DTN ROUTING PROTOCOLS FOR DRONE SWARM TELEMETRY

Jason R. Brown and Justin P. Rohrer
{jrbrown1|jprohrer}@nps.edu
Department of Computer Science
Naval Postgraduate School Monterey, CA, 93943
https://tancad.net/

ABSTRACT

Drone swarms pose a particular challenge to telemetry networks, due to the number of airborne nodes involved, and their potential to overwhelm the available bandwidth on the communications channel with simultaneous telemetry streams. Previously, we saw that mobile ad-hoc (MANET) routing protocols could exacerbate this issue by flooding the network with routing-control packets. In this work we model the Naval Postgraduate School fixed-wing drone swarm and compare the performance of several disruption-tolerant networking (DTN) routing protocols designed to address these challenges.

INTRODUCTION

This work models the network formed by a swarm of fixed-wing aerial vehicles developed at the Naval Postgraduate School (NPS) and compares the performance of several Disruption Tolerant Network (DTN) routing protocols, observing tradeoffs between the potential routing protocols for the current swarm configuration to select candidates for future larger-scale swarms. In previous work, we performed this comparison using Mobile AdHoc Network (MANET) routing protocols, and found that the density of nodes in the radio communication environment posed a significant challenge to those protocols [1].

In 2015, a group of NPS professors and students set the record for largest fixed-wing unmanned aerial vehicle (UAV) swarm flown at one time [2]. The swarm had 50 vehicles flying simultaneously and successfully demonstrated distributed decision-making with all processing occurring on swarm vehicles rather than a centralized control station. This swarm uses broadcast messaging, and no multi-hop routing is implemented.

While this approach provides low-latency communications when all nodes are within range of each other, it introduces two problems. The first is that as the number of nodes increases, it will eventually saturate the communication channel, since no spacial-reuse is possible with this scheme. Early signs are that this will occur with swarm sizes not much larger than the current 50 nodes. Secondly, future mission tasking is likely to require distributing nodes geographically such that not all are in communication range of each other, as well as sub-swarm missions where the swarm splits temporarily and later regroups.

BACKGROUND AND RELATED WORK

In this section we examine the existing state-of-the-art in drone-swarm research.

A. Swarm Networking Issues

Drones and drone swarms may move rapidly with respect to their radio range, causing them to frequently lose and regain connectivity to ground stations and other vehicles. In the case of the NPS drone swarm, the cruising speed is 18 m/s. Currently, the NPS swarm uses no multi-hop routing protocol and instead uses 802.11n infrastructure-mode wireless. Messages to drones are addressed to the IP broadcast address application-layer message header determines whether a specific node should receiver or drop each message. Depending on future mission planning, keeping all nodes within range of the ground station may not be desirable, leading to the need to multi-hop messages from more distant nodes back to the ground station.

In addition to network and routing layer limitations, prior work has shown TCP to perform poorly in lossy wireless environments [3], and this effect is exacerbated with each additional wireless link in the path. The TCP routing protocol was designed for long-lasting connections along an established path. Attributes of TCP, like the three-way handshake, slow start algorithm, and congestion control algorithm were not designed for scenarios where packets are lost due to causes other than congestion, or where paths are frequently broken. UDP is not encumbered by the same restraints as TCP, but offers no acknowledgment that packets were delivered to the destination [4].

Channel bandwidth is also an ever-present constrain in telemetry systems, and can easily be overwhelmed with 10s or 100s of nodes attempting to communicate simultaneously. On one hand, multi-hop routing may help with this, by permitting more spacial reuse in the channel allocation plan, however overhead from routing protocol control messages can also consume significant amounts of bandwidth.

B. DTN Routing Protocols

The following routing protocols were employed for comparative analysis:

- DSDV: a proactive distance-vector based MANET routing protocol [5].
- AODV: a reactive distance-vector based MANET routing protocol [6].
- OLSR: a proactive link-state MANET routing protocol [7].
- Epidemic: Simple flooding-based DTN routing protocol [8].
- Vector: Epidemic-based routing protocol that limits flooding based on direction of travel [9].
- Gapr: Geolocation-assisted predictive DTN routing protocol [10].
- Gapr2: Improvement on GAPR that reduces message replication [10].

For the sake of space, we do not go into the details of each protocol, but refer you to their respective source documents.

METHODOLOGY

This section provides a review of the initial simulation built to model the NPS swarm in ns-3, as well as modifications made by this study to improve the mobility model and test additional routing protocols. Ns-3 is a discrete-event network simulator intended to implement the entire network stack as a platform for testing a variety of real-world networking scenarios [11]. The ns-3 Consortium developed it with the intent to improve research and education tools, and recent work shows that when compared with the ONE simulator, ns-3 provides a more robust, real-world implementation of various networking aspects [12].

Table 1: Message Parameters

Message	Data Rate	Packet Size	Frequency
Heartbeat	160 bps	20 Bytes	1 Hz
Flight Status	768 bps	48 Bytes	2 Hz
Pose	4480 bps	56 Bytes	10 Hz
Red Pose	3200 bps	40 Bytes	10 Hz
Asynchronous	as required	variable	as required

C. Traffic Generation

Four messaging applications were modeled in the initial NPS swarm simulation. Each of the applications comprised a subset of the synchronous messages described in [1] and all were sent as broadcast messages with specific non-changing sizes and data rates. Table 1 depicts these messaging applications' characteristics with their respective data rates, message sizes and transmission frequencies.

In the initial study all messages, regardless of type, included a standard 16-Byte header added at the application layer. Messaging applications were started at staggered times in order to ensure that collisions were not generated by artificial synchronization. In order to employ DTN routing protocols, a DTN header is employed at the application level. Ns-3 includes various application helpers, to accommodate varying application types. Originally, the "OnOffHelper" class is used to allow a messaging application to send at its assigned transmission and data rates. For the DTN version of the applications, this study employed the "DTNHelper" class, allowing the selected routing protocol to implement a custom-built 18-Byte DTN header as described in [13].

Table 2: Swarm Configurations

Mobility Model	Ground Station	Sub-swarms	Sub-swarm nodes	Simulation nodes
6 v 6	1	2	6	13
10 v 10	1	2	10	21
15 v 15	1	2	15	31
25 v 25	1	2	25	51

D. Node Mobility

The initial NPS swarm implementation in ns-3 utilized real-world captured data from a flight demonstration of the NPS swarm in 2015. Table 2 represents the breakdown of each flight. One of the results of the demonstration was a corpus of node location data corresponding to each "mobility model". The captured location data was a comma-separated document with each line including a time stamp, swarm identification number, node identification number, x-position, y-position and z-position. The initial ns-3 implementation used this data as input to a mobility model scheduler, employing the constant position mobility model such that each node would start at the positions in the line corresponding to that node's first entry in the real-world data, and remain stationary at that position until the time of its next entry and "hop" to the new position.

While employing a constant position mobility model is an effective method for implementing a simple mobility schedule, it does not adequately describe the flight motion of highly mobile nodes in an aerial scenario. Various other methodologies exist in ns-3 to implement many mobility scenarios, from which this study chose to utilize the "Ns2-mobility-helper". The helper reads in a file that requires a specific format, allowing three possible statements, of which this study used two. The real world flight data was parsed and formatted into the proper format prior to utilization in the simulations. The required format statements utilized were:

Initial Position: `node n set [XYZ] [xyz]`
Set Destination: `ns at time t node n setdest x2 y2 z2 speed`

In the Initial Position command, n is the node id, [XYZ] tells the simulator which position component is being read next and [xyz] is the actual component position. After the initial node position was set, all subsequent node positions were read in as Set Destination commands.

Ns-3 documentation and the Ns2-mobility-helper require that the format not include the z value; however, modifications were made to the mobility helper such that the z value could be read in and incorporated in the subsequent scheduling. Since the original NPS swarm data corpus did not include a velocity or speed field, this study used a standard cartesian speed calculation (distance/time) to acquire an average speed between positions one and two for the node being read in. Originally, the output of this calculation resulted in multiple consecutive lines in the mobility input file with the same position and zero-velocity, effectively creating a similar stop-and-wait mobility scenario as the initial simulation.

A finding of this study is that the real-world data was provided at such a fine time-precision that the provided positions did not have enough refinement to distinguish a unique position across two subsequent records. In other words, a given node reported the same position twice in a row (with different time stamp) various times throughout the trace. In the final implementation, this was accounted for by skipping over the first line used to calculate the ns2 format's required "setdest" information during formatting if and only if it was the same as the node's following position. In this way, an accurate velocity was obtained, and constant 3d motion could be implemented in the scheduler by the mobility helper. A final note worth mentioning is that because the data came from real-world flight records, the timestamps, while ordered correctly, started at different values for each flight. Ns-3 does not enforce a hard requirement that the initial positions be set at time 0.0 for obvious reasons; however, when using the provided time stamps, the simulation never reached the start time for mobility, so no nodes were ever set to a starting position before network activity began. This was fixed by offseting every ns2 mobility line time stamp by the value of the first record time stamp.

E. Experiment Parameters

The following parameters were varied to exploit the various features of the routing protocols used:

- Node count: the number of nodes in a given swarm. Values: 12, 20, 30, 50.
- Radio gain: the relative gain (integer value) of the 802.11n radio simulated. Values: 0, 3, 6, 9, 12, 20.
- Mobility model: The type of mobility scheduled in ns-3. Values: Constant Position, Ns2 Helper.

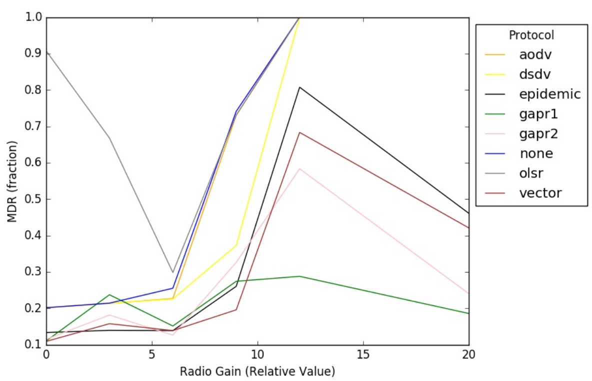
F. Experiment Metrics

The following metrics are utilized for quantitative comparison of protocol performance:

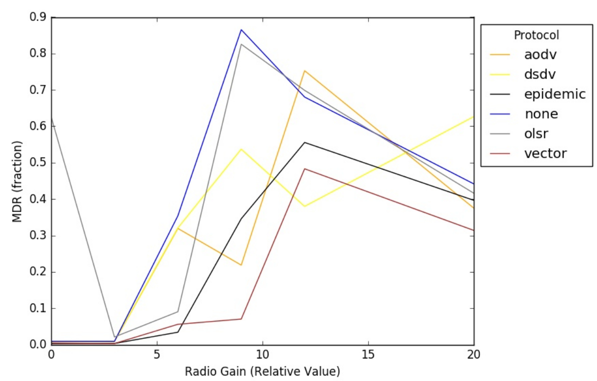
- Message Delivery Ratio (MDR): calculated by dividing the number of messages delivered to the assigned destination by the number sent.
- Goodput: usable transmitted application data successfully received at the assigned destination.
- Latency: time delay calculated as time of receipt at destination minus time of transmission.
- Overhead: message copies, retransmissions, and routing control messages.

RESULTS AND ANALYSIS

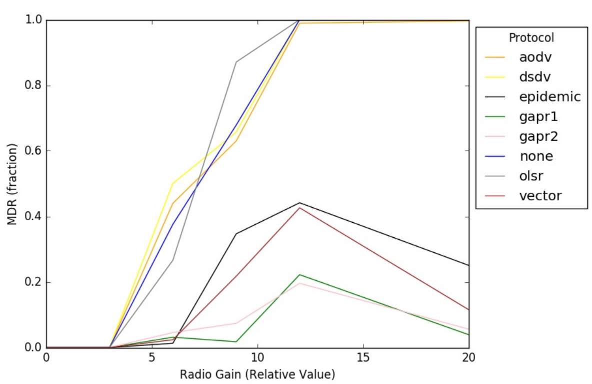
This section reviews the results from the 650 simulations and highlights a number of illustrated trends and relationships. It should be noted that in running simulations, both Gapr and Gapr2 simulations were cut off around one-half of the total run time, after the simulations ran for 72hrs each, and their 10v10 results were not captured due to simulator error. For this reason, Gapr and Gapr2 results will be displayed where available, however, further results will be measured at a later time. In any case, early results show general consistency with the other DTN protocols.



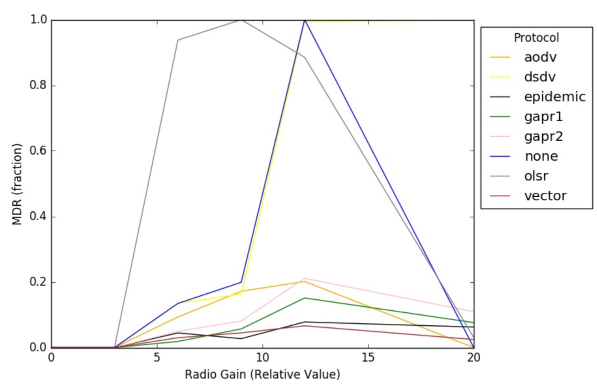
(a) 6 v 6 Swarm



(b) 10 v 10 Swarm



(c) 15 v 15 Swarm



(d) 25 v 25 Swarm

Figure 1: Message Delivery Ratio

The initial NPS swarm simulation measured PDR (Packet Delivery Ratio) as a function of the individual messaging application, and showed that generally, broadcast is the most effective method of transmission with the swarm scenario across all applications. Since the aim of this study was to elaborate on that work, and to provide a baseline for future work in studying DTN protocols through the NPS swarm scenario, the results of this work will be discussed in a broader sense. As a function of gain and swarm size, MDR was recorded to measure each routing protocol's performance.

One of the most significant findings of the swarm simulation MDR output is that broadcast (”none”) consistently and significantly outperforms almost all of the employed routing protocols across all swarm sizes and gains. Additionally, when comparing MANET and DTN routing protocols, for this scenario the MANET routing protocols almost always have a higher MDR than the DTN protocols. The performance of the MANET routing protocols is constistent with previous work [1], so it is unlikely that the use of a more realistic mobility model significantly influenced the performance of the protocols. Alternatively, since MANET results are consistent, it is safe to assume that the new mobility model worked effectively, and the DTN protocols simply underperform when compared with the MANET protocols, in general, for the swarm scenario employed.

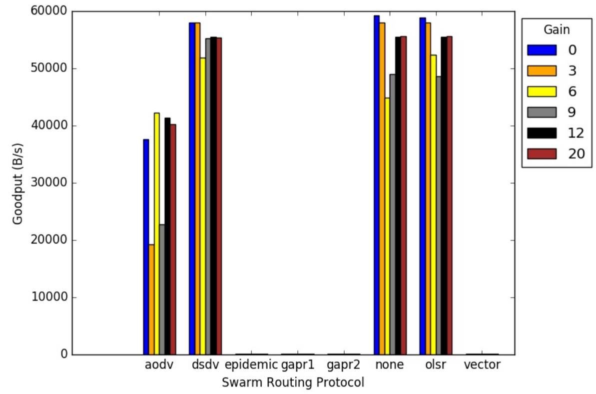
A final observation on MDR is that these outputs show that as the node density increases, complexity also increases, as evidenced by the downard trend in protocol performance across increasing swarm sizes. The increased complexity in neighbor and route discovery makes it significantly more difficult for both the MANET and DTN protocols to operate at a high capacity, however it seems to have a slightly more significant impact on the DTN protocols. As has previously been mentioned, the GAPR scenarios all ran for more than 72hrs without exceeding one half of the total simulation time (their trace files end around 500s in the farthest case).

G. Goodput

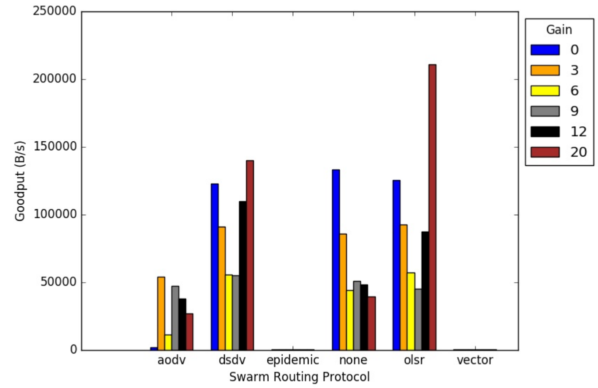
Consistent with MDR performance, MANET protocols performed better than DTN protocols for this scenario in terms of goodput. Additionally, as with MDR, the broadcast scenario (”none” protocol) showed high goodput performance, which implies that for all swarm sizes the locations of the drones were close enough that inter-node routing was not required for message delivery to the destination. For that reason, it is not entirely unexpected that the DTN protocols under-performed in comparison to the MANET protocols. Since inter-node routing was often not necessary, there is no reason to impose the overhead and delay of routing decision-making techniques required for DTN protocols. While not unexpected, it is still unfortunate that DTN performance in terms of goodput was so low, as it is hard to make a clear judgment of the DTN protocols' performance in general other than to note that the MANET protocols performed so much better.

H. Latency

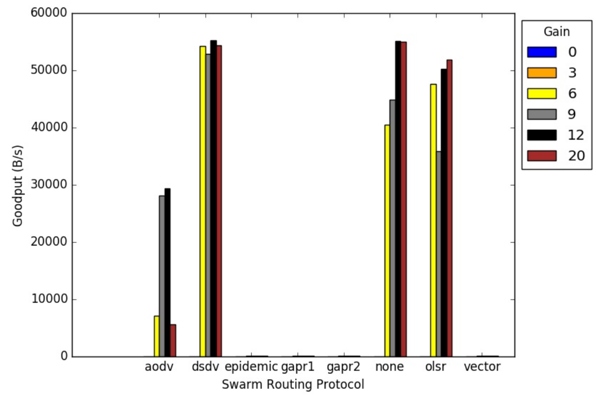
Another useful metric for studying DTN performance is latency, or delay in delivering messages. For this study, latency was measured as a cumulative value for each simulation (total sum of latency for every message delivered). Consistent with MDR and goodput measurements, the DTN protocols have a significantly higher latency in general than the MANET protocols. It is likely that latency in each message delivery is a direct contributor to the poor MDR and goodput observed for the DTN protocols. Conversely, the low latencies recorded for the MANET protocols



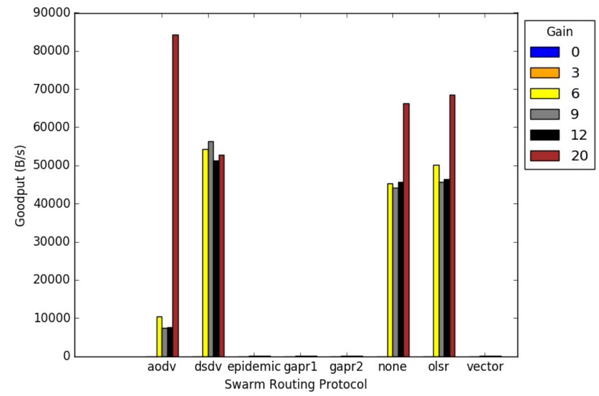
(a) 6 v 6 Swarm



(b) 10 v 10 Swarm



(c) 15 v 15 Swarm



(d) 25 v 25 Swarm

Figure 2: Goodput

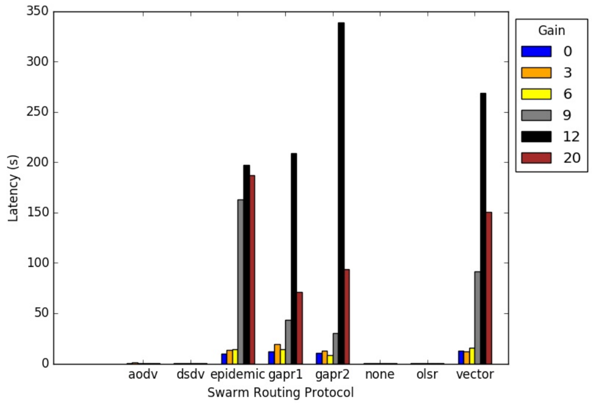
are consistent with their better MDR and goodput performance.

I. Overhead

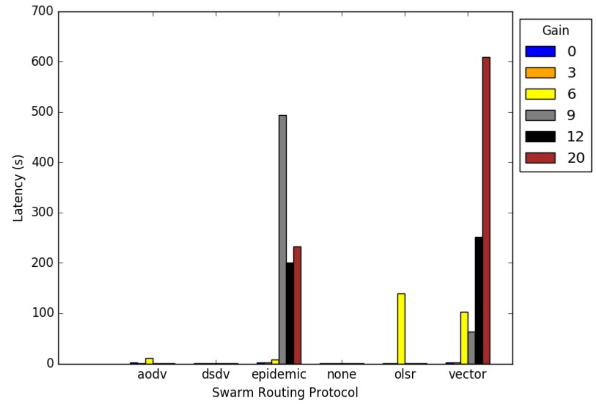
All of the measured protocols displayed significant overhead. As a record of total overhead exercised throughout each scenario, all DTN protocols approached the Megabyte range, with the exception of Gapr, Gapr2 and Vector in the 6v6 scenario. The poor performance in terms of overhead is not entirely unexpected, as DTN routing protocols exhibit the behavior of frequent beaconing to maintain connectivity, and the NPS Swarm in particular has high messaging rates at the application level on top of the routing protcol's native behavior. Additionally, since overhead was measured as an aggregate value for each total scenario, the Gapr and Gapr2 overheads will likely increase as further testing progresses.

CONCLUSIONS AND FUTURE WORK

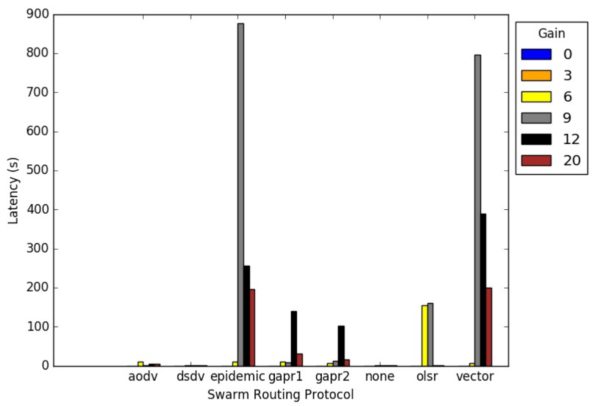
This study extended previous work to implement real-world mobility data from the NPS swarm live-fly exercise in 2015, in order to create a testbed of high-mobility platforms for Disruption (and Delay) Tolerant Networks. By employing a mobility model helper that assigned nodes movement information in the form of destination locations and speed, this study was able to increase the realism of the mobility model for more robust testing. Implementation of DTN routing protocols



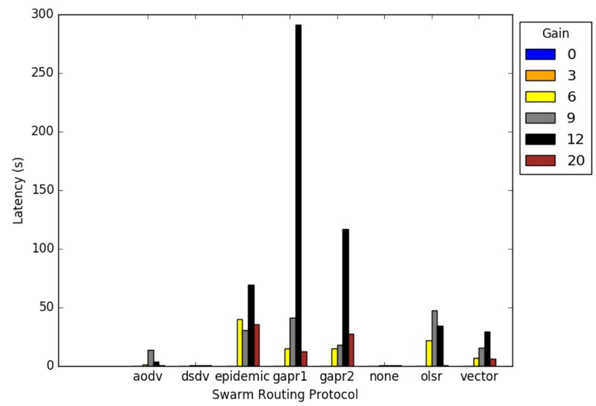
(a) 6 v 6 Swarm



(b) 10 v 10 Swarm



(c) 15 v 15 Swarm



(d) 25 v 25 Swarm

Figure 3: Overall Latency

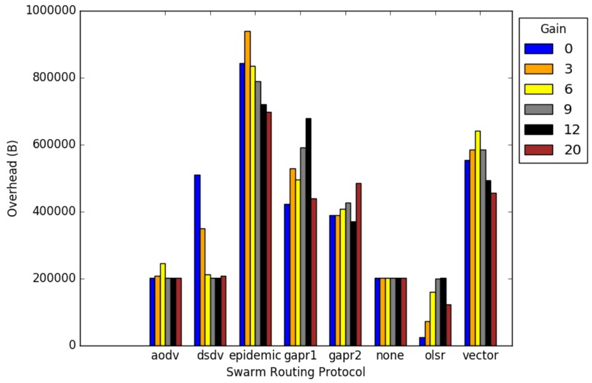
revealed moderate performance, in comparison to the previous work done with the NPS swarm data.

Future work to follow up this study will need to improve performance of simulations for the Gapr family of protocols. An extension of this work would be to implement and test further DTN routing protocols that exhibit various different behaviors not encompassed in Epidemic, Vector or either of the GAPR protocols.

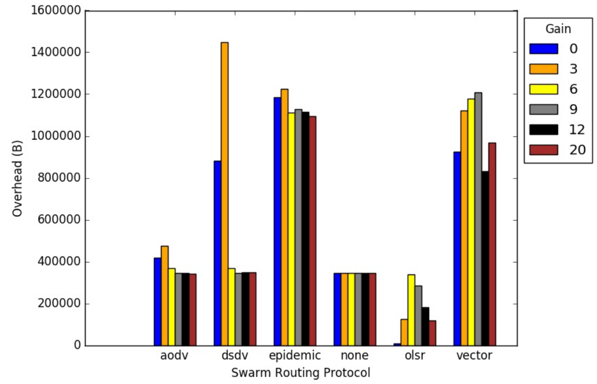
Finally, more robust testing will be helpful for the progression of this study. In the future, more testing will reveal more accurate characteristics of the DTN protocols. Additionally, having more variation of node types and movements in the corpus of real-world data will provide a more useful testbed for evaluating routing protocols for future swarms.

ACKNOWLEDGMENTS

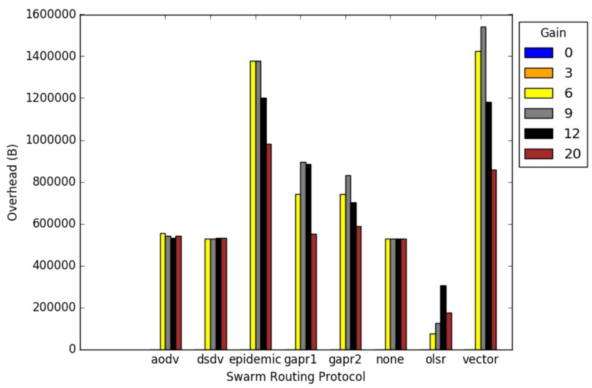
Special thanks to Maj. Matthew Hopchak and Lt. Nicholas Davis for their contributions to this study. Their many long hours and significant effort in testing, coding and debugging to ensure that all of the ns-3 modules worked together to perform consistent, repeatable behavior was critical to the completion of this work. This effort would also not have been possible without the prior work of Lt. Andrew Mauldin creating the DTN routing protocol models in ns-3.



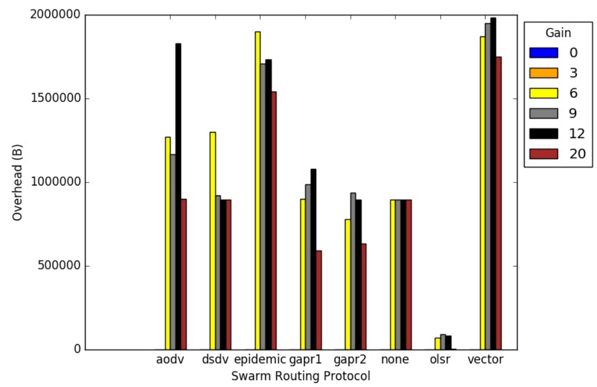
(a) 6 v 6 Swarm



(b) 10 v 10 Swarm



(c) 15 v 15 Swarm



(d) 25 v 25 Swarm

Figure 4: Total Overhead

REFERENCES

[1] A. Pospischil and J. P. Rohrer, "Multihop routing of telemetry data in drone swarms," in *Proceedings of the International Telemetering Conference (ITC)*, (Las Vegas, NV), pp. 1–10, October 2017.

[2] T. H. Chung, M. R. Clement, M. A. Day, K. D. Jones, D. Davis, and M. Jones, "Live-fly, large-scale field experimentation for large numbers of fixed-wing UAVs," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1255–1262, 2016.

[3] J. P. Rohrer, A. Jabbar, E. Perrins, and J. P. G. Sterbenz, "Cross-layer architectural framework for highly-mobile multihop airborne telemetry networks," in *Proceedings of the IEEE Military Communications Conference (MILCOM)*, (San Diego, CA, USA), pp. 1–9, November 2008.

[4] K. S. Pathapati, A. Nguyen, J. P. Rohrer, and J. P. Sterbenz, "Performance analysis of the AeroTP transport protocol for highly-dynamic airborne telemetry networks," in *Proceedings of the International Telemetering Conference (ITC)*, (Las Vegas, NV), October 2011.

[5] C. E. Perkins and P. Bhagwat, "Highly dynamic destination-sequenced distance-vector routing (dsdv) for mobile computers," in *The conference on Communications architectures, protocols and applications (SIGCOMM)*, (New York, NY, USA), pp. 234–244, ACM, 1994.

[6] C. Perkins, E. Belding-Royer, and S. Das, "Ad hoc On-Demand Distance Vector (AODV) Routing." RFC 3561 (Experimental), July 2003.

[7] T. Clausen and P. Jacquet, "Optimized Link State Routing Protocol (OLSR)." RFC 3626 (Experimental), Oct. 2003.

[8] A. Vahdat and D. Becker, "Epidemic routing for partially-connected ad hoc networks," Technical Report CS-200006, Duke University, April 2000.

[9] H. Kang and D. Kim, "Vector routing for delay tolerant networks," in *Proceedings of the IEEE 68th Vehicular Technology Conference*, pp. 1–5, Sept 2008.

[10] J. P. Rohrer and K. M. Killeen, "Geolocation assisted routing protocols for vehicular networks," in *Proceedings of the 5th IEEE International Conference on Connected Vehicles (ICCVE)*, (Seattle, WA), pp. 1–6, September 2016.

[11] "The ns-3 network simulator." `http://www.nsnam.org`, July 2009.

[12] J. P. Rohrer and A. N. Mauldin, "Implementation of epidemic routing with ip convergence layer in ns-3," in *Proceedings of the 2018 Workshop on ns-3 (WNS3)*, (Surathkal, India), ACM, June 2018.

[13] A. N. Mauldin, "Comparative analysis of disruption tolerant network routing simulations in the ONE and ns-3," Master's thesis, Naval Postgraduate School, Monterey, CA, December 2017.