Operating a UAV Mesh & Internet Backhaul Network using Temporospatial SDN

Brian Barritt
Google
1600 Amphitheatre Pkwy.
Mountain View, CA 94043
bbarritt@google.com

Tatiana Kichkaylo
Google
1600 Amphitheatre Pkwy.
Mountain View, CA 94043
kichkaylo@google.com

Ketan Mandke
Google
1600 Amphitheatre Pkwy.
Mountain View, CA 94043
kmandke@google.com

Adam Zalcman
Google
1600 Amphitheatre Pkwy.
Mountain View, CA 94043
viathor@google.com

Victor Lin
Google
1600 Amphitheatre Pkwy.
Mountain View, CA 94043
victorlin@google.com

***Abstract*— In this paper we describe an application of Temporospatial Software Defined Networking (TS-SDN) to UAV networks. Airborne platforms (airplanes, balloons, airships) can be used to carry wireless communication nodes to provide direct-to-user as well as backhaul connections. Such networks also include ground nodes typically equipped with highly directional steerable transceivers. The physics of flight as well as state of the atmosphere lead to time-dynamic link metrics and availability. As nodes move around, the network topology and routing need to adjust to maintain connectivity. Further, mechanical aspects of the system, such as time required to mechanically steer antennas, makes the reactive repair approach more costly than in terrestrial applications. Instead, TS-SDN incorporates reasoning about physical evolution of the system to proactively adjust the network topology in anticipation of future changes. Using airborne networks under development at Google as an example, we discuss the benefits of the TS-SDN approach compared to reactive repair in terms of network availability. We also identify additional constraints one needs to account for when computing the network topology, such as non-interference with other stationary and moving sources. Existing SDN standards do not support scheduled updates necessary in a TS-SDN. We describe our extensions to control messages and software implementation used in field tests.**

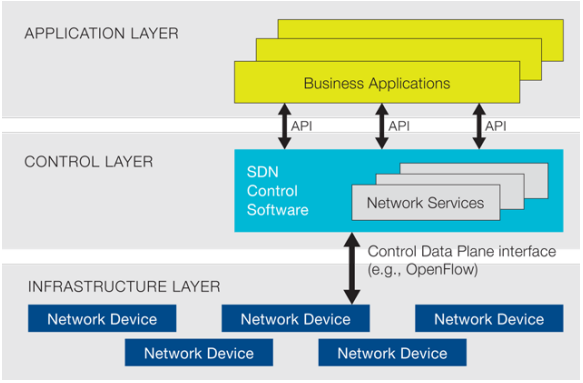

Figure 1. A high-level overview of the software-defined networking architecture

TABLE OF CONTENTS

1. INTRODUCTION

The use of unmanned aerial vehicles (UAVs) and other atmospheric platforms has recently emerged as a potential solution for providing Internet access to rural populations in emerging economies. Titan Aerospace, which Google acquired in 2014, designed a high-altitude, long-endurance UAVs for this purpose. Google's parent company, Alphabet, is developing a network of stratospheric balloons to extend LTE access - a moonshot project in its X lab. Facebook has been developing a high-altitude, long-durance UAV called Aquila after acquiring the British company Ascenta.

Other past and present efforts to develop airborne platforms that could be used for similar purposes include Airbus's Zephyr program, Thales Alenia Space's Stratobus, the Vulture Program's SolarEagle, NASA's Helios, and joint US DoD and DARPA projects such as the ISIS Airship, among others [1].

This paper is organized as follows. In the following section we briefly present the concepts of Software Defined Networking (SDN), its existing extensions to wireless networks and discuss some measures for accommodating time-dynamic properties of UAV-powered networks. In section 3, we describe their overall architecture. In section 4, we discuss unique challenges presented by these networks and the constraints that they impose on the network control mechanisms. In section 5, we describe the temporospatial SDN and its advantages more generally. In section 6, we discuss some technical details of our implementation. In section 7, we contrast and compare the centralized, SDN-based routing control in UAV-powered networks with various distributed approaches. Finally, we conclude in section 8.

978-1-5090-1613-6/17/31.00

2. BACKGROUND

Software Defined Networking (SDN) technology has become more common in terrestrial networking. SDN enables the implementation of services and applications that control, monitor, and reconfigure the network layer and switching functionality. SDN provides a software abstraction layer that yields a logically centralized view of the network for control plane services and applications. This centralized view, in turn, enables coherent management of modern large scale networks, whose topology continuously evolves over time [2]. A high-level overview of the SDN reference architecture [3] is shown in Figure 1.

Recently, new requirements have led to proposals to extend this concept for Software-Defined Wireless Networks (SDWN), which decouple radio control functions, such as spectrum management, mobility management, and interference management, from the radio data-plane [4]. As before, SDWN enable centralized management of various aspects of constantly evolving networks, which reduces labor requirements and improves agility of the network.

In traditional networks, wired or wireless, failure is unpredictable. Such unpredictable outages may be caused by hardware failure or occluded links. This is typically addressed by building redundancy into the network and/or reactively repairing breakages when they occur. Outside of random failures, the properties of traditional networks remain stable.

In aerospace networks, however, the properties of the candidate network topology change constantly, but in a somewhat predictable manner. While it is possible to treat such changes the same way as traditional failures, doing so leads to inefficient use of resources and to user-visible network disruptions.

A better solution is to extend the SDN concept again, this time by adding a temporospatial aspect [5]. In this extended model, called Temporospatial-SDN (TS-SDN), the holistic view of the topology provided to SDN applications is annotated with the predicted, time-dynamic properties of its network links and nodes, and network changes are *scheduled* as opposed to executed as soon as possible. In TS-SDN, the SDN controllers utilize knowledge of the physical position and trajectory of each platform and its antennas to make predictions about the future state of the lower-level network [5]. For instance, the state and performance of current or potential future line-of-sight wireless links in a UAV or satellite network are relatively easy to accurately predict using modeling and simulation tools such as STK.[2]

3. NETWORK ARCHITECTURE

We use the hierarchical internetworking model [6] to describe the high-level architecture of a network that provides 4G LTE mobile Internet service from airborne platforms.

As depicted in Figure 2, the Access Layer is comprised of an LTE base station (eNodeB) on each airborne platform, which connects directly to many end user handsets, tablet, etc, referred to as user equipment, or UEs. In accordance with 3GPP standards, each eNodeB establishes a stateful connection (S1 interface) to an Evolved Packet Core (EPC) in the Core Layer of the network for data plane flows and control plane signaling. The EPC peers with the Internet, performs billing and subscriber management functions, and handles Access Layer mobility management.

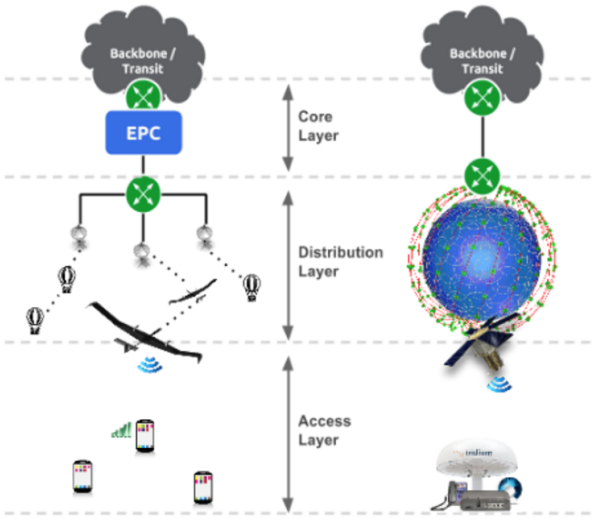

Figure 2. UAV and LEO satellite networks are unique in the high degree of mobility in their Distribution Layer.

The Distribution Layer connects the eNodeB to the Core Network infrastructure. In traditional LTE networks, this is accomplished via static, wired (typically fiber) infrastructure. Our Distribution Layer, however, is comprised of a time-dynamic multi-hop wireless mesh/backhaul network, which is significantly more complex.

Because these networks are designed to provide Internet service to underserved populations of users, we cannot assume that terrestrial fiber backhaul infrastructure is available within the coverage area of a given UAV. As such, bent pipe architectures in which the UAV acts as an analog transponder for LTE service cannot be used. Instead, we favor a packet-based multi-hop wireless mesh network for our backhaul.

Terrestrial ground stations must be located in regions with sufficient fiber or wired backhaul, and may be beyond line-of-sight (LoS) for some UAVs. Because of the long distances between peers, to keep transmit power within safe and realizable limits, ground stations and UAVs must use mechanically or electronically steered beams (gimbaled parabolic antennas, phased arrays, or steerable free-space optics assemblies) to establish highly-directional links, even to nearby peers. Different platforms in the network may have varying constraints on the number of mesh links that they can form with peers. For example, a ground station may only be able to track a single UAV, while each UAV may be able to simultaneously steer multiple beams towards several other platforms. Different link types may be used within the same mesh, e.g., optical links between airborne platforms and radio links to ground stations. These links in aggregate form a mesh network that enables multi-hop wireless communication between non-LoS peers.

In some architectures, UAVs and ground stations may also establish links with platforms at even higher altitudes, such as satellites. The highly-directional nature of the Distribution Layer motivates the use of such links as an additional out-of-band control channel to bootstrap the network. That is, since narrow beam link establishment requires both peers to know that they must point at each other at the same time, when nodes cannot *see* anyone they could bootstrap themselves via

[2]http://www.agi.com

such a link. This control channel may be very low data rate and expensive. In many systems, existing control and non-payload communication (CNPC) links may be suitable for this purpose.

4. MESH & BACKHAUL CHALLENGES

The use of highly-directional and narrow beams in a mobile multi-hop, wireless Distribution Layer creates unique challenges. The mobility of network platforms and changing weather conditions impart time-dynamic properties to the availability and fidelity of wireless mesh/backhaul links. Additional complexity in control is caused by the fact that beam steering (whether mechanical or electrical) results in non-zero link acquisition times. Any reconfiguration of the Distribution Layer topology therefore risks disrupting connection-oriented network protocols that need to transit across this time-dynamic network.

These challenges impart the following constraints:

- Centralized Topology Control - Constructing a mesh topology that may include narrow, mechanically steered beams (with slew and link acquisition times on the order of at least several seconds) is prohibitively difficult using distributed protocols. A centralized controller is therefore required to coordinate link establishment between peers.
- No Addressing Hierarchy - In traditional 3GPP standards-based radio access networks (RANs), the S1 interface between the EPC and eNodeB uses IP at layer 3, and both the EPC and each eNodeB could reasonably assume that the other has a static IP address. The hierarchy inherent in a traditional RAN's topology allows for routers in the Distribution Layer to use routing prefixes in the forwarding information base (FIB), which helps to minimize the size of forwarding tables. In contrast, the lack of implicit hierarchy in a UAV-based Distribution Layer network means that the assignment of static addresses or address blocks to access points limits our ability to prefix and aggregate routes in the FIB of network routers and switches.
- Disruption Avoidance - These networks are designed to provide Internet access, and most user-traffic uses the TCP protocol. TCP throughput backs off exponentially in the face of packet loss, and prolonged disruptions (on the order of several seconds to minutes, depending on the configuration) can result in socket closures. These can cause users to experience call drops in interactive voice and video, cancelled file transfers, and other problems. It is therefore important that we avoid disruption to the S1 user plane connection between the eNodeB and EPC.
- Traffic Engineering - These networks need to support sufficient capacity to carry all traffic between eNodeB and EPC, or between eNodeBs. The network needs to ensure it can provide sufficient capability to carry all traffic. For example, the routing algorithm should investigate use of all available links, instead of using only the shortest path route.

5. TEMPOROSPATIAL SDN

As previously mentioned, the TS-SDN concept was developed to capture the potential for SDN controllers to utilize knowledge of physics and physical relationships to make predictions about the future state of the lower-level network [5]. Whether in multi-hop wireless mesh/backhaul networks consisting of UAVs and ground stations, or in low-earth orbiting (LEO) satellite networks [7], the predictably evolving time-dynamic nature of highly-directional steerable links presents a significant opportunity to benefit from the application of TS-SDN.

The advantages of TS-SDN in these networks include:

- Topology Management - Since the entire set of potential links and paths available at the current time or any near-future time can be accurately predicted, optimal link sets can be selected, and unnecessary links (e.g. redundant cross-links that are not part of an optimal path) can be disabled. This can provide power savings, among other advantages.
- Proactive Packet Routing - Routes programmed via TS-SDN can be adjusted in advance of handovers or other events that would temporarily disrupt a path. Since the physical timing of link events is highly predictable, packet routes at the network layer can be updated in direct accordance with the predicted physical layer events. This can avoid user service impacts including dropped packets, reduced throughput, multimedia stream disruptions, etc.
- Radio Resource Management - Radio parameters such as channel and transmit power can be centrally controlled in order to avoid and minimize interference and to comply with radio spectrum regulations.
- Scheduled operation - Due to the predictable nature of the system, the controller can schedule operations for each switch controlled by the TS-SDN controller. This should virtually eliminate the network disruption for expected topology changes.

Many different control plane architectures are feasible for TS-SDN. While OpenFlow generally operates remotely between switches and servers hosting the controller software over a network, controllers can also be co-located with switches (e.g. onboard a relay spacecraft). TS-SDN allows centralized intelligence to completely orchestrate the future network state, distribute timed directives to localized relay controllers, and have those directives executed at set times. If control plane connectivity to the centralized intelligence is broken, the controller may still deliver these directives to a network node via a (potentially more expensive) out-of-band control channel.

6. APPLICATION & IMPLEMENTATION

Google's implementation of a TS-SDN operating system follows the general architecture depicted in Figure 1. Its central component is the Control Layer which interfaces with a set of SDN applications in the Application Layer and a set of Control Data Plane Interface (CDPI) agents running on the network devices in the Infrastructure Layer.

The Control Layer employs a network data model to expose an abstract, mutable, high-level view of the network to the SDN applications. The data model consists of a number of typed entities that describe both networking and physical attributes of the network nodes and links. Examples of supported node attributes include: assigned IP address, time-dynamic position and orientation, transponder ICAO address, antenna radiation patterns, etc. Examples of network link attributes include: physical implementation of the link (wired or wireless; fixed, omnidirectional, or steerable), link state (enacted or candidate) and time-dynamic quality metrics such as expected bit error rate. Figure 3 shows an example of a network graph with different types of nodes and their candidate links. These links are accessible according to models, but are not yet established in the network.

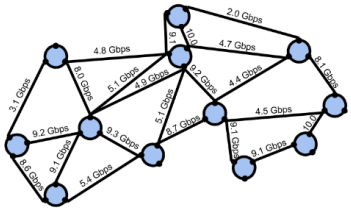

Figure 3. The network data model represents the nodes (verticies) and links (edges) in the topology and includes all accessible wired and wireless links.

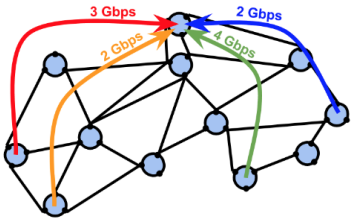

Figure 4. The network topology is annotated with required end-to-end packet connectivity and provisioned flow capacities.

In addition to describing the current state of the network and its properties, the model also allows SDN applications to express network configuration objectives, such as a request to establish end-to-end packet connectivity between a pair of network nodes or interfaces, or to provision and reserve a minimum, time-varying network capacity for transiting network flows. Figure 4 shows an example in which the data model is expressing that four UAV nodes in the network need to establish end-to-end packet connectivity with certain minimum capacities to the EPC node.

The network data model serves as a shared state as well as an API between SDN applications, managers, and services – each of which consumes and produces data of certain kinds. For example, one controller service produces link quality data based on telemetry from vehicles, weather data, and physics models. A backhaul request manager sets connectivity objectives as UAVs enter and exit LTE service regions. Together, this data specifies the input problem to be solved by a separate SDN application that is responsible for jointly optimizing the network topology and routing as shown in Figure 5. Our experience with developing these and other SDN applications highlights the ease with which the TS-SDN operating system can be extended to include complex and powerful network control logic.

The abstract, high-level intents of SDN applications are continuously translated by the Control Layer to the low-level network node state information exchanged with CDPI agents in the Infrastructure Layer. The information exchanged consists of state change requests sent to the CDPI agents and state notifications sent back in the opposite direction. In traditional SDN systems, this exchange uses one of the existing CDPI protocols, such as OpenFlow.

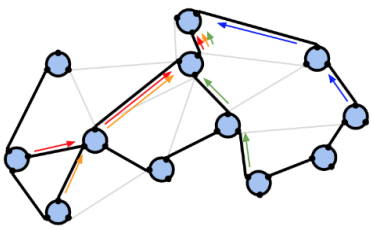

Figure 5. A traffic engineered solution is created by finding a satisficing subgraph or spanning tree.

Table 1. Comparison of Three Mesh Routing Protocols

Properties	AODV	DSDV	OLSR
Route Format	table driven	table driven	table driven
Route Discovery	reactive	proactive	proactive
Route Maintenance	periodic	periodic	periodic
Core Algorithm	searching	Bellman-Ford	Dijkstra

In our TS-SDN, however, existing CDPI protocols turned out to be insufficient for two reasons. First, potentially intermittent connectivity between the Control Layer and the CDPI agents and the need to synchronize state changes across affected nodes imposes the requirement that state change requests be scheduled for a specified future time. Second, while existing CDPI protocols only support modification of forwarding rules in the Infrastructure Layer, the wireless and temporospatial features of TS-SDN require the control of a broader range of parameters, such as steerable beam tasking and cognitive radio control. Therefore, we have developed a new, OpenFlow-inspired CDPI protocol that facilitates full control over topology, routing, radio parameters and wireless link establishment.

7. SDN vs. Distributed Routing

In SDN, the Control Layer gathers and integrates state information from network nodes to form a coherent, abstract view of the entire network, which is a highly centralized approach. As explained in prior sections of this paper, a centralized controller is required to coordinate wireless link establishment in a network that may consist of narrow, highly-directional, mechanically-steered beams. However, while the establishment and maintenance of an optimal network topology is tightly coupled with the establishment and maintenance of an optimal packet routing or switching solution, the use of a centralized controller for SDN style forwarding is not strictly required. An alternative is to embed sufficient intelligence into network nodes themselves so that their behavior and communication with other nodes leads to the emergence of an adequate switching/routing solution across the whole network. This section compares the centralized and distributed approaches to packet forwarding.

Mesh Routing Protocols

In the past two decades, many dynamic routing protocols have been proposed for wireless ad hoc networks. However we compare TS-SDN routing with only the three popular ones implemented in the ns-3 network simulator[8], namely AODV[9], DSDV[10] and OLSR[11]. Table 1 shows the comparison of the three protocols from four perspectives: route format, route discovery mechanism, route maintenance mechanism and the core algorithm of finding routes.

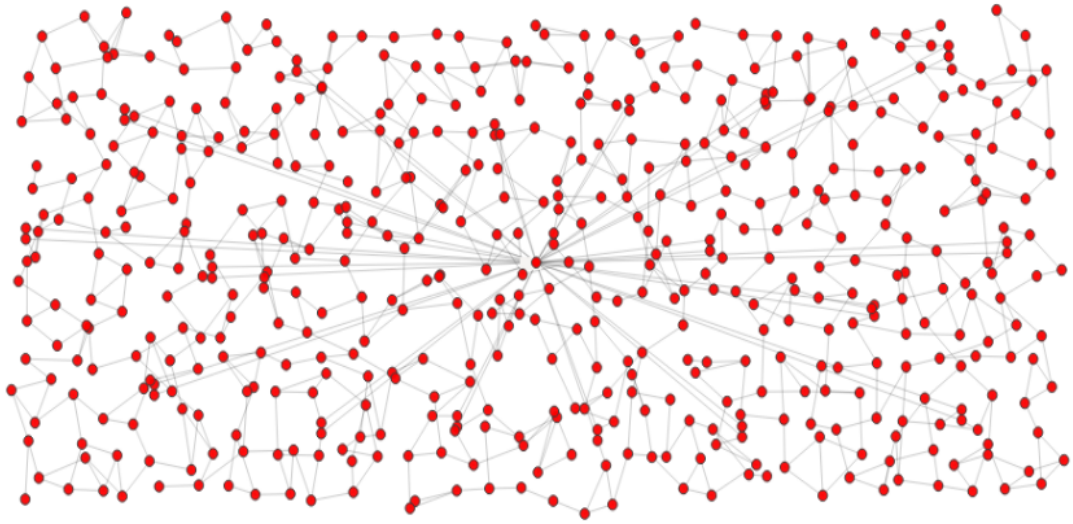

Figure 6. Visualization of the simulated network topology.

Simulation Scenarios

To study the performance of these three routing protocols, we defined a plausible simulation scenario in the ns-3 network simulator. The scenario consists of a static network topology of 487 UAVs and 38 ground stations. Each UAV may establish wireless mesh/backhaul links with up to 3 other nodes, while each ground station node may connect to at most 1 UAV. Ground stations are also connected via a wired, point-to-point packet connection to a network node representing the EPC and Core Layer of the network. Figure 6 provides a visualization of the simulated network topology.

We consider the performance of the three protocols in two scenarios: startup and link failure. In the first scenario, the three routing protocols start from empty routing tables, and we study how long it takes them to find routes to the EPC node for all the UAVs. In the second scenario, we first let the routing protocols run for the same period of time, during which they can build routing tables. Then, we break some links and study how long it takes the routing protocols to recover from the link failures. Therefore, in this scenario, the metric to evaluate performance of the routing protocols is convergence time, i.e., the average period of time it takes the routing protocols to find routes to the EPC node for all the UAVs. As such, the definition of convergence time used here is different from the traditional one, which is the time until the routing protocols find routes for every node to all the other nodes. Here, we say the routing protocol converges for a single UAV if the EPC node receives at least one packet from that UAV.

We used the existing AODV, DSDV and OLSR models implemented in ns-3. However, since these models are hard-coded for wireless broadcast networks (e.g., WiFi), we modified the source code to make them act like point-to-point links in our network model. Since convergence time of the two proactive protocols (i.e., DSDV and OLSR) is determined by the route update period, we set them to small values to be comparable with the convergence time of AODV. Specifically, in DSDV, the periodic updating interval is set to 1 second and the settling time is set to zero. In OLSR, the TC, Mid, Hna and Hello intervals are all set to 0.5 seconds. The trade-off is that a smaller route updating period results in more control overhead throughout the network.

Table 2. Comparison time at startup (in seconds)

Protocol	Convergence Time
AODV	0.0238 s
DSDV	0.0579 s
OLSR	2.1358 s

In the startup simulation scenario, we let all the UAVs send packets to the EPC node at time 0 at a constant bit rate of 50 Kbps. To forward the packets to the EPC node, the routing protocols needed to first search for routes for all the UAVs. In the link failure scenario, we first let the routing protocols run for 800 msec, and then we break some links. Then, at 800 msec into the simulation, all UAVs begin sending packets to the EPC node at a constant bit rate of 50 Kbps. On the EPC node, we measure the time required to receive packets from every UAV in the topology.

Simulation Results

Startup—For the startup scenario, Table 2 shows the average convergence time of three protocols, in which we can see that AODV converges the fastest. One reason is that AODV only searches for routes in an on-demand manner. That is, in our simulation, all UAVs only search for the route to the EPC node. In contrast, in both DSDV and OLSR, each UAV must find routes to all other UAV, ground station, and EPC nodes. The second reason is that the convergence time of the two proactive routing protocols is determined by their topology update period. Shorter update periods can achieve faster convergence, but it an also generate more control overhead. Recall that in our simulation, we set the periodic update interval of DSDV to be 1 second, and we set the TC updating interval of OLSR to be 0.5 seconds.

Figure 7 illustrates the probability density function (PDF) of convergence times in the topology. We can see that, given the aforementioned simulation parameters, AODV achieves the shortest convergence time while OLSR requires the longest time to converge.

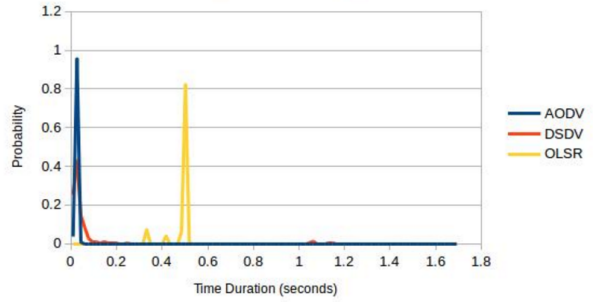

Figure 7. Probability distribution function of mesh routing protocol startup convergence times for all nodes to find a route to a designated EPC node.

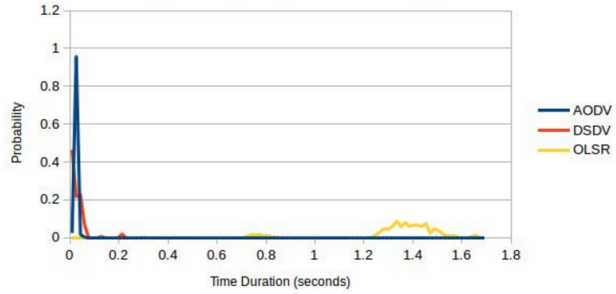

Figure 8. Probability distribution function of mesh routing protocol convergence times in repairing a single route upon link failure.

Link Failure—Figure 8 shows the PDF of convergence time for the three protocols. We can see that AODV also achieves the shortest convergence time due to its on-demand feature. In addition, we can see that DSDV usually converges in less than 200 msec in our simulated topology. In contrast, when the TC interval of OLSR is set to 0.5, its route recovery time is much longer than the other two protocols.

TS-SDN can establish and maintain a wireless mesh/backhaul topology that is more robust and stable because it computes a routing solution with global knowledge of the present and predicted future state of the network. Our simulation considered the use of a centralized controller for topology management, while relying on distributed mesh routing protocols for forwarding (absent TS-SDN). The results show a layer 3 disruption of the S1 interface between the airborne eNodeB and EPC of between 20 msec and 2 seconds depending on the choice of mesh routing protocol, *plus* disruptions due to beam steering and link establishment time, for every reconfiguration of the Distribution Layer topology. In contrast, TS-SDN can completely eliminate disruptions by predicting changes in link accessibility and proactively rerouting provisioned network flows to avoid disruption during reconfiguration of the Distribution Layer topology.

It is however still desirable to use distributed routing algorithms and protocols when a UAV node is disconnected from the controller in the case of an unpredicted link failure. A distributed routing algorithm can provide routing solutions for the network while the TS-SDN controller can learn from the current state of the network and re-program it for a better solution. This hybrid approach can minimize the service interruption and also avoid using expensive, out-of-band CNPC communications channels for network reconstitution.

8. Conclusions

This paper describes a variety of challenges posed by networks with nodes carried aboard UAVs. The challenges originate in the desire to manage the scarce network resources in the optimal fashion in the face of a range of disruptive, but generally predictable events, such as changes in vehicle position over time or the generally long time of wireless signal acquisition.

We discuss the constraints imposed in this environment on the network and its control mechanisms, and propose a solution based on extending SDN architecture to account for time-dynamic properties of the network.

We describe an actual SDN operating system built at Google. In particular, we note the role of the network data model as an API for SDN applications, provide examples of powerful network control logic implemented as SDN applications in our SDN operating system and describe some extensions required by temporospatial SDN in the CDPI protocol.

Finally, we contrast and compare our centralized SDN-based approach to network routing control with the distributed approach. We find that the unfavorable trade-offs between convergence time and cost in terms of network overhead make a TS-SDN approach generally preferable. However, distributed routing protocols constitute a viable fallback mechanism to help nodes recover from the loss of connectivity to the SDN controller.

References

[1] F. A. d'Oliveira, F. C. L. d. Melo, and T. C. Devezas, "High-altitude platforms - present situation and technology trends," *Journal of Aerospace Technology and Management*, vol. 8, no. 3, pp. 249–262, 2016.

[2] A. Vahdat, "SDN @ Google: Why and how," *Open Networking Summit*, 2013.

[3] O. Committee *et al.*, "Software-defined networking: The new norm for networks," *Open Networking Foundation*, 2012.

[4] S. Costanzo, L. Galluccio, G. Morabito, and S. Palazzo, "Software defined wireless networks: Unbridling SDNs," in *Software Defined Networking (EWSDN), 2012 European Workshop on*, Oct 2012, pp. 1–6.

[5] B. Barritt and W. Eddy, "Temporospatial SDN for aerospace communications," 2016.

[6] K. Raza and M. Turner, "Cisco network topology and design," *Retrieved November*, vol. 29, p. 2008, 2002.

[7] B. Barritt and W. Eddy, "SDN enhancements for LEO satellite networks," in *34th AIAA International Communications Satellite Systems Conference*, 2016, p. 5755.

[8] G. F. Riley and T. R. Henderson, "The ns-3 network simulator," in *Modeling and Tools for Network Simulation*. Springer, 2010, pp. 15–34.

[9] I. D. Chakeres and E. M. Belding-Royer, "AODV routing protocol implementation design," in *Distributed Computing Systems Workshops, 2004. Proceedings. 24th International Conference on*. IEEE, 2004, pp. 698–703.

[10] G. He, "Destination-sequenced distance vector (DSDV) protocol," *Networking Laboratory, Helsinki University of Technology*, pp. 1–9, 2002.

[11] P. Jacquet, P. Muhlethaler, T. Clausen, A. Laouiti, A. Qayyum, and L. Viennot, "Optimized link state routing protocol for ad hoc networks," in *Multi Topic Conference, 2001. IEEE INMIC 2001. Technology for the 21st Century. Proceedings. IEEE International*. IEEE, 2001, pp. 62–68.

BIOGRAPHY

Brian Barritt *is a software engineering Tech Lead/Manager at Google, where he has worked on building the Project Loon and Titan Aerospace networks. Prior to joining Google, Brian co-founded Alanax Technologies, a startup specializing in the modeling & simulation of aerospace networks. He has led successful engineering projects at Cisco and for NASA's Space Communications and Navigation (SCaN) program. Brian holds a BS and MS in Computer Engineering as well as an MBA, and he's currently a part-time Ph.D. Candidate at Case Western Reserve University.*

Tatiana Kichkaylo *is a software engineer at Google, where she worked on optimisation for temporospatial networks, modeling of physical systems, and control of steerable antennas. Prior to joining Google, she was a research scientist at USC ISI working on mission planning, systems engineering, and decision support systems. Tatiana holds a Ph.D. in Computer Science from New York University.*

Ketan Mandke *is a software engineer at Google, where he works on building wireless networks. Prior to joining Google, he implemented software-defined radio systems and developed signal processing, detection, and estimation algorithms for geolocation. Ketan holds a Ph.D. in Electrical Engineering from the University of Texas at Austin.*

Adam Zalcman *is a software engineer at Google, where he has worked on the Project Loon, led and launched a successful internal software project and worked on a number of distributed storage systems behind Drive and Gmail. Prior to joining Google, Adam developed operational spacecraft simulators for the European Space Agency and led an international team in GTOC4 (spacecraft trajectory design competition). Adam holds a MSc in Computer Science from Jagiellonian University in Krakow, Poland.*

Victor Lin *is a Tech Lead/Manager at Google, where he has worked on building various SDN networks for Google. Prior to joining Google, Victor worked on wireless network technology for various startup companies in Silicon Valley. Victor holds a MS in Computer Science from University of Minnesota, Minneapolis.*