users and content servers in events, or betweenfirst responders and/or affected people in emergencies. In such scenarios, incurring the wireless capacity bottleneck due to the EPC on the ground is unwarranted.

A simple illustration in Fig. 6 shows that the capacity offered by an EPC-on-ground architecture even for a small 4-UAV network can be 6× lower than if the local traffic were to be served directly between the UAVs. Assume the capacity of the EPC-UAV link (shared among all UAVs) is X and similar to that of each UAV-UAV link. If there are N user sessions between every pair of UAVs, each having bandwidth request N/X, the EPC-on-ground can satisfy $N/6$ of the sessions as opposed to the whole in the distributed EPC version.

In addition, UAV and UE mobility is highly pronounced in these networks, which also leads to increased control signaling and associated latency over multiple wireless hops between the ground EPC node and the UAVs.

One option is to deploy multiple EPC nodes on the ground to allow for more reliable connectivity to all UAVs and to add capacity (akin to provisioning multiple gateways in wireless mesh networks [17]). However, this adds to both the cost as well as reliance on ground deployments, working against thefl exibility offered by UAVs in thefi rst place.

2.4 Challenges in Edge EPC Architecture

To counteract the challenges in deploying a legacy EPC architecture, we focus our attention on a radically different "edge"

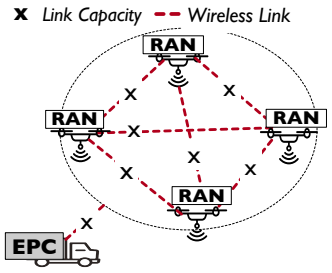


Fig. 6: The capacity bottleneck in the legacy wireless EPC.

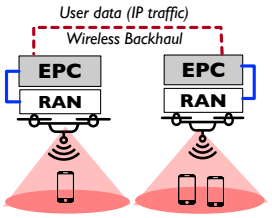


Fig. 7: Edge-EPC for UAV-based LTE networks.

EPC architecture (referred to as Edge-EPC). Here, the *entire* EPC is collapsed and located as a single, self-contained entity on each of the UAVs as shown in Fig. 7. Being completely distributed at the edge of the network, such an architecture would completely eliminate the crippling drawbacks faced by the previous architecture resulting from wireless connectivity between the EPC and eNBs (UAVs). [1] [2]

While the Edge-EPC idea is definitely promising at the outset, it does encounter a different set of challenges in its realization.

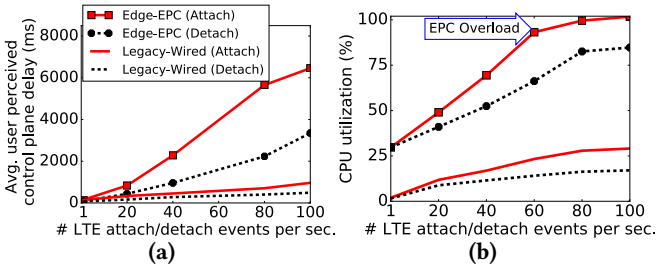


Fig. 8: Edge-EPC has high overheads on UAVs and results in performance bottlenecks and degraded user experience.

2.4.1 Resource-challenged. An EPC consists of multiple network functions along with the complex interfaces and tunneling protocols between them (Fig. 2). Further, most of these are stateful network functions and consist of both control and data plane functionality. These network functions, which used to be deployed by operators on specialized hardware, are now slowly migrating to a virtualization environment with the recent advances in NFV (network function virtualization [11, 18, 45]). Nevertheless, the compute resources consumed by these network functions is appreciable and becomes a concern when all the EPC functionality is collapsed onto a single node. Deploying an EPC node on the

[1] In Edge-EPC, the hotspot use case is realized by connecting the local EPC hosted on a subset of UAVs that are in proximity of macrocells to the Internet. To create a standalone network, content servers are co-located with and attached to the local EPC on a desired subset of UAVs. In both modes, the EPC-RAN (EPC-UAVs) link is highly reliable (being co-located on the same UAV), unlike the drawbacks experienced by the legacy wireless EPC architecture with a wireless EPC-RAN link.

[2] Implementing the EPC functionality at macrocells is not a favorable option for realizing Edge-EPC. Due to reliance on thefi xed infrastructure, it cannot be applied to the standalone use case (e.g., deployment in remote areas). Similarly, it is not viable for the hotspot use case due to lack of compute resources for running EPC on cell towers.

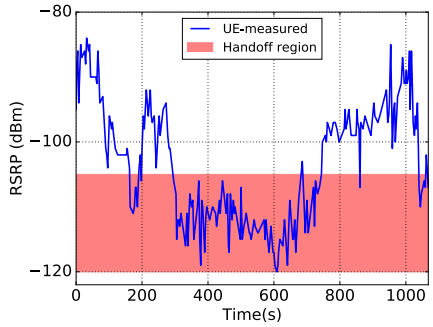


Fig. 9: Edge-EPC fails in seamlessly handling increased handoffs in our LTE UAV environment.

UAV could significantly affect both its operational lifetime as well as the processing (control and data plane) latency of its traffic, thereby resulting in a highly reduced traffic capacity. This can be observed in Figs. 8a, 8b, where the latency and CPU utilization of control plane functions can be an *order higher* in Edge-EPC, when the platform (such as that on a UAV) is resource-constrained (experimental details in section 6). Given the EPC is a complex distributed system, identifying the key sources of resource consumption in the EPC and to the extent that each source contributes to resource consumption is out of the scope of this work. We refer the reader to previous works reporting other relevant issues in the EPC [23, 30, 36].

2.4.2 Handling Mobility at the Edge. The conventional EPC has a hierarchical structure, where a single PGW spans multiple SGWs, and a single SGW spans multiple eNBs. As the UE (in active mode) moves from one eNB to another (handoff), this is handled locally by its SGW. Further, every UE has a tracking area (TA, set of neighboring eNBs) associated with it, which the EPC will use to page (all eNBs in its TA) to locate it when in idle mode. When the UE moves out of its current TA, it notifies the EPC of its updated TA. Thus, UE mobility is handled seamlessly in the legacy EPC.

Active-mode mobility (Handoffs): Network dynamics in the form of UE and/or UAV mobility forms a significant part of our operating environment. However, with the collapse of the hierarchical architecture in Edge-EPC, one needs to now enable communication between the EPC entities on individual UAVs to enable seamless handoff across UAVs. In today's mobile networks, a UE hardly moves across different PGWs within the same operator's network (a single PGW spans a significantly large area - hundreds of miles). When such an event does happen, the connection is terminated with the existing PGW and re-established with the new PGW causing service disruption. However, such events are the norm rather than an exception in our environment. Fig. 9 illustrates the number of potential handoff events that can be triggered due to appreciable signal strength variations even during a short UAVfl ight (less than 20 m) in our experiments. Each UE-measured Reference Signal Received Power (RSRP) value in the shaded region by itself can trigger a handoff event in the network if another UAV that delivers sufficiently

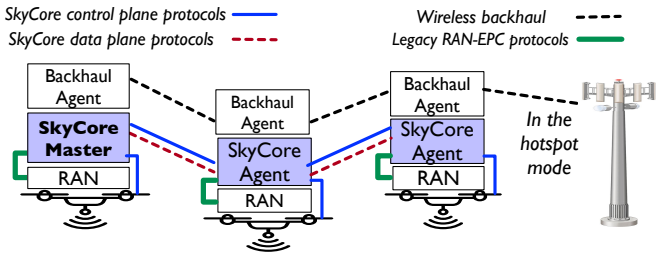


Fig. 10: The SkyCore network architecture.

higher RSRP is available. Hence, it becomes critical to enable seamless EPC-EPC communication for handling mobility in the Edge-EPC architecture. This is needed to also handle UAV mobility, i.e. when one UAV goes down for a re-charge and is replaced by another UAV – a migration of state from one UAV (EPC) to another is imperative.

Idle-mode mobility (Tracking/paging): With the ability to page idle UEs over large tracking areas (spanning several BSs), it is fairly straightforward to locate any UE in the network in the legacy EPC. This is, however, a challenge for the Edge-EPC architecture, where there is no single PGW that spans all the UAVs (eNBs). Further, since the notion of tracking area disappears (due to collapsed EPC), locating a UE when in idle mode appears to be infeasible, prompting the need for new or adapted mobility mechanisms.

3 SKYCORE: DESIGN OVERVIEW

SkyCore adopts the Edge-EPC architecture as shown in Fig. 10. SkyCore collapses the entire EPC and pushes it to the edge of our network, namely at each of the UAVs themselves, where it is co-located with the RAN. While this completely eliminates wireless from the critical path between the EPC and RAN, to address the challenges associated with the Edge-EPC architecture, SkyCore introduces two novel design components. Briefly,

Software refactoring of the EPC functionality: To reduce its compute footprint on the UAV, SkyCore adopts a software refactoring approach to eliminate distributed EPC interfaces and collapse all distributed functionalities (Fig. 2) into a single logical entity. It realizes this by transforming the distributed data plane functions into a series of switching flow tables and associated switching actions (corresponding to functions like GTP-U encapsulation/decapsulation, charging, etc.). It also reduces control plane signaling and latency by precomputing and storing (in-memory) several key attributes relating to security keys, QoS profile, etc. for UEs that can be accessed locally in real time without any computation.

Efficient inter-EPC communication: With every UAV now running its own EPC agent, even a simple eNB-eNB handoff of an active UE across two UAVs now becomes an inter-MME handoff, which needs to be accomplished across two different EPC agents. SkyCore enables a new control/-data interface that allows agents on different UAVs to *proactively* (in the background) synchronize the state of UEs. This bypasses the real-time impact of wireless (UAV-UAV links) on critical control path functions, allowing for seamless handoffs and tracking of idle-mode UEs right at the edge. The HSS equivalent in each SkyCore agent maintains the location (anchoring SkyCore agent) of all UEs in the network. Hence, when an agent sends a UE location update, the agents in other UAVs update their HSS accordingly. Thus, whenever traffic needs to be sent from a SkyCore agent to a UE located at another UAV, the HSS will reveal the destination SkyCore agent at which the UE is anchored and to whom the traffic has to be routed. The actual routing path to be taken by the traffic on the mesh backhaul is then determined by SkyCore, with the underlying backhaul topology information made available by a backhaul agent that resides on the UAV[3].

We now explain each of these design components in detail.

4 SOFTWARE REFACTORING OF EPC

4.1 Minimalistic SkyCore Agent Architecture

Each SkyCore agent has a minimalist and UAV-aware SDN-based architecture (Fig. 11), consisting of a controller that executes the control functions to process UEs' signaling traffic and to coordinate with other agents, and a switch that processes user data traffic. In the following, we describe six high-level steps that we take to refactor and extend the EPC functionality onto our agent architecture.

Step 1. Decoupling the EPC control and data plane pipelines. One of the main reasons behind the high complexity and overhead of the EPC is its nodes performing mixed control and data plane functions. To make the EPC functionality suitable for UAVs, wefi rst decouple the EPC control and data planes. Among the EPC nodes, the MME, PCRF, and HSS are pure control nodes. Hence, our decoupling does not affect these elements, and only affects the SGW and PGW. The resulting control components from the decoupling are the PGW-C, SGW-C, MME, PCRF, and HSS, and the data elements include the SGW-D and PGW-D (C stands for control and D for data). While the benefits of decoupling control and data planes have been articulated before [28, 34], we apply it in the context of UAV networks and enhance it substantially with the following mechanisms.

Step 2. Categorizing the functionality of the EPC control plane. Next, we categorize the EPC control nodes based on their high-level functionality. In our decoupled EPC, there are three types of nodes: (1) the SGW-C and PGW-C are responsible for managing QoS policy enforcement on and routing of user data traffic, (2) the MME exchanges signaling traffic with UEs and eNBs, and (3) the PCRF and HSS dynamically generate network security and QoS policies for the other nodes. To compress the EPC functionality, we consolidate the nodes in each category on top of our agent controller and remove the EPC distributed protocols as follows.

[3]The design of the backhaul agents responsible for maintaining a well-provisioned, connected wireless mesh topology is outside the scope of this work.

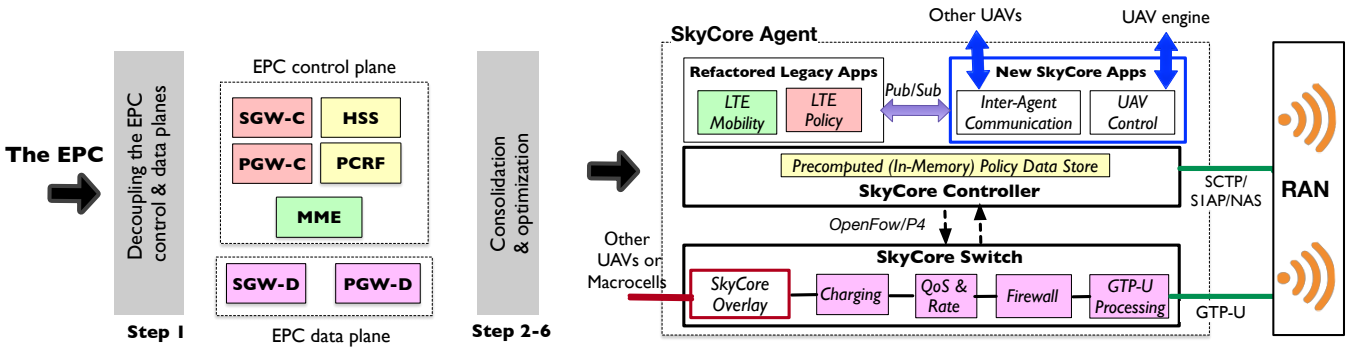


Fig. 11: SkyCore refactors the EPC functionality into a lightweight software-driven agent having new interfaces for interaction with the local UAV and other UAVs.

Step 3. Collapsing the SGW-C, PGW-C, and MME into light-weight SDN applications. We extract the internal functions in the SGW-C and PGW-C and refactor them into a single SDN application, LTE Policy Application, on top of the controller. We do the same process for the MME and transform it into LTE Mobility Application. One notable aspect of this consolidation is that we naturally eliminate the complex GTP-C protocol, its six interfaces, and continuous control messages from the core network (Fig. 2). This makes the SDN applications extremely lightweight and extensible without hurting their original functionality. Note that these applications still exchange information with each other but through simple local publish-subscribe mechanisms.

Step 4. Eliminating the HSS and PCRF from the core and replacing them with a precomputed policy data store. Next, we focus on the HSS and PCRF that are known to be the source of today's signaling storms in cellular networks [10, 13]. The HSS stores hundreds of database tables containing different UEs' states often on disk. Moreover, it acts as a proxy between the MME and these tables, and performs different types of complex security and location tracking computations. The PCRF often accesses a logical database (sometimes implemented in the HSS) and dynamically generates different QoS and charging policies for UEs. In SkyCore, we completely eliminate these two nodes from our agents and show that dynamic policy generation can be carefully replaced with a precomputed in-memory policy data store. Precomputation combined with in-memory transactions substantially minimizes the overhead of the core on resource-challenged UAVs (elaborated in section 4.2). This also removes the complex Diameter protocol (Fig. 2) from the core.

Step 5. Adding UAV-specific SDN applications to the core. One of the key differences between SkyCore and the traditional EPC is in its continuous interaction with the UAV hardware and its APIs. In particular, we advocate for two new applications on top of our agents. Each SkyCore agent runs UAV Control Application that listens tofl ight change events from UAV and remaining battery resources on the UAV. This is necessary for our agents to properly handoff UEs to each other, e.g., when a UAV needs to immediately leave the network for recharging. Such use cases clearly show the potential of our SDN-based UAV-aware architecture. In addition, we design an Inter-Agent (UAV) Communication Application (section 5) that exchanges control plane messages with its neighbor agents to synchronize states *proactively*, thereby enabling seamless mobility (active and idle). The legacy EPC applications and new SkyCore core applications that need to exchange information with each other, do so through our local publish-subscribe protocols.

Step 6. Replacing the hierarchical data plane gateways with a compact SDN switch. Since SkyCore is aflat architecture, it eliminates the need for hierarchical gateways on each UAV. To further make our agents compact, we refactor the SGW-D and PGW-D functionality into a single software switch. Each data plane function in S/PGW-D is implemented as a separate Match+Action table in this software switch. Each table performs a lookup on a subset of user's data trafficfi elds and applies the actions corresponding to thefi rst match. Users' traffic travels through these tables before leaving or entering the UAV. In particular, our software switch performs UL/DL data rates enforcement, stateful firewall operations, and QoS control by transport-level mechanisms (e.g., setting DiffServ) based on QoS class identifier (QCI) associated with each UE. While the legacy EPC tunnels each UE's traffic into two tunnel segments across the RAN, PGW-D, and SGW-D, SkyCore departs from this approach and terminates GTP-U tunnels inside our agent switch (decapsulates GTP-U header from uplink packets sent by the eNB and encapsulates downlink packets to the eNB into a proper GTP-U header) for two reasons. First, per-UE tunnels do not scale in LTE UAV networks as UEs are mobile and these tunnels are subject to frequent changes. Second, our consolidation already eliminates the need for complex GTP-U tunnels between the SGW-D and PGW-D functionality.

4.2 SkyCore Precomputed Policy Data Store

We now describe how the HSS/PCRF can be replaced with precomputed network policies on SkyCore agents. As shown in Fig. 12b, the SkyCore data store associates each UE's IMSI (International Mobile Subscriber Identity) information with its precomputed policies, which can be quickly accessed by different applications on the agent. Our precomputation approach not only reduces the user-perceived network access delay but it also makes the core extremely resource-efficient.

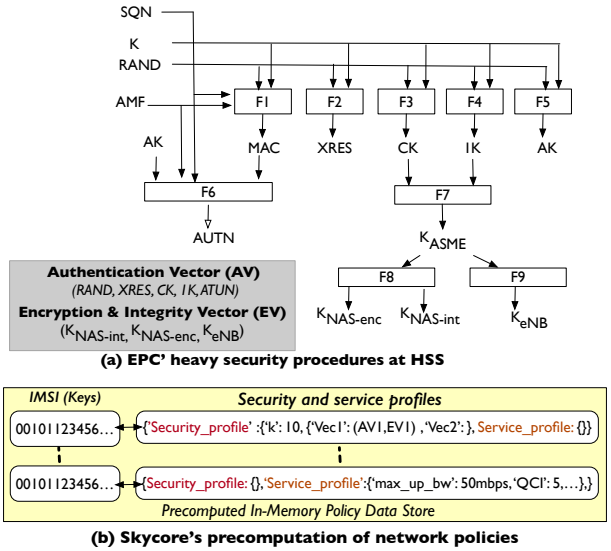


Fig. 12: SkyCore's precomputation of network policies not only makes the core resource-efficient but also minimizes network access delay.

Note that our target deployment area spans a few tens of square miles and each operator knows the last location of each UE in its network (stored in the HSS). Thus, performing the policy precomputation for the UEs located in a superset of the deployment area is sufficient. In rare scenarios, where a valid UE (coming from outside of the superset) tries to connect to a UAV, we can fall back to online computation at the local agent and then update the other agents in the network to avoid future online computation for the same UE. In section 5, we will discuss how online policy computation and policy changes by an agent are propagated to other agents in the network to ensure their state is consistent.

4.2.1 Precomputation of Security policies. Network Access Security in LTE networks relies on the shared user-specific key, K, that is stored in the HSS and UE SIM cards. The LTE security processes assume that cloned UEs and spoofed networks do not know the correct value of K. From K, the HSS dynamically computes (shown in Fig. 12a) an authentication vector (AV) and an encryption vector (EV) as part of a larger LTE attach process, when UEs switch on or enter an area with LTE coverage. The EPC and the UE confirm each other's identities using the AV. The signaling traffic between the UE and the network is encrypted using the EV to ensure intruders cannot read and modify them. The computation of these vectors involves resource-intensive cryptographic operations (F1-F9) on 256-bit long strings, thereby wasting valuable clock cycles on UAVs.

Offline computation of security vectors. When there are a few UEs, the overhead of computing such security vectors on UAVs is manageable. However, when UAVs are providing on-demand LTE connectivity over a large geographical region, many UEs are likely to send LTE attach requests to the network at the same time. Such realistic workloads can quickly use the available compute resources (for the core) on UAVs and substantially degrade the QoE of users. To resolve these issues, our key idea is to depart from real-time security vector computations on UAVs. We precompute and store a reasonable number of security vectors for each UE and store them on the SkyCore agent. All these vectors are computed with the same K, and $RAND$ (a random number generated by the HSS) but with different consecutive sequence numbers (SQN). Different SQN numbers ensure signaling messages cannot be replayed by intruders.

Since each pair of vectors is computed with a different SQN number, it can be used only once by SkyCore during the LTE attach procedure. If the same pair is reused, the UE will reject the network assuming it is a spoofed network that is trying to replay old authentication messages. Thus, each of our agents locally removes a *used* pair of security vectors and invalidates it at other agents through our Inter-Agent Communication application (see Fig. 11 and section 5). Note that the number of attach requests generated by a legitimate UE, when it switches on or comes back into network coverage, is limited. Hence, SkyCore precomputes a small number of such vectors for a UE. In rare cases, when a UE uses all its precomputed vectors (e.g., due to frequent restarts), SkyCore agents fall back to computing new vectors for such UEs in real time and propagate them to other agents in the network. Tofi nd the best configuration that simultaneously minimizes the number of online computations and storage needed for precomputed vectors at UAVs, a SkyCore operator can look at histograms of the number of LTE attach/detach events [28] for each UE in its network.

4.2.2 Precomputation of Service policies. In LTE networks, the PCRF dynamically generates quality of service (QoS) and charging rules for a UE. The PCRF continuously feeds the PGW and SGW with real-time QoS rules. Rather than generating these rules in real time by accessing many different tables, we precompute the entire rule set that must be applied to UEs' traffic, and consolidate and store them onto our agents. In particular, SkyCore consolidates three types of rules that deal with (i) *QoS* (bit rate, loss rate, etc.), (ii) *priority* (flow handling during congestion), and (iii) *charging* (offline, online and time-dependent). Most of these service policy rules are almost always static (e.g., max. bit rate/loss rate) and do not change over time. Thus, we do not need to synchronize these policies among our agents as UEs move around in the network. However, some of these rules operate on some internal states (e.g., remaining bandwidth in an online charging model). Such dynamic states for different UEs are batched and synchronized periodically among our agents as they are often not time-critical (section 5).

5 EFFICIENT INTER-AGENT COMMUNICATION

5.1 Scalable SDN Control and Data Overlays

SkyCore agents seamlessly exchange control and data traffic with each other, a functionality that is lacking among today's

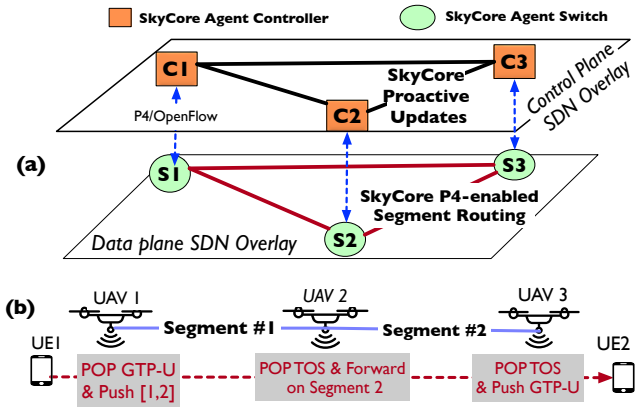


Fig. 13: (a) SkyCore's network-wide control and data plane connectivity for LTE UAV networks. (b) Example of our segment routing for data traffic from UE1 to UE2.

EPC instances. Rather than relying on distributed and multi-hop wireless routing protocols, we choose to adopt an SDN approach in the design of SkyCore to support the traffic exchange between agents. SDN enables us to perform global optimization (e.g., multipath traffic engineering) and offerfine-grained programmability (e.g., to effectively support different QoS classes), which are necessary tools to instantly and efficiently reconfigure the core in response to network dynamics (e.g., UAV departures and arrivals) in our environment. In particular, we leverage SDN overlays to create two virtualized network layers (slices) on top of the physical UAV network (Fig. 13a). One of these network slices is used for control plane traffic between SkyCore agents and the other is for data traffic. Our separation of the control and data traffic ensures time-critical control plane traffic is not affected when the network is saturated. To form the overlays, we use traffic tunneling technologies but depart from existing approaches used in the EPC and SDN-based datacenter (DC) networking [26, 27] since they require frequent changes to the network configuration (will be discussed shortly). We adopt a novel variant of segment-based routing in SkyCore and propose a design for its optimization based on the most advanced capability in SDN, i.e., P4 language [20]. P4 allows us to define new packet headers and packet processing actions for the SDN switch inside our agents to minimize the packet header overhead on inter-UAV links, which is caused by forming the overlays.

Segment-based overlays equipped with global source routing. *Tunneling:* We inter-connect each pair of *neighboring* agents (geographic proximity) using a tunnel defined with a label. Whenever an agent decides to send control or data plane traffic to any other agent in the network, it pushes a stack of labels onto the packets. The top-of-the-stack (TOS) label corresponds to the next tunnel segment that the packet must traverse. Whenever an agent receives a packet from its neighbor, it checks the TOS label from the packet and forwards the packet based on the inner label to its neighbor. There is a master that is responsible for globally computing an efficient label stack that each agent must use to communicate with the other agents. Instead of adding separate MPLS packet headers for each label, SkyCore designs a new packet header based on the P4 language to contain *all* the labels in the stack to reduce overhead. It equips switches with new actions to read the labels at different positions. *Routing:* In SkyCore, one of the UAVs is selected (periodically and randomly) to double up as a master agent that is responsible for global route computation. It periodically collects information from other agents, related to average loss rate and bandwidth on wireless links between different agents (UAVs), remaining battery capacity on UAVs, and the amount of traffic demand between different UAVs. The master agent uses this information to compute and disseminate forwarding rules for routing traffic over the UAV mesh backhaul in the sky. For efficiency, a simplified version of the leader selection algorithm introduced in the LEACH protocol [25] can be used for the master selection in SkyCore. *Proximity-based segments enable scalability:* Note that UAV and UE mobility is common in our environment. Hence, a conventional EPC approach to establishing per-UE tunnels (GTP-U tunnels–see section 2) will require frequent tunnel updates (tearing down, modification, or setting up). Similarly, employing a tunnel between every pair of UAV agents (akin to remote DC-DC tunnels in Google's B4 backbone network [27]) will require updates to a large fraction of the tunnels, even when only a small number of UAVs move. In contrast, most of SkyCore's tunnel segments do not change in such scenarios as they are designed to carry aggregate traffic only between nearby pairs of UAVs. *Example:* A simplified example of our tunneling scheme is depicted in Fig. 13b, transporting traffic from UE1 to UE2 across three UAVs. Note that the source and sink agents perform an additional GTP-U processing, introduced in section 3, to seamlessly eliminate GTP-U from the core.

5.2 Proactive Stateless Mobility Support

SkyCore replaces the notion of centralized HSS and PCRF with a precomputed policy data store replicated at different agents. Hence, it is essential that the UE states and policies be consistent across different agents, particularly during UE mobility. Reactive approaches to consistency management e.g., Distributed hash table (DHT) [46], put wireless (inter-UAV links) on the critical path of control functions. SkyCore avoids this real-time dependence by adopting a *proactive synchronization* of state between agents – each agent proactively broadcasts its changes to UE policies and states to other agents in the network. Such an approach (i) minimizes the control plane delay between agents, particularly in mobility scenarios as the destination agent already knows the latest information about the mobile UE; (ii) enables seamless handoff of active UEs to a neighboring UAV, when the current UAV goes down for a recharge; and (iii) is scalable because the amount of control plane traffic that is broadcasted on inter-agent backhaul links is negligible compared to user data plane traffic among agents (section 7).

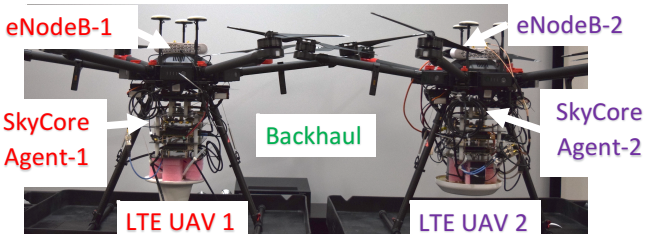


Fig. 14: Multi-UAV SkyCore prototype.

A SkyCore agent needs to send only three types of broadcast update messages in the network to build up a consistent network-wide view: (i) security update to notify other agents that it has used one of the security vectors precomputed for a UE and to request other agents to invalidate the vectors, (ii) location update to inform other agents that a particular UE has attached to its UAV, and (iii) policy update to communicate its local changes to the precomputed QoS and charging profile of a UE.

5.2.1 Idle/Connected-mode Mobility. SkyCore's proactive state synchronization scheme accelerates the handling of increased mobility events in multi-UAV LTE networks.

Idle-mode mobility (Paging). The underlying Edge-EPC design in SkyCore limits each UAV (BS) to its own tracking area (TA). Hence, when an idle-mode UE moves from one UAV to another, it realizes a change in its TA on waking up (prompted by a periodic timer) and sends a TA update request to the SkyCore agent on the new UAV. Since the agent at the new UAV already has the UE's latest policies and states from SkyCore's proactive updates, it knows which security vectors to use for communication with the UE. Hence, it immediately sends a TA update response back to the UE, which can then quickly switch back to its idle mode to continue saving power. In the background, it also broadcasts the updated location of the UE to all other SkyCore agents in the network, eliminating the need for explicit UE paging. This also ensures that the other agents can push the correct label stack on the packets destined for this UE.

Active-mode mobility (Handoffs). Now, consider the UE to instead be in the connected (active) mode during the move. Based on the LTE protocol, it performs continuous signal strength measurements and sends them to thefirst UAV's eNB. If the eNB detects the signal strength of the neighboring second UAV to be stronger, it sends a handoff request message to its SkyCore agent. This agent then notifies the agent on the second UAV of the incoming UE (without having to transfer/update any state on the destination agent) and then confirms the handoff with its own eNB, which then informs the UE. Then, the UE connects to the eNB on the second UAV, whereupon its SkyCore agent notifies all other agents in the network with a location update for this UE. Finally, our agent on thefi rst UAV pushes the updated label stack corresponding to the UE onto its pending downlink packets and forwards them to the second UAV.

6 IMPLEMENTATION

SkyCore prototype. We prototyped a complete version of SkyCore that involved extensive engineering effort. Our prototype has three notable features: (1) seamlessly works with commercial LTE RANs and off-the-shelf UEs (SIM cards are programmed to connect to SkyCore) by exchanging signaling and data traffic with them; (2) is fully virtualized and can manage multiple LTE UAVs out of the box by forming a wireless network of SkyCore agents; and (3) fully adheres to our proposed designs both for a single agent (Figs. 11 and 12) and across agents (inter-agent communication) (Figs. 10 and 13). Each SkyCore agent consists of a controller enforcing control plane policies and a switch processing user data traffic. We developed a high-performance multi-threaded controller in C++ and built our SkyCore switch on top of OVS [42] software switch in the kernel space. We substantially instrumented and optimized OVS as it does not support our customfl ow tables and switch actions (e.g., our P4-enabled tunneling scheme and GTP-U tunnel encapsulation/decapsulation operations). Since our baseline (Edge-EPC based on OpenEPC [12]–will be described shortly) operates in the user space, we developed another variant of the SkyCore switch in the user space on top of Lagopus software switch [9]. This ensures that our comparisons are at the architecture level and independent of a particular packet forwarding technology.

UAV experiments. We conduct three kinds of experiments. *(1) Outdoor Small-scale: 2 UAV, few UEs.* We deploy the SkyCore prototype on two advanced DJI Matrice 600 Pro drones. We securely install two machines on each drone. One of the machines (platform P1) is a low-end single-board 4-core server with 8 GB of RAMs and 1.9 GHz CPU that executes SkyCore and Edge-EPC. It is also equipped with a wireless network card to support our inter-agent communication. The other machine is a commercial LTE small cell (ip.access S60 eNB) supporting LTE UEs (50 Mbps downlink rate per UE) and connects through an Ethernet cable to platform P1. *(2) Outdoor Large-scale: 2 UAV, tens of UEs.* To stress test SkyCore's control and data planes in the presence of a large number of UEs, we replace the eNB on the drone with another single-board server that runs a unified RAN/UE emulator (emulates both an eNB and activity of a large number of UEs). The emulator interacts with the LTE core similar to real UEs. *(3) Emulating Powerful UAV platforms.* To understand SkyCore's performance with more powerful UAVs, we emulate the latter by replacing platform P1 with a high-end server (platform P2) – an Intel Xeon E5-2687W processor operating at 3.0 GHz with 12 CPU cores and 128 GB of RAM. Since it is not possible tofl y our current drone with such a server, only these experiments are conducted in the lab.

Baseline. Through outdoor drone-based experiments, we have already demonstrated the limitations of the legacy wireless EPC on the ground (section 2). Thus, this section is focused on comparisons between the Edge-EPC architecture (a standard EPC on each LTE UAV) and SkyCore. We choose to implement Edge-EPC using OpenEPC [12] because it is

the most complete open-source implementation of the 3GPP EPC architecture that can work with commercial devices (e.g., LTE eNBs and smartphones). Other non-standard and more optimized EPC designs (listed in section 9) are not publicly available, often do not work with the commercial LTE devices, and inherit most of the standard EPC problems in our UAV-based LTE network environment (e.g., high resource usage, lack of support for inter-EPC coordination).

Metrics. We study four performance metrics under different network saturation levels: (1) UE-perceived control delay in network access (LTE attach/detach), (2) UE-perceived service disruption time in LTE active/idle-mode mobility, (3) CPU usage on our resource-constrained UAVs, and (4) supported data plane rate for user traffic.

7 EVALUATION

Wefi rst show the basic functionality and potential of SkyCore in realizing hotspot and standalone LTE UAV networks. We then demonstrate that SkyCore is more efficient and lightweight than the Edge-EPC architecture on different platforms both in small and large-scale experimental settings, thanks to SkyCore's software refactoring and efficient inter-agent communication scheme.

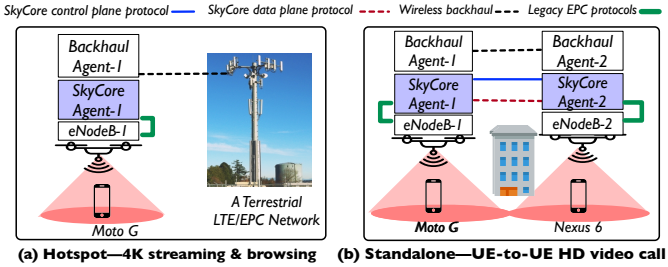


Fig. 15: Our UAV-based setup for the basic functionality experiments. In Edge-EPC (the baseline), each SkyCore agent is replaced by a 3GPP EPC (OpenEPC).

7.1 Small-Scale On-Drone Evaluation

We form a two-drone LTE network (Fig. 15), each in the partial line of sight (affected by one building) of a single mobile UE on the ground. Each drone covers a region with the diameter of 650 feet. The drones operate in a small overlapping area for our mobility experiments.

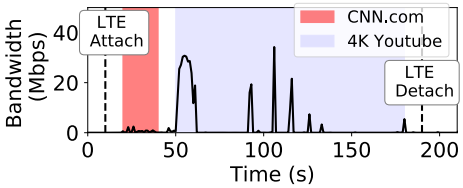


Fig. 16: Basic functionality–hotspot UAV-based LTE network use case: exchanged data traffic and control events over time.

7.1.1 Basic Functionality–LTE Hotspots Use Case. Forming on-demand hotspots is an important use case for LTE UAV as well as 5G networks (section 2). In a single-drone experiment, we show this functionality by connecting one of our drones to the Internet through a terrestrial LTE network not accessible to our UEs on the ground (see Fig. 15a). Next, we turn on a Moto G phone on the ground, which sends an LTE attach request to the SkyCore agent through the on-drone eNB. SkyCore agent successfully completes the LTE attach process by quickly accessing its precomputed policy data store. Then, we visit CNN.com and watch a 4K Youtube video on the phone. Finally, we take the Moto G into the airplane mode, causing the UE to properly detach from our agent. Fig. 16 shows this basic functionality by depicting the data traffic exchanged between the UE and the Internet.

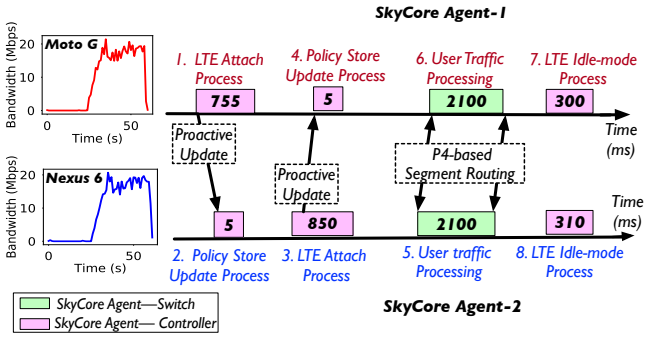


Fig. 17: Basic functionality-standalone UAV-based LTE network use case: UE-to-UE HD video call enabled by SkyCore's efficient inter-agent communication scheme. Control and data plane processing times and traffic exchanges inside and between the SkyCore agents on the two drones.

7.1.2 Basic Functionality–Standalone LTE Use Case. Next, we demonstrate SkyCore's ability to create standalone LTE networks (e.g., betweenfi rst responders across an impassable mountain as discussed in section 2). To emulate such a scenario, we establish a direct video call between our two UEs across a building, each connected to a separate drone, through our inter-agent data plane overlay (see Fig. 15b). Fig. 17 shows the timeline of control and data plane traffic exchanges between the two SkyCore agents. We again turn on a Moto G phone in the area covered by thefi rst drone. Its SkyCore agent handles the LTE attach process and sends a background SkyCore update message to the other drone's agent. The message consists of location, policy and security updates as described in section 3. After the second agent processes this update, we turn on a Nexus 6 phone in the area covered by the second drone, triggering a similar SkyCore update message to thefi rst agent in the background. Finally, we establish a 35-sec HD video call from the Nexus 6 to the Moto G. Owing to SkyCore's proactive background updates, the agent corresponding to the Nexus 6 does not have to wait to discover the location of the other UE. Based on our segment-based tunneling scheme, it immediately pushes the correct label stacks on its egress user data traffic and forwards it to the other agent. A similar process manifests in the reverse direction. In this two-UAV enabled video call, 7.5K video packets were successfully exchanged between the UEs.

7.1.3 Performance Benefits of Refactoring. Using the same setting, we demonstrate that SkyCore is significantly more

lightweight than Edge-EPC. For a fair comparison with Edge-EPC, we employ SkyCore's user-space version here. We sample and average the LTE attach/detach delay and uplink/-downlink bandwidth for the Moto G in the area covered by thefi rst drone at 40 locations. As Fig. 18 and Table 1 show, SkyCore on average reduces the network control plane delay (spent in the core) by 69%-90% and the UE-perceived control plane delay by 40%-60%. In addition, it doubles the uplink/downlink rates measured for the UE. Further, SkyCore lowers the avg. CPU usage on the machine running the core network by 25% in the LTE attach/detach events. These savings come from our precomputation of network policies and consolidation of the EPC functionality onto our compact SDN-driven agents.

Table 1: Benefits of SkyCore's software refactoring of the EPC functionality on UE-perceived QoS.

	Avg. Data plane Bandwidth (Mbps)		Avg. UE-perceived Control delay (ms)	
	Downlink	Uplink	Attach	Detach
SkyCore	48.2	17.8	921	300
Edge-EPC	21.7	10.9	1545	750

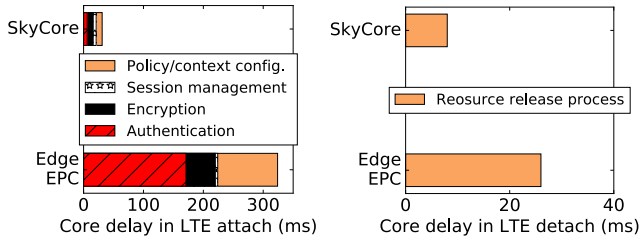


Fig. 18: Breakdown of the network access delay in the core.

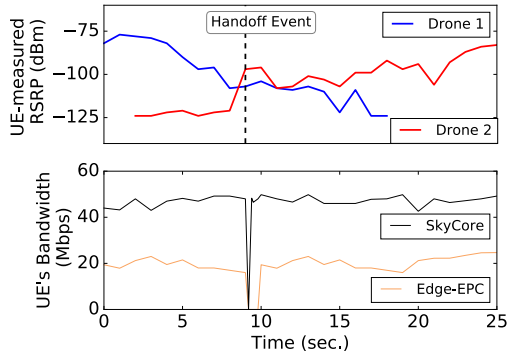


Fig. 19: Benefits of SkyCore's inter-agent communication scheme: SkyCore provides seamless active-mode mobility support while Edge-EPC causing severe connection drops.

7.1.4 Efficient Inter-Agent Communication–Handoff. Unlike Edge-EPC, SkyCore supports seamless UE mobility, owing to its efficient inter-agent communication scheme. In this experiment, we measure the service disruption experienced by a mobile UE moving between the regions covered by our two drones and triggering a handoff event. Fig. 19 depicts the signal strength received from the two drones on the UE and its continuous bandwidth measurements using `iPerf3`. The RAN on thefi rst drone collects UE-measured RSRP values and sends a Handoff Required message to its local SkyCore agent when the RSRP values from the second drone become higher. Since SkyCore agents on the drones are already synced, the UE gets migrated to the second drone within a minimal 140 ms (incurred in the inter-agent coordination). In contrast, Edge-EPC does not support mobility of the UE and thus forces the UE to go through the detach process with the EPC on thefi rst drone, followed by the heavy attach process with the EPC on the second drone. The entire process results in 2 seconds of disconnection time, significantly impacting mobile application performance.

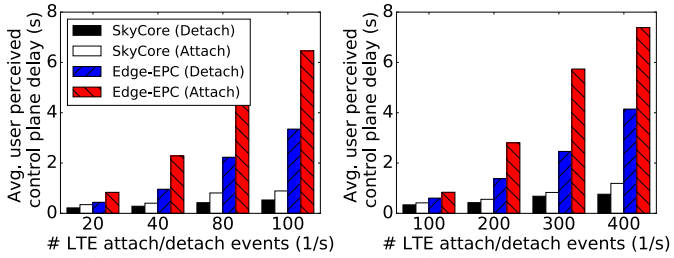


Fig. 20: Benefits of SkyCore's software refactoring at scale. SkyCore substantially reduces network access time in LTE UAV networks within the limits of their resources.

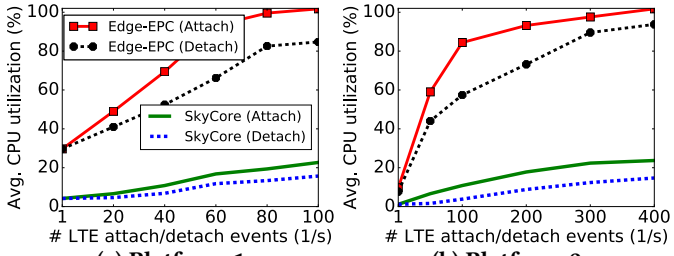


Fig. 21: Benefits of SkyCore's software refactoring of the EPC at scale: SkyCore uses minimal CPU resources to handle large-scale network access requests.

7.2 Large-Scale On-Drone Evaluation

Using the same two-drone experimental setting, we replace the ip.access eNB with a RAN/UE simulator on each drone to test SkyCore and Edge-EPC under large-scale network access and mobility workloads.

7.2.1 Performance Benefits of Refactoring at Scale– Attach/detach Storm. This experiment demonstrates SkyCore's operating potential in highly resource-constrained UAV environments. Our RAN/UE emulator on thefi rst drone emulates afl ash crowd event with a large number of users entering the region covered by a drone. Similarly, the emulator creates LTE detach storms having many users gracefully disconnecting from the drone. During this process, we sample the CPU utilization of the LTE core machine and measure the average control plane delay perceived by the UEs. In Fig. 20a, we observe that the UEs experience exponentially larger delays when the attach/detach load on Edge-EPC increases. In particular, when the number of attach requests per sec. reaches 100, the UEs must wait by up to 6 seconds before

connecting to the network, thereby degrading QoE. In contrast, we notice that the network access delay is below 1s when the drone employs SkyCore because of our software refactoring of the EPC functionality. To better understand the reason, we look at Fig. 21a showing the CPU utilization of the LTE core machine. Since the EPC is a complex system, we observe that Edge-EPC quickly uses available CPU resources on the drone and thus faces performance bottlenecks. Although user-perceived control plane delay in the detach process is usually less critical in practice, the same trend can be observed for both SkyCore and Edge-EPC.

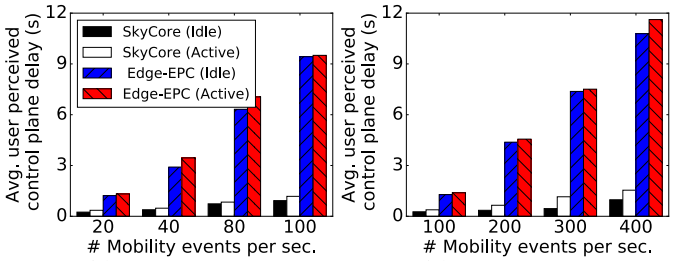


Fig. 22: Benefits of SkyCore's efficient inter-agent communication scheme at scale: SkyCore seamlessly supports large-scale idle/active-mode UE mobility between UAVs.

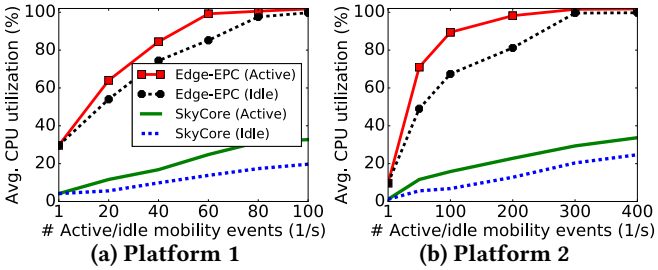


Fig. 23: Benefits of SkyCore's efficient inter-agent communication at scale: SkyCore supports large-scale idle/active-mode UE mobility in a resource-efficient manner.

7.2.2 Efficient Inter-Agent Communication at Scale– UE Mobility Storm. This experiment demonstrates SkyCore's capability in handling increased mobility events in LTE UAV networks. We add the second drone to our experiment. Our RAN/UE simulator on thefi rst drone and second drone simulate scenarios where a large number of connected and idle UEs move between the areas covered by the two drones. It sends a variable number of LTE Handoff Required Messages and TA Update Requests to the core network to trigger active-mode and idle-mode mobility events. We increase the number of mobility events until either Edge-EPC or SkyCore face performance bottlenecks. We measure the service disruption experienced by the UEs when Edge-EPC and SkyCore are in place as well as the CPU utilization of the LTE core machine. Fig. 22a shows that the UEs experience a large control plane delay and service disruption in the Edge-EPC deployment. Due to lack of control plane communication between the Edge-EPC instances, the UEs have to undergo a complete LTE detach process (with thefi rst drone) and attach process (with the second drone) both during connected-mode and idle-mode mobility. In Edge-EPC, when 100 mobility events occur per second, users on average experience by up to 10 second of disruption, which is very significant. More importantly, by transforming each mobility event to a pair of LTE attach-detach events, we observe in Fig. 23a that Edge-EPC creates severe bottlenecks on the drone platform. In contrast, the SkyCore agents sitting on the two drones quickly and seamlessly execute the handoff and TA update operation, owing to our proactive synchronization of network policies and states associated with different UEs in the background. Thus, they incur minimal computation for mobility workloads.

Table 2: SkyCore's inter-agent communication scheme (broadcast proactive updates) has a negligible overhead on inter-UAV backhaul links and thus is scalable.

	LTE attach/detach & mobility events per sec.							
	Platform P1				Platform P2			
	20	*40*	*80*	*100*	*100*	*200*	*300*	*400*
Bandwidth overhead on Inter-drone Link (Mpbs)	1.2	2.2	3.8	5.5	5.5	8.5	14.2	18

7.2.3 Efficient Inter-Agent Communication at Scale– Overhead of Proactive Updates. Next, we demonstrate the scalability of SkyCore's proactive synchronization scheme in our multi-UAV environment. Our metric is the overhead of proactive updates on inter-drone backhaul links. For the above two experiments and various number of LTE attach/detach and mobility events per second, Table 2 shows the maximum rate of the broadcast traffic on the backhaul link connecting the SkyCore agents. We observe that the bandwidth overhead does not increase linearly with respect to the number of LTE events in the system as the updates corresponding to the different UEs are periodically (every 100 ms in our setup) batched together. Our second observation is that when the load is 100 events per sec (representing tens of thousands of UEs in today's 4G/LTE networks [28]), the overhead on the backhaul link is only 5.5 Mbps. Compared to potential user data traffic in the system, this overhead is negligible. Using a back-of-the-envelope calculation, it is straightforward to show that this overhead remains small in our target environment consisting of a few to tens of UAVs.

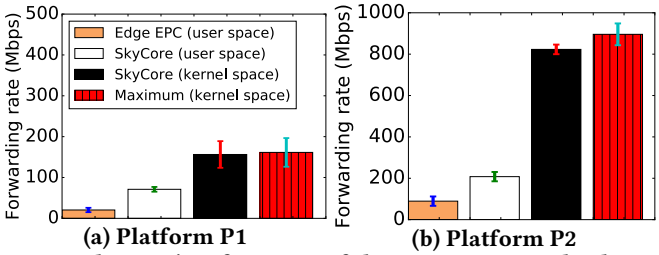
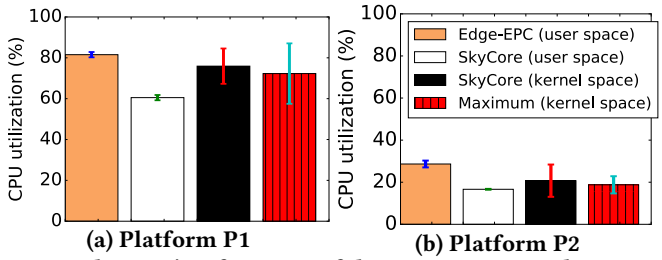


Fig. 24: SkyCore's refactoring of the EPC increases the data rate support on resource-challenged UAVs.

7.2.4 Performance Benefits of Refactoring at Scale– Stress-testing Data Plane. In a single-drone experiment, we instruct our RAN/UE simulator to generate data traffic for a variable number of UEs in the network in parallel. It encapsulates the traffic of each UE into a separate GTP tunnel similar to real RANs. We run `iPerf3` bandwidth tests for the simulated UEs



(a) Platform P1 (b) Platform P2

Fig. 25: SkyCore's refactoring of the EPC minimizes the CPU resource needed on UAVs for a specific forwarding rate.

in parallel. Fig. 24a shows the aggregate, steady forwarding rate supported by SkyCore and Edge-EPC. When using the same packet forwarding technology, we observe that SkyCore (user space) supports 2× more packet forwarding rate compared to Edge-EPC on the drone. Our software refactoring and data plane consolidation substantially removes the I/O costs and processing delays from the LTE core data plane. We were able to further improve the throughput by 2× (close to a Gbps) by moving our software switch to the kernel space. Maximum is the ideal version of our OVS switch that processes data traffic without applying any network policies.

7.3 Scaling to Powerful UAV Platforms.

We replace the core machine (platform P1) with a high-end server (platform P2) to emulate more power UAV platforms in the future. Figs. 20b, 21b, 22b, 23b, and Table 2 show our results with platform P2 for the previous four experiments. We observe that SkyCore is substantially more resource-efficient than Edge-EPC even on high-end servers. SkyCore is able to scale and provide almost line-rate forwarding rate while using a fraction of the drone's CPU resources. We plan to move SkyCore's implementation to OVS-DPDK to more efficiently leverage the available CPU cores.

8 RELATED WORK

SDN/NFV-based EPC. Recently, the wireless networking community has proposed several software-defined EPC solutions. SoftCell [28], SoftMoW [34, 35, 38] and others [11, 47] enhance the programmability of the EPC by decoupling its control and data planes. KLEIN [43] and SCALE [18] optimize the placement of the EPC nodes on geo-distributed DCs. ECHO [40] deals with EPC-node failure in unreliable public clouds. PEPC [44] and SoftBox [36] scale the EPC data plane by creating a per-UE EPC-in-box. While there are some similarities between SkyCore and these proposals, the differences are significant. These prior designs are customized for highly-reliable, often hierarchical DC infrastructure, where over provisioning and reactive network updates are inexpensive. In contrast, SkyCore operates in an unreliable and resource-constrained wireless environment, where such approaches scale poorly.

SDN control and data planes. There is a rich literature in distributed SDN control planes designs with hierarchical andfl at structures (e.g., ONOS [19], [24, 29, 39, 50]). Most of the schemes are designed for DC networks and operate based on a centralized data store or complex consensus algorithms, which are ill-suited for our unreliable multi-UAV environment. There are some memory-efficient switch designs for SDN [33, 37, 49] that can be leveraged to improve packet forwarding between SkyCore agents.

RAN optimization for LTE UAVs. DroneNet [22] extends the coverage of existing LTE cells by creating WiFi on-drone hotspots. Some recent works [31, 32, 48] investigate the theoretical optimization of a UAV trajectory for certain mobile users on the ground (e.g., maximize the minimum average rate among all user). These RAN efforts are predominantly for a single UAV and complementary to SkyCore that focuses on the EPC design for multi-UAV LTE networks.

9 DISCUSSIONS AND REMARKS

We presented the design and implementation of a novel Edge-EPC architecture- SkyCore, supporting the untethered and reliable operation of multi-UAV LTE networks. Several aspects of SkyCore are worthy of further investigation.

Relevance to high-altitude platforms: While today's high-altitude UAV networks [6, 8] might aim to provide Internet connectivity through a legacy EPC on the ground, we believe the benefits of a SkyCore design for an edge EPC significantly outweigh its drawbacks and apply to such deployments as well.

Impact of wireless inter-UAV backhaul design. We did not discuss the design of the backhaul agents forming the physical wireless mesh network among UAVs. The design of an efficient backhaul needs to be jointly optimized with the RAN as the position of the UAV simultaneously affects the performance of the backhaul as well as the access to UEs.

Scalability implication for large UAV-based LTE networks. SkyCore's design focuses on the challenges unique to multi-UAV LTE networks that typically span from a few to at most tens of UAVs (city-scale). While our design decisions (e.g., inter-agent proactive updates, policy precomputation) are efficient and scalable for our target environment, they are not designed to scale in nation-wide LTE networks with hundreds of millions of UEs.

Applicability to terrestrial 5G networks: While EPC-RAN communication is often reliable in terrestrial networks, operators can leverage Edge-EPC designs (e.g.,SkyCore) to move the EPC functionality to their edge clouds or cell towers and realize ultra-low latency required by many 5G use cases. Such deployments are motivated by operators' push towards mobile edge computing (MEC) [21, 41] and their efforts in deploying white box switches at cell towers [2].

ACKNOWLEDGMENT

We would like to thank the reviewers, our shepherd, Ayon Chakraborty and Amir Khojastepour at NEC Labs America, and Harsha V. Madhyastha and Mosharaf Chowdhury at the University of Michigan for providing valuable feedback on our work. This work was conducted when Mehrdad Moradi was an intern at and funded by NEC Labs America.

REFERENCES

[1] Amazon patents fulfillment center towers to increase drone delivery efficiency. https://goo.gl/x66KXF.

[2] AT&T is deploying white box hardware in cell towers to power mobile 5G era. https://goo.gl/snRW6M.

[3] AT&T prepares for hurricane season with disaster response exercise. https://goo.gl/8zLvLn.

[4] CBRS Spectrum. https://goo.gl/3zbYyo.

[5] Drones can provide cell phone service . https://goo.gl/GmR93Y.

[6] Facebook Project Aquila. https://goo.gl/gHYVa7.

[7] Flying COW connects Puerto Rico. https://goo.gl/NEq1HA.

[8] Google X: Project Loon. https://goo.gl/skSz1z.

[9] Lagopus: SDN switch. http://www.lagopus.org/.

[10] LTE signaling storm. https://goo.gl/ZuwYfe.

[11] M-Cord: Mobile Cord. http://goo.gl/pZgSvg.

[12] OpenEPC. http://www.openepc.com/.

[13] Oracle communications LTE diameter signaling index. https://goo.gl/6BZ8Fo.

[14] Verizon Continues tests using drones. https://goo.gl/ba2Wz8.

[15] Verizon trials drones asfl ying cell towers. https://goo.gl/q9YjNv.

[16] When COWsfl y: AT&T sending LTE signals from drones. https://goo.gl/9u33qC.

[17] B. Aoun, R. Boutaba, Y. Iraqi, and G. Kenward. Gateway placement optimization in wireless mesh networks with QoS constraints. *IEEE JSAC*, 2006.

[18] A. Banerjee, R. Mahindra, K. Sundaresan, S. Kasera, K. Van der Merwe, and S. Rangarajan. Scaling the lte control-plane for future mobile access. In *Proc. ACM CoNEXT*, 2015.

[19] P. Berde, M. Gerola, J. Hart, Y. Higuchi, M. Kobayashi, et al. ONOS: towards an open, distributed sdn os. In *Proc. ACM SIGCOMM Workshop on HotSDN*, 2014.

[20] P. Bosshart et al. P4: Programming protocol-independent packet processors. *ACM CCR*, 2014.

[21] J. Cho et al. ACACIA: Context-aware edge computing for continuous interactive applications over mobile networks. In *Proc. ACM CoNEXT*, 2016.

[22] A. Dhekne et al. Extending cell tower coverage through drones. In *Proc. ACM HotMobile*, 2017.

[23] B. Han, V. Gopalakrishnan, et al. Network function virtualization: Challenges and opportunities for innovations. *IEEE Communications Magazine*, 2015.

[24] S. Hassas Yeganeh and Y. Ganjali. Kandoo: a framework for efficient and scalable offloading of control applications. In *Proc. ACM SIGCOMM Workshop on HotSDN*, 2012.

[25] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. In *Proc. IEEE HICSS*, 2000.

[26] C.-Y. Hong, S. Kandula, R. Mahajan, et al. Achieving high utilization with software-driven WAN. *ACM CCR*, 2013.

[27] S. Jain, A. Kumar, S. Mandal, et al. B4: Experience with a globally-deployed software defined WAN. *ACM CCR*, 2013.

[28] X. Jin, L. E. Li, et al. Softcell: Scalable andfl exible cellular core network architecture. In *Proc. ACM CoNEXT*, 2013.

[29] T. Koponen, M. Casado, N. Gude, et al. Onix: A distributed control platform for large-scale production networks. In *Proc. USENIX OSDI*, 2010.

[30] Y. Li, Z. Yuan, and C. Peng. A control-plane perspective on reducing data access latency in lte networks. In *Proc. ACM MobiCom*, 2017.

[31] X. Lin, R. Wiren, S. Euler, A. Sadam, et al. Mobile networks connected drones: Field trials, simulations, and design insights. *arXiv preprint arXiv:1801.10508*, 2018.

[32] X. Lin, V. Yajnanarayana, S. D. Muruganathan, et al. The sky is not the limit: LTE for unmanned aerial vehicles. *arXiv preprint arXiv:1707.07534*, 2017.

[33] Y. Lin, U. C. Kozat, J. Kaippallimalil, M. Moradi, A. C. Soong, and Z. M. Mao. Pausing and resuming networkfl ows using programmable buffers. In *Proc. ACM SOSR*, 2018.

[34] M. Moradi, L. E. Li, and Z. M. Mao. SoftMoW: a dynamic and scalable software defined architecture for cellular wans. In *Proc. ACM SIGCOMM Workshop on HotSDN*, 2014.

[35] M. Moradi, L. E. Li, and Z. M. Mao. Softmow: A dynamic and scalable software defined architecture for cellular WANs. *Presented as part of the Open Networking Summit (ONS)*, 2014.

[36] M. Moradi, Y. Lin, Z. M. Mao, S. Sen, and O. Spatscheck. Softbox: A customizable, low-latency, and scalable 5G core network architecture. *IEEE JSAC*, 2018.

[37] M. Moradi, F. Qian, Q. Xu, Z. M. Mao, D. Bethea, and M. K. Reiter. Caesar: High-speed and memory-efficient forwarding engine for future internet architecture. In *Proc. ACM/IEEE ANCS*, 2015.

[38] M. Moradi, W. Wu, L. E. Li, and Z. M. Mao. SoftMoW: Recursive and reconfigurable cellular wan architecture. In *Proc. ACM CoNEXT*, 2014.

[39] M. Moradi, Y. Zhang, Z. M. Mao, and R. Manghirmalani. Dragon: A scalable,fl exible and efficient traffic engineering framework for isp networks. *IEEE JSAC*, 2018.

[40] Nguyen, Binh and Zhang, Tian and Radunovic, Bozidar and others. A reliable distributed cellular core network for hyper-scale public clouds. MSR Technical Report, 2018.

[41] M. Patel et al. Mobile-edge computing introductory technical white paper. *White Paper, Mobile-edge Computing (MEC) industry initiative*, 2014.

[42] B. Pfaff, J. Pettit, T. Koponen, et al. The design and implementation of open vswitch. In *Proc. USENIX NSDI*, 2015.

[43] Z. A. Qazi et al. Klein: A minimally disruptive design for an elastic cellular core. In *Proc. ACM SOSR*, 2016.

[44] Z. A. Qazi, M. Walls, A. Panda, et al. A high performance packet core for next generation cellular networks. In *Proc. ACM SIGCOMM*, 2017.

[45] A. S. Rajan et al. Understanding the bottlenecks in virtualizing cellular core network functions. In *Proc. IEEE LANMAN*, 2015.

[46] I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, M. F. Kaashoek, F. Dabek, and H. Balakrishnan. Chord: a scalable peer-to-peer lookup protocol for internet applications. *IEEE/ACM TON*, 2003.

[47] A. Syed and J. Van der Merwe. Proteus: a network service control platform for service evolution in a mobile software defined infrastructure. In *Proc. ACM MobiCom*, 2016.

[48] Q. Wu, Y. Zeng, and R. Zhang. Joint trajectory and communication design for multi-UAV enabled wireless networks. *IEEE/ACM TON*, 2018.

[49] Y. Yu, D. Belazzougui, C. Qian, and Q. Zhang. Memory-efficient and ultra-fast network lookup and forwarding using othello hashing. *IEEE/ACM TON*, 2018.

[50] Y. Zhang and M. Moradi. Sdn based interdomain and intradomain traffic engineering, July 4 2017. US Patent 9,699,116.