

Aircraft Mesh Networks

Sam Crow
scrow@eng.ucsd.edu

Duong Nguyen
duong@eng.ucsd.edu

ABSTRACT

Aircraft flying over oceans, out of range of ground stations, must either use expensive satellite relays to communicate with the rest of the world or go without communication for long periods of time. To solve this problem, we propose a system of mesh networks to connect aircraft to distant ground stations. We simulate an aircraft mesh network and compare the OLSR and DREAM protocols. Our results show that either protocol may be an effective way to create mesh networks.

1 INTRODUCTION

Commercial aircraft have many ways to communicate with land-based transmitters and receivers. They support voice communication over VHF radio for air traffic control and navigation, text communication over ACARS for aircraft maintenance and dispatch information, and Internet access and phone service. These systems all work well for aircraft flying over land or near coasts. However, many flights travel across oceans, far outside the range of any standard land-based communication method. In these areas, aircraft use high-frequency (HF) radio for air traffic control. HF radio works over long distances but does not have enough bandwidth for more data-intensive communication tasks. The only current alternative is to communicate with satellites that relay messages to ground stations. Satellite communication works worldwide, but it can be very expensive due to the high cost of launching satellites.

In remote areas with dense air traffic, like the north Atlantic Ocean, aircraft could instead form a mesh network that would allow any aircraft to communicate with a ground station over a series of high-bandwidth, short-range communication links. This would reduce the costs of non-critical communications that would otherwise use satellites, and it could allow ADS-B transmissions from aircraft to be recorded on the ground. Figure 1 shows a possible mesh network that connects several aircraft to a few ground stations.

Because each aircraft knows its position and velocity, the aircraft can use this information to form a more stable network of aircraft moving in approximately the same direction. Multipath routing could make transmissions more reliable.

In this paper, we start by describing some related work and our basic design ideas for aircraft mesh networks. Next, we explain the two routing protocols that we have investigated.



Figure 1: A possible aircraft mesh network in the north Atlantic Ocean. Black circles represent aircraft, green circles represent ground stations, and lines represent communication links. The background map is based on data from the OpenStreetMap project, which is copyrighted by OpenStreetMap contributors.

Finally, we describe our simulations of the protocols and the results.

2 RELATED WORK

A few other projects have worked on similar topics involving aircraft communication. The most closely related project is Wi-Fly [1], which simulates how aircraft flying over land can use their existing Internet connections to provide Internet access to people on the ground below flight paths. Another paper, Simulating Large-Scale Airborne Networks with ns-3 [6], describes a similar network of aircraft flying over land and communicating with directional links. The authors developed a simulation based on ns-3 and tested the OLSR protocol.

3 DESIGN

We designed this networking system to take advantage of communication technology that is already installed on aircraft.

3.1 Addressing

Aircraft already have unique 24-bit ICAO addresses. These addresses are currently used to identify aircraft in ADS-B broadcasts and mode C transponder communication. Our simulations use ICAO addresses to identify aircraft and ground stations.

3.2 Network Layer and Physical Layer

Two existing communication standards describe network-layer communication methods that aircraft mesh networks could use. VHF Data Link (VDL) mode 2 allows aircraft to communicate with ground stations and exchange data. Because controller-pilot data link communication uses VDL mode 2, many aircraft implement it. VDL mode 4 was designed for communication between aircraft, which is essential for mesh networks. The International Civil Aviation Organization supports both of these standards.

4 PROTOCOLS

We investigated two protocols for potential use in aircraft mesh networks: Optimized Link State Routing (OLSR) and Distance Routing Effect Algorithm for Mobility (DREAM).

4.1 OLSR - Optimized Link State Routing

OLSR is a proactive protocol that attempts to find a route to each other aircraft before any packets are transmitted. It does not take advantage of position information. OLSR is described in [4]. This section provides a simplified explanation of the protocol.

OLSR is a link-state and selective flooding protocol. In OLSR, a node does not forward packets to all of its neighbors, only to a subset of its neighbor set. The selected neighbors are called multipoint relays (MPRs). To reduce the number of retransmissions and guarantee network coverage, OLSR chooses a minimal subset of one-hop neighbors that can cover all of its two-hop neighbors. Figure 2 shows the MPR nodes (in red) of node 4. Finding that set is a NP-hard problem [3], so OLSR uses a greedy algorithm instead.

In each loop (lines 3-6), a node of the one-hop neighbor set ($N_1(v)$) which connects to the highest number of uncovered two-hop neighbors will be selected to become the next MPR. The selecting process is repeated until all of the two-hop neighbors are covered. In general, each node in the network

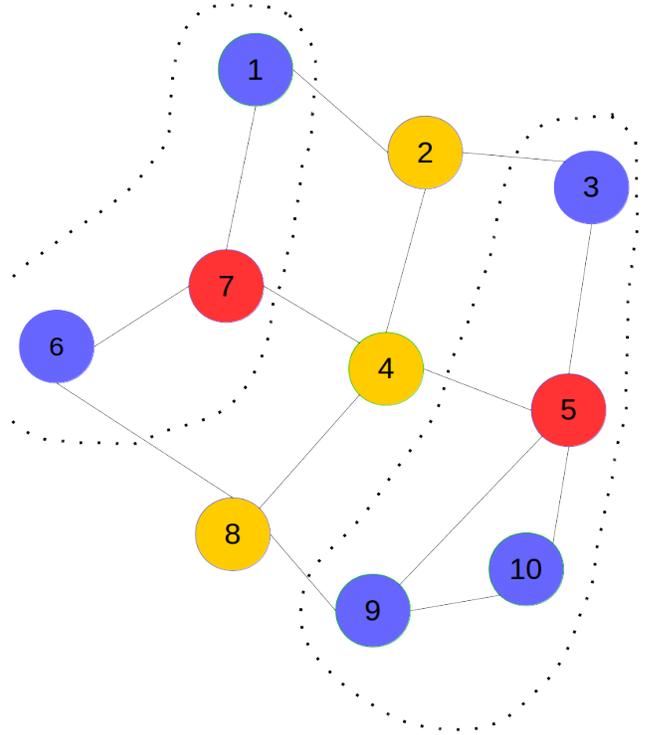


Figure 2: An illustration of the MPR set of node 4 in the OLSR protocol. The one-hop neighbors of node 4 are nodes 2, 5, 7, and 8. The MPR set contains nodes 5 and 7, which together provide connectivity to all the two-hop neighbors of node 4.

Algorithm 1: A greedy algorithm for finding an MPR set

Input: A network $G = (V, E)$ and node $v \in V$

Output: The MPR set of v

```

1  $MPRS \leftarrow []$  /* Initialize the MPR set */
2  $Uncov \leftarrow N_2(v)$  /* The uncovered two-hop neighbors of
    $v$  */
3 while  $Uncov \neq \emptyset$  do
4    $u \leftarrow \arg \max_{z \in N_1(z)} |Uncov \cap Adj(z)|$ 
5    $MPRS \leftarrow MPRS \cup \{u\}$ 
6    $Uncov \leftarrow Adj(u)$ 
7 return  $MPRS$ 

```

should have a neighbor table that contains information on its one-hop and two-hop neighbors. Each entry of the table has a holding time. When the node does not receive any

messages that update the entry, the holding expires and the entry is removed from the table.

Each node also has an MPR selector table, which keeps track of neighbors for which that node is a multipoint relay. The table also stores a sequence number which specifies the most recent MPR set. After a node updates its MPR selector table, the sequence number is increased by 1. Nodes periodically send messages with information from this table. Based on these messages, each node also builds a topology table, which stores the one-hop neighbors of every other node in the network. Each node uses the topology table to find a route to every other node.

4.2 DREAM - Distance Routing Effect Algorithm for Mobility

DREAM is a reactive protocol that makes routing decisions one packet at a time. It decides where to forward each packet based on the positions of the local node and the packet's destination. DREAM is described in [2]. This section provides a simplified explanation of the protocol.

In the dream protocol, we assume that each node stores other nodes' location and velocity. At time t_1 , the sender S receives a packet whose destination is R . Sender S will look at its local information to retrieve the last location (x_0, y_0) and velocity v of the destination (receiver R) that it obtained at time t_0 . Based on the information, S can estimate the area that R can travel from t_0 to t_1 . It should be a circle of radius $L = v(t_1 - t_0)$. Then S will forward the packet to its neighbors that are in the same direction as the destination. Figure 3 illustrates the method for selecting forwarders. The angle between the tangent line and the line SR can be calculated as $a = \arcsin \frac{L}{D}$ where D is distance between the sender and the receiver at time t_0 . Neighbors of S which are in the angle interval $[z - a, z + a]$ will be selected as the forwarders.

Figure 4 shows a special case when the distance that the receiver can travel during time $t_1 - t_0$ exceeds the distance between the sender and the receiver. This can happen if the velocity of the receiver is high or the sender's local information is out-of-date. In this case, the receiver can be in any direction at time t_1 and the sender should send the packet to all of its one-hop neighbors to guarantee that forwarding nodes can cover all possible positions of the receiver.

5 SIMULATION

To compare the performance of OLSR and DREAM, we created a packet-level simulation based on the ns-3 library.

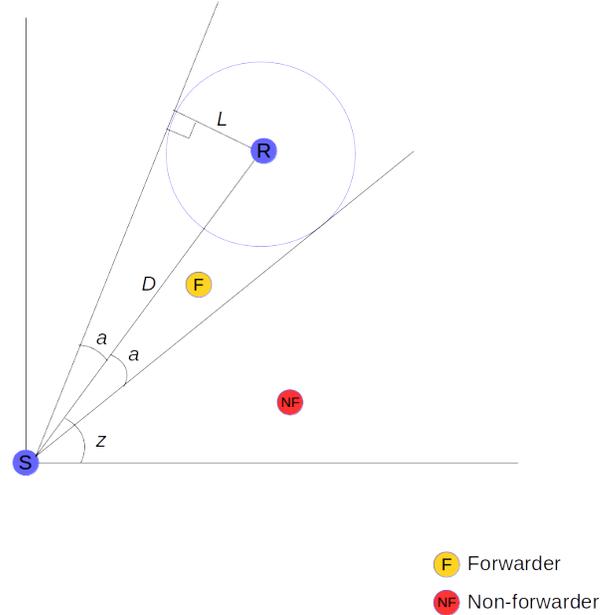


Figure 3: An illustration of selecting forwarders in the DREAM protocol.

5.1 Flight Data

To define the movement of aircraft in the simulation, we used flight tracking data from real-world flights. We downloaded records of many flights from FlightAware (<https://flightaware.com>) and selected 19 transatlantic flights to use in our simulation. Appendix A contains details about the selected flights.

The tracking information for each flight was provided by the website as a Keyhole Markup Language (KML) file. Each file contains the origin and destination airports, as well as a list of data points. Each data point contains a time and the latitude, longitude, and altitude of the aircraft. The list of points can contain gaps during periods when the aircraft was flying over an ocean, out of range of all ground-based tracking stations.

We created a simple C++ library to parse KML files into lists of points. Because the ns-3 components use a Cartesian coordinate system, our simulation converts the points from latitude, longitude, and altitude into an Earth-Centered,

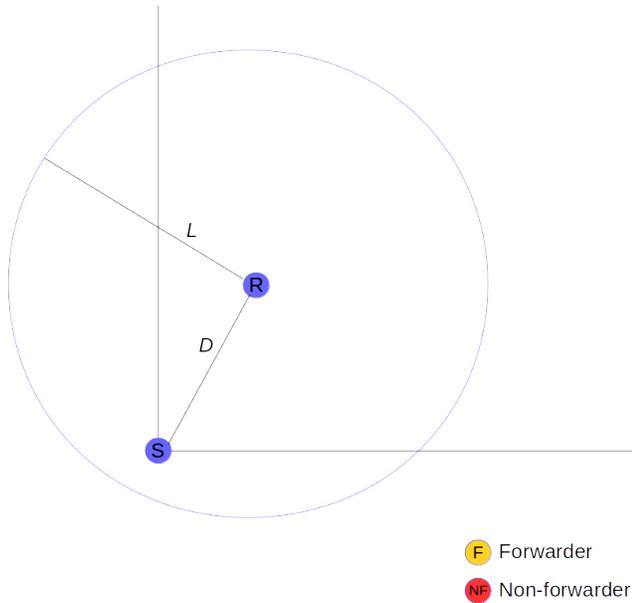


Figure 4: A special case happens when the maximum distance that the receiver can travel from time t_0 to t_1 exceeds the distance D . In this case the sender needs to forward the packet to all of its neighbors.

Earth-Fixed (ECEF) coordinate system. The ECEF coordinate system, as shown in figure 5, is a Cartesian coordinate system with its origin at the center of the Earth.

The simulation adds the converted points for each flight to an ns-3 WaypointMobilityModel instance. The mobility model calculates the position and velocity of each aircraft over time.

Because the waypoint mobility model interpolates between points linearly in the ECEF coordinate system, aircraft do not follow the curvature of the earth during gaps in the source data. This causes aircraft to sometimes have lower altitude than in reality. Because the gaps in the data are relatively small and the altitudes of aircraft are small relative to the range of communication, we expect that this error will not have a significant effect on the results.

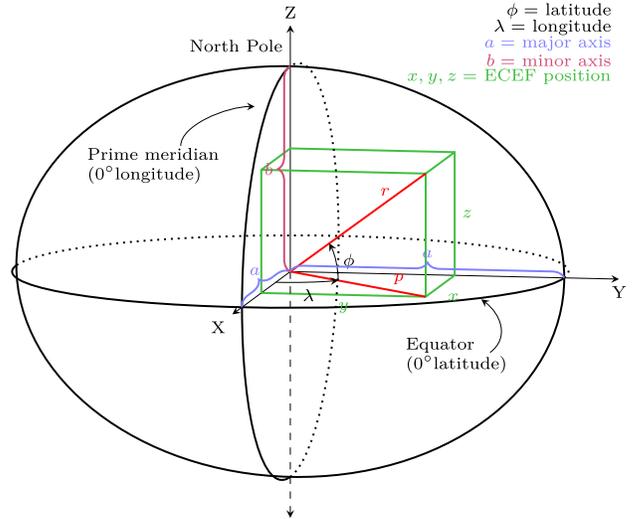


Figure 5: An illustration of the Earth-Centered, Earth-Fixed coordinate system. This image is used with permission from [5].

5.2 Ground Stations

Although our simulation software supports many ground stations, for the tests described here we created one ground station near Reykjavik, Iceland. Several of the flights in the simulation fly near this ground station.

5.3 Addressing

The aircraft and ground stations in the simulation use 24-bit ICAO addresses. The downloaded flight information does not specify the addresses of the aircraft, so the simulation assigns addresses to flights sequentially. The ground station has the address 0x800000.

5.4 Applications

Each aircraft or ground station can have one or more associated applications. The simulation sets up an ADS-B sender application for each aircraft. This application periodically sends four bytes of data to the ground station.

5.5 Network Layer

In each simulation, each aircraft and ground stations uses either OLSR or DREAM as the network-layer protocol connecting the ADS-B sender application to the network interface.

5.6 Link Layer

The link layer simulation connects the network interfaces on all aircraft and ground stations. When a network interface

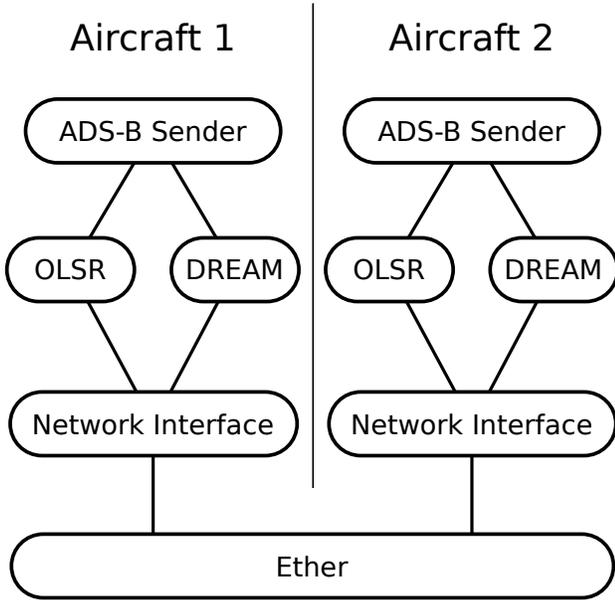


Figure 6: The topology of two aircraft nodes in the simulation. All nodes in each simulation run use either OLSR or DREAM.

sends a packet, the link layer delivers the packet to all other network interfaces that are within range. The packets arrive after a delay that is calculated based on the speed of light and a data rate of 100 000 bits per second.

5.7 Output

The simulation records information with two different subsystems. The first subsystem periodically records the location of each aircraft and ground station. When the simulation is using OLSR, the subsystem also records each OLSR routing table. After the simulation ends, the subsystem writes its recorded information to a JSON file. The second subsystem records the send time and receive time of every packet. It categorizes each packet as a management packet, used to maintain the structure of the network, or a data packet that contains actual information to be transmitted. It saves the recorded information in a CSV file.

6 VISUALIZATION

We created an application to visualize the output of the simulation. This application reads a JSON file produced by the simulation that contains information about aircraft positions and routing. It uses the World Wind library to display an interactive 3-dimensional model of the Earth, with the path of each flight overlaid. Each orange line shows a

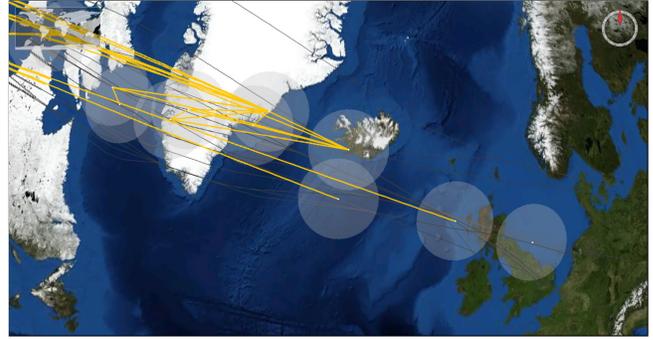


Figure 7: The main window of the visualization application, showing several flights and routes

connection between an aircraft and the next hop on a route. A slider at the bottom of the window allows the user to navigate through time and see how the routes change.

7 RESULTS

7.1 OLSR Range Comparison

To see how communication range effects performance, we ran three simulations using OLSR with communication range values of 300, 600, and 900 kilometers. Figure 8 shows the numbers of data packets that arrived and the number of data packets that were lost for the three different range values. Figure 9 shows the results for management packets.

As expected, with longer communication range more packets were received. For data packets, the relationship between range and number of packets received is approximately linear. Because data packets are only sent to the one ground station, the packets can be delivered from more distant aircraft. The fact that each aircraft may be within range of a larger number of other aircraft does not affect data packets. However, for management packets, changing the range from 300 to 600 kilometers results in a larger increase in delivered packets than changing the range from 600 to 900 kilometers. Because OLSR management packets are broadcast to all aircraft in range, the first range increase allows each aircraft to successfully send more management packets. Increasing the range to 900 kilometers only caused a small increase in the number of other aircraft within range of each aircraft.

7.2 OLSR and DREAM comparison

To compare the performance of the two protocols, we ran one simulation with OLSR and one simulation with DREAM. Both simulations used a communication range of 300 kilometers.

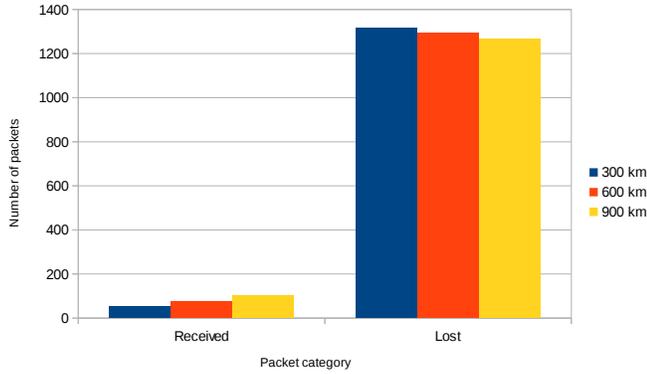


Figure 8: The numbers of data packets received and lost using OLSR with different range values

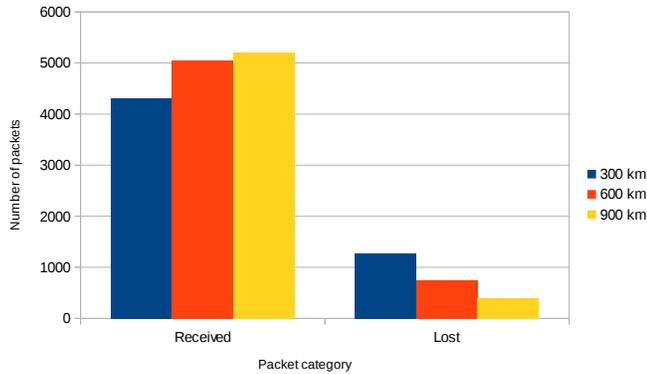


Figure 9: The numbers of management packets received and lost using OLSR with different range values

Table 1: Packet counts from the two simulations comparing OLSR and DREAM

Protocol	Data		Management	
	Received	Lost	Received	Lost
OLSR	51	1317	4308	1278
DREAM	53	1315	5197	2103

Packet delivery. Table 1 shows the numbers of packets that were received and lost. Figures 10 and 11 show graphs of the results. The DREAM protocol delivered a few more data packets than OLSR. The DREAM protocol sent more management packets, and had a slightly higher portion of lost management packets.

Because DREAM is a reactive protocol, we expected it to have low overhead and send relatively few management

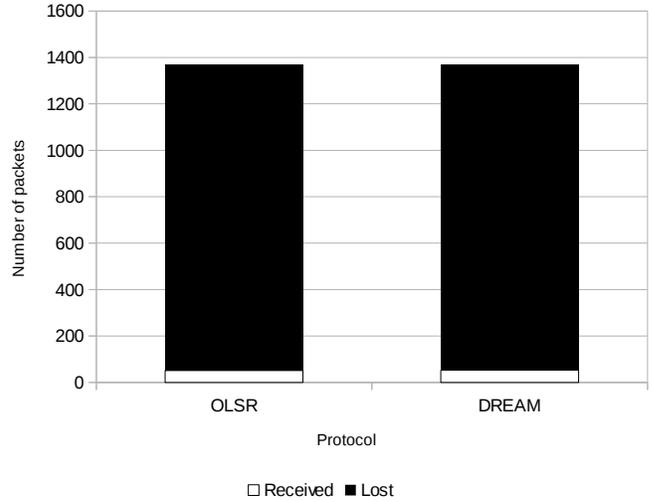


Figure 10: A comparison of the number of data packets received and lost for the two protocols

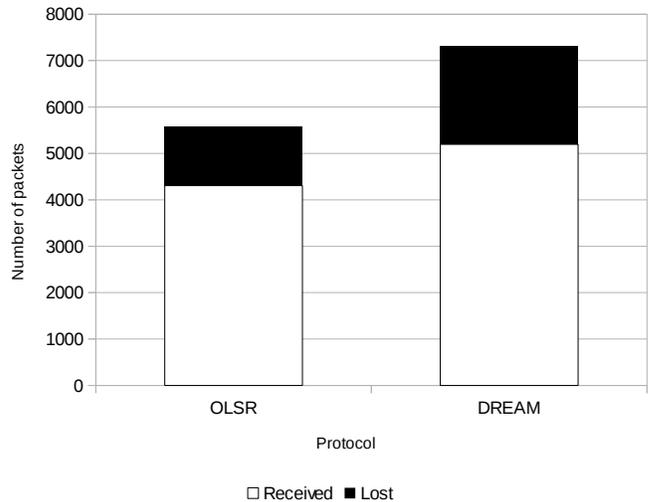


Figure 11: A comparison of the number of management packets received and lost for the two protocols

packets. In these simulations, however, it sent more management packets than OLSR. The number of management packets sent is heavily influenced by the amount of time each aircraft waits between sending management packets. We chose initial values for these time intervals somewhat arbitrarily without trying to optimize performance. With better tuning, we expect that the DREAM protocol would send fewer management packets.

With both protocols, only a few percent of all data packets arrived. Because this simulation involves only one ground station and nineteen flights, the density of aircraft is only

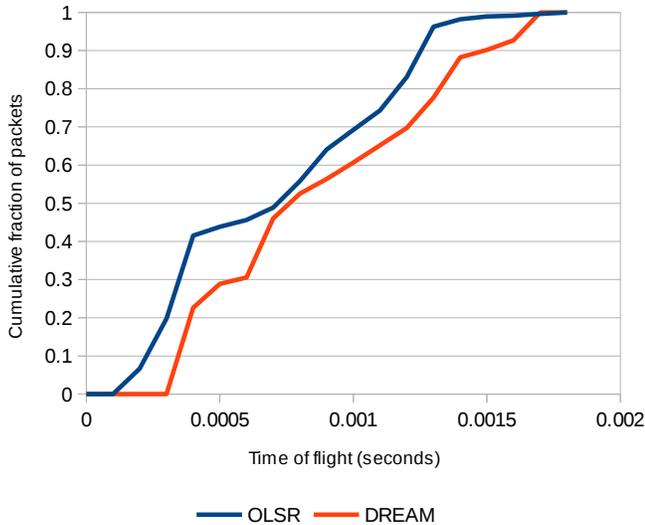


Figure 12: CDF of the time of flight for all received packets

rarely high enough to allow more than two aircraft to connect to each other. A more detailed simulation, with more aircraft and more ground stations, would allow the network to perform better.

Packet time of flight. Figure 12 shows the cumulative distribution function of the time between when each packet was sent and when it was received.

Because light can travel 300 kilometers in one millisecond, an empty packet would take about one millisecond to travel the maximum communication range of 300 kilometers. The results are consistent with this assumption. Some packets that are sent over short distances arrive in less than one millisecond, while some packets containing large amounts of data take more than one millisecond to send.

Packets sent by the DREAM protocol consistently take slightly longer to arrive than with OLSR. DREAM management packets include the position and sometimes the velocity of the sender, which take up large amounts of space. OLSR management packets can sometimes be smaller because they include addresses instead of position information. The number of addresses in a DREAM hello or topology control message depends on the number of neighbors of the sending node. In these simulations, each aircraft had a small number of neighbors. OLSR most likely created smaller management packets overall, allowing them to be received sooner.

8 FUTURE WORK

8.1 Simulation Scalability

Our simulation results so far are based on only 19 of the 557 transatlantic flight records that we have available. When we tried to run a simulation with more aircraft, the simulation proceeded very slowly and, after a few minutes, consumed about ten gigabytes of RAM. The simulation software has some limitations that make it scale poorly to large numbers of aircraft.

The most well-understood limitation is caused by the module that delivers packets from one aircraft to the other aircraft in range. For each packet, the simulation iterates over all other aircraft and calculates the distance from the sender to check if each aircraft is in range. If each aircraft regularly sends packets, the total time spent delivering packets is proportional to the number of aircraft squared. With a more optimized method of delivering packets, the simulation would be able to handle more aircraft.

We do not fully understand why the simulation uses several gigabytes of memory when simulating large numbers of aircraft. One possible cause is a memory leak. We tested the simulation with 19 aircraft under `valgrind` to look for leaks and found no obvious problems. Another possibility is that some of the simulation data structures are storing large amounts of unnecessary information. We have not yet found any problems, but we are continuing to investigate.

8.2 Ground Stations

Our simulations so far include only one ground station, which does not reflect reality. With more ground stations, ideally based on the positions of real-world ADS-B receivers and antennas used for in-flight Internet access, the simulation would allow each aircraft to communicate with the nearest ground station.

8.3 Simulation Accuracy

Physical Layer. The simulation software has a very basic model of the physical layer. Every packet is sent immediately and arrives after a delay based on the speed of light, the length of the packet, and a preset bit rate. The simulation allows each aircraft to send and receive an unlimited number of packets at the same time. When a packet is sent, it always arrives at all aircraft in range, with no possibility of noise or interference blocking the transmission. In general, the simulation currently ignores congestion and non-ideal behavior, which are important factors for producing accurate results.

Table 2: The 19 flights used in the simulations

Airline	Flight	Departure date
Air France	84	2018-03-16
British Airways	285	2018-03-16
British Airways	287	2018-03-16
Lufthansa	454	2018-03-16
Lufthansa	458	2018-03-15
Lufthansa	458	2018-03-16
Aer Lingus	147	2018-03-16
KLM	281	2018-03-16
Scandinavian (SAS)	935	2018-03-16
Swiss	38	2018-03-16
Turkish	79	2018-03-16
United	59	2018-03-16
United	900	2018-03-16
United	949	2018-03-16
United	955	2018-03-15
United	984	2018-03-16
Virgin Atlantic	19	2018-03-16
Virgin Atlantic	41	2018-03-16
WOW Air	161	2018-03-16

OLSR Routing. By looking at the visualization of the routes that the OLSR protocol finds (figure 7), we observe that many aircraft have active routes to other nodes that are too far away for communication to possibly succeed. These routes should not last as long as they do in the simulation. Our implementation of OLSR may have a bug that causes this behavior.

9 CONCLUSION

In this paper, we have introduced our ideas aircraft mesh networks. We identified the Optimized Link State Routing and Distance Routing Effect Algorithm for Mobility protocols as potentially useful. We simulated both protocols using real-world flight data and compared their performance. Based on our simulation results, both OLSR and DREAM may be effective routing protocols for aircraft mesh networks. Although we do not have enough information to determine which protocol works better, our simulation software is a useful foundation for continuing research in this area.

A FLIGHTS USED IN SIMULATION

The simulations used 19 transatlantic flights departing from several airports in Europe and arriving at the San Francisco Airport, California. Table 2 shows the flights.

B SOURCE CODE

Our simulation software and visualization application are open-source. The code is available at <https://github.com/CSE202A-w2018-AMN>.

ACKNOWLEDGMENTS

The ns-3 and World Wind open-source projects provided libraries that allowed us to create our simulation and visualization software easily and within the available time. We would also like to thank FlightAware for providing flight tracking information in a convenient file format.

REFERENCES

- [1] Talal Ahmad, Ranveer Chandra, Ashish Kapoor, Michael Daum, and Eric Horvitz. 2017. Wi-Fly: Widespread Opportunistic Connectivity via Commercial Air Transport. In *Proceedings of the 16th ACM Workshop on Hot Topics in Networks (HotNets-XVI)*. ACM, New York, NY, USA, 43–49. <https://doi.org/10.1145/3152434.3152458>
- [2] Stefano Basagni, Imrich Chlamtac, Violet R. Syrotiuk, and Barry A. Woodward. 1998. A Distance Routing Effect Algorithm for Mobility (DREAM). In *Proceedings of the 4th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '98)*. ACM, New York, NY, USA, 76–84. <https://doi.org/10.1145/288235.288254>
- [3] Alberto Caprara, Paolo Toth, and Matteo Fischetti. 2000. Algorithms for the Set Covering Problem. *Annals of Operations Research* 98, 1 (01 Dec 2000), 353–371. <https://doi.org/10.1023/A:1019225027893>
- [4] P. Jacquet, P. Muhlethaler, T. Clausen, A. Laouiti, A. Qayyum, and L. Viennot. 2001. Optimized link state routing protocol for ad hoc networks. In *Proceedings. IEEE International Multi Topic Conference, 2001. IEEE INMIC 2001. Technology for the 21st Century*. 62–68. <https://doi.org/10.1109/INMIC.2001.995315>
- [5] Krishnavedala. 2014. ECEF.svg. (2014). <https://commons.wikimedia.org/wiki/File:ECEF.svg> File: ECEF.svg.
- [6] Ben Newton, Jay Aikat, and Kevin Jeffay. 2015. Simulating Large-scale Airborne Networks with Ns-3. In *Proceedings of the 2015 Workshop on Ns-3 (WNS3 '15)*. ACM, New York, NY, USA, 32–39. <https://doi.org/10.1145/2756509.2756514>