Transmission Scheduling and Routing Algorithms for Delay Tolerant Networks

Rachel Dudukovich[1] and Daniel E. Raible[2]
NASA John H. Glenn Research Center, Cleveland, OH 44135

The challenges of data processing, transmission scheduling and routing within a space network present a multi-criteria optimization problem. Long delays, intermittent connectivity, asymmetric data rates and potentially high error rates make traditional networking approaches unsuitable. The delay tolerant networking architecture and protocols attempt to mitigate many of these issues, yet transmission scheduling is largely manually configured and routes are determined by a static contact routing graph. A high level of variability exists among the requirements and environmental characteristics of different missions, some of which may allow for the use of more opportunistic routing methods. In all cases, resource allocation and constraints must be balanced with the optimization of data throughput and quality of service. Much work has been done researching routing techniques for terrestrial-based challenged networks in an attempt to optimize contact opportunities and resource usage. This paper examines several popular methods to determine their potential applicability to space networks.

I. Introduction

The next decade will see the continued expansion of NASA's interplanetary telecommunication capabilities as the agency continues the push for more spacecraft positioned in the vicinity of Mars leading up to the first deep space human exploration mission. The current DSN (deep space network) is strained beyond capacity, so this expansion requires new technology development and deployment to support the science data return demands of upcoming missions. There exists a push for higher wavelength radio frequency (RF) systems as well as laser communication technologies to handle the increased throughput requirements. One of the key technologies necessary to evolve the architecture is a communications system functionality to manage the link delays, disconnections and disruptions due to such events as planetary obscuration, solar

1 Computer Engineer, Flight Software Branch (LSS), rachel.m.dudukovich@nasa.gov.

2 Aerospace Technologist in Telecommunications, Optics and Photonics Branch (LCP), daniel.e.raible@nasa.gov, AIAA Member.

conjunction, time of flight delays, node timing, ground terminal mission congestion and scheduling policy along with space and atmospheric weather disruptions. These deleterious effects all imply the need for network protocol solutions to ultimately manage the physical layer in a transparent manner to the end user. Delay Tolerant Networking (DTN) is an approach which addresses these challenges, and has been in a research and development phase for several years.

DTN is a store, carry, and forward network overlay that can operate over heterogeneous subnetworks. DTN provides autonomous link management, buffer management, and security for applications. DTN also includes quality of service (QoS) mechanisms to prioritize data and offers a standardized approach facilitating seamless integration and removal of nodes from a network. There has been a wealth of previous research on network management of heterogeneous RF and optical link architectures in the near-Earth environment [1] [2], but many of the techniques and parameter tunings are not extensible to the deep space domain due to the inherent dynamic differences between the environments and the lack of real time feedback to control from.

A multi-hop multi-path hybrid RF and optical test bed has been constructed to emulate a heterogeneous future deep space network and to support protocol and hardware refinement utilizing the Interplanetary Overlay Network (ION) implementation of DTN [3]. Initial experimental results characterized several of the aforementioned challenges and evaluated the effectiveness of DTN as a solution to mitigate them, revealing the need for significant amounts of local high speed memory to accommodate large and numerous bundles sent across high data rate physical layers. Further challenges associated with the Bundle Protocol Specification include the lack of reliability checks within the DTN bundle, no support for fragmentation, lack of definition for convergence layers, a flat address space makes scaling and routing difficult, and no standardized discovery mechanism [4].

Adoption of DTN into future high speed space networks, such as those realized by laser communications, hinges on the ability to successfully transmit data in the Gb/s order of magnitude range over the next few years. A successful test was performed at JPL with ION running within a Free-Space optical (FSO) network [5]. Forcing the CPU's to move data from non-volatile storage to RAM to the communications system interface at these rates would cause undue burden and bottlenecking. A potential solution being researched is the partial implementation of ION in FPGAs to affect a form of direct memory access. Offloading the non-computational overhead to hardware should significantly decrease ION's footprint without adding excessive complexity to the rest of the system; the data transfer could reside in the same FPGA as the encoder and modulator. To maintain flexibility and the ability to update the protocol, most of ION would remain in software form on the computer. Early experiments of this paradigm have examined the implications of custody transfer on the distribution of transfers and the inclusion of Contact Graph Routing (CGR) to allow establishment of one link to preclude all others – at least when they share a common outduct [6].

HiDRA

The High Data Rate Architecture (HiDRA) project has been developing a model of an extensible network communications interface providing multiple research payloads with

high speed optical and RF communication downlink capability. As a potentially multiple input – multiple output architecture, the complexity of resource management and job scheduling becomes critical. While ION does provide this functionality to a degree, it lacks the ability to opportunistically discover neighboring nodes, as well the capability to select a best path based on the current network state. As noted in [4], future missions may function both deterministic and non-deterministic networks, as such they may benefit greatly from a more adaptive routing paradigm.

An example of this can be seen in the Mars deep space network. Martian surface assets and the Martian orbiters can benefit from the use of opportunistic routing techniques due to their relatively close proximity [4]. Furthermore as the resources (bandwidth, storage, power, CPU utilization) of both types of assets are limited, care must be taken in the selection of routing techniques, as many can be resource hungry.

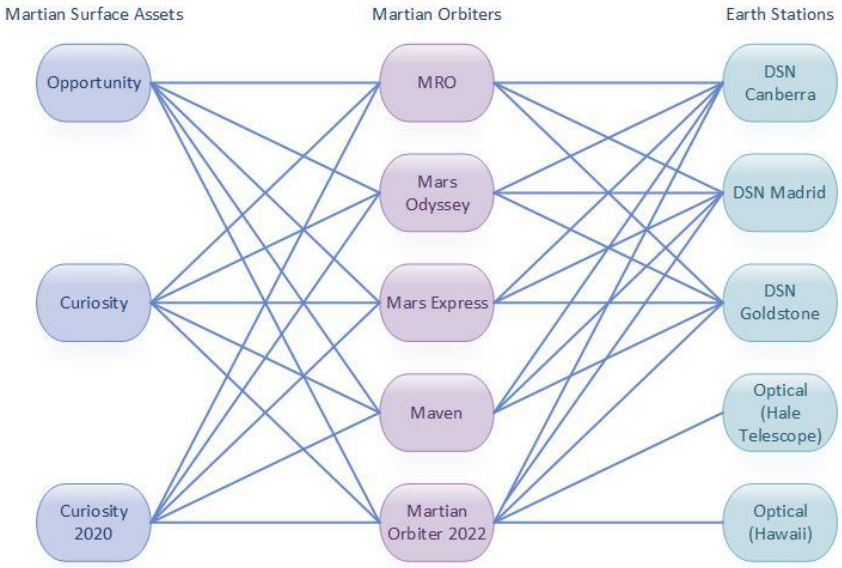

Figure 1. Notional schematic for the Mars Deep Space Network

Figure 1 shows a conceptual diagram for the Martian DSN. Surface assets such as the Opportunity and Curiosity rovers may use the Martian orbiters to relay data to the deep space ground stations. The orbiters have longer periods of contact with the Earth ground stations, as well as higher data rates in comparison to the rovers' capability to transmit directly to Earth. Figure 2 shows an example contact analysis for the Curiosity and Opportunity rovers to the Mars Reconnaissance Orbiter (MRO) and Mars Odyssey. In addition, it shows the contact times for MRO and Odyssey to the Canberra, Goldstone, and Madrid Deep Space Communications Complex (CDSCC, GDSCC, and MDSCC, respectively). This demonstrates the complexity of determining a best path to send data based on the assets that are currently available, the amount and priority of the data to be transmitted, the data rates of the associated communications links and the duration they will be available for. Furthermore, as this is a simple example for a small number of assets, it can be inferred that as the DSN matures, the number of potential paths will only increase.

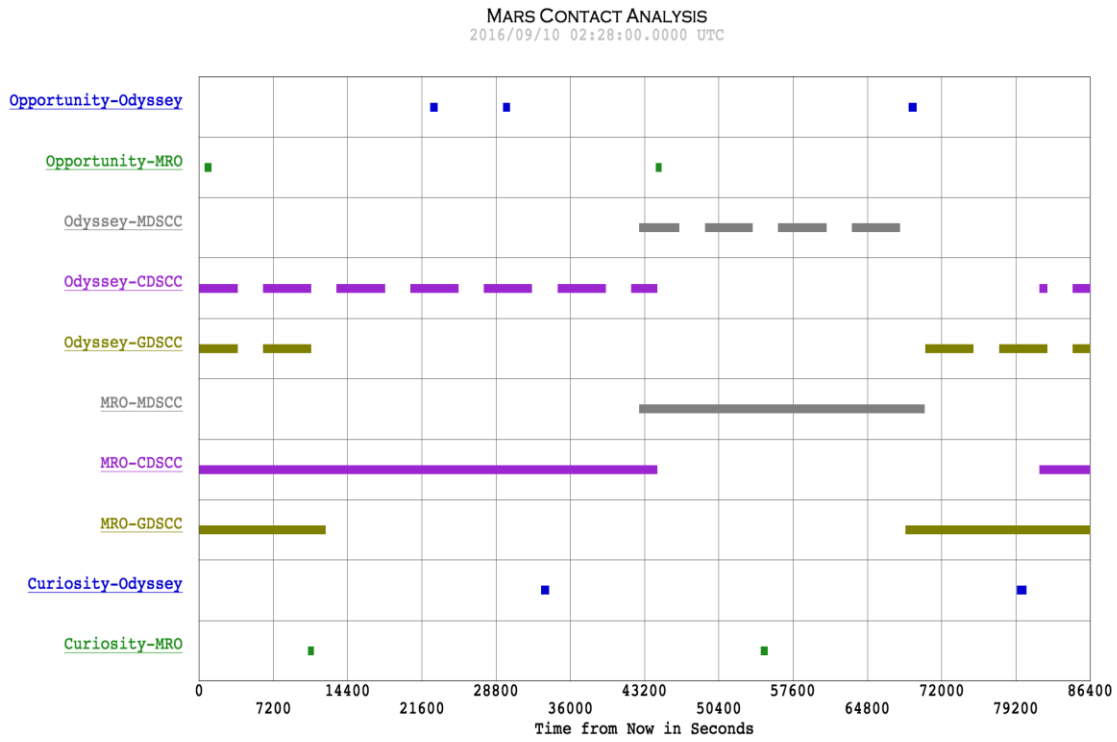

Figure 2. Example 24 Hour Contact Analysis for Martian DSN

II. Related Work

Classification of DTN Routing Algorithms

Balasubramanian et al. classify most DTN routing protocols as either based on packet forwarding or packet replication [7]. Replication based routing, or epidemic routing protocols, create multiple copies of a packet to send to neighboring nodes with the intent that the packet will traverse multiple paths and have a greater likelihood to reach its final destination. Forwarding based routing protocols create a single instance of a packet and employ various methods to determine a suitable path, often requiring global knowledge of the network.

In the case of a space network, it can be seen that both of these approaches have their own benefits and drawbacks. As noted in [7], naïve flooding can consume excessive resources on any node by generating multiple copies of unnecessary bundles. In the case of satellite networks, on-board avionics are often quite processor and memory limited, making this unnecessary processing particularly troublesome. Benefits of replication include redundancy to prevent lost packets, and potentially simplified algorithms which require limited knowledge of the global network. The need for feedback regarding the network state in particular can be impractical for deep space communication, where information will likely be stale by the time it reaches its destination. In contrast, while forwarding-based protocols require fewer resources, they often have lower message delivery rates [8]. Furthermore, the use of an oracle with future knowledge of the

network, or a knowledge base of the existing network may be difficult to implement in many real-life scenarios [7].

PRoPHET

The PRoPHET routing protocol attempts to reduce the number of replicated bundles in the network by calculating the probability of successful message delivery to a given destination. PRoPHET is based on the human mobility model and the observation that a large number of contact opportunities between two nodes follow a non-random pattern [9]. Messages are replicated and sent to neighboring nodes that have a high probability of delivering it to its destination. PRoPHET determines this likelihood based on a delivery predictability metric. Each node maintains a vector of delivery predictabilities for all nodes encountered and exchanges this information with other nodes during an initial contact phase. The delivery predictability is calculated whenever two nodes are in contact. Nodes which are frequently in contact have a higher delivery predictability and as such the algorithm will choose that pair of nodes as the preferred path. The delivery predictability *P(A,B)* for node *A* to destination *B* is calculated as follows [10]:

$$P(A,B) = P(A,B)_{old} + (1 - P(A,B)_{old}) \times P_{enc}. \quad (1)$$

The probability of direct encounter P_{enc} is a configurable parameter meant to increase the delivery probability of nodes that are frequently encountered. Delivery predictabilities for other nodes encountered by *B* are updated for node *A* using the transitive property. The transitive property is based on the concept that if node *A* frequently encounters *B* and node *B* frequently encounters node *C*, then node *A* can be used to forward messages to *C* via node *B* [10]. In Eq. 2, the value of β is a scaling factor for the transitivity of predictability and is a configurable parameter;

$$P(A,C) = P(A,C)_{old} + (1 - P(A,C)_{old}) \times P(A,B) \times P(B,C) \times \beta. \quad (2)$$

To reflect changes in the network, the delivery predictability for each node *i* decays over time according to Eq. 3:

$$P(A,i) = P(A,i)_{old} \times \gamma^{T}. \quad (3)$$

In Eq. 3, *T* represents the length of time since the probability was last aged and γ is constant. The PRoPHET Internet Draft recommends values of 0.75 for P_{enc}, 0.25 for β, and 0.99 for γ as a starting point, though they may be tuned for a particular application [9].

Delay Tolerant Link State Routing

Delay Tolerant Link State Routing (DTLSR) is based on conventional link state routing [11]. Nodes attempt to learn the network topology by sending flooding messages containing connectivity information for the current state of the network. The network topology is stored by each node in the form of a network graph. Routes are computed using Dijkstra's shortest path algorithm. Link State Announcement messages may contain

the source node's endpoint identifier, sequence number and link state information such as the next hop destination and queue status.

DTLSR differs from standard link-state routing (LSR) in that currently unavailable nodes are still considered in the best path computation. For nodes that are available, hop count can be used as a simple metric to determine the best path. This does not allow the algorithm to take advantage of better paths that may not currently be available but will be in the future when the message arrives at a remote node. To account for this, DTLSR attempts to minimize the estimated expected delay. For nodes that are available, the delay is estimated based on the total queue size *qlen*, number of messages in the link queue *qnum*, the per-message latency and bandwidth. The estimated delay is given by Eq. 4 [11]:

$$qnum \times latency + qlen \times bandwidth. \quad (4)$$

The estimated delay associated with unavailable nodes is inferred from the duration of the current outage. This is based on the assumption that if a node has been unavailable for a long amount of time, it is likely to continue to be unavailable. The duration is limited to 24 hours [11].

RAPID

The Resource Allocation Protocol for Intentional DTN (RAPID) was developed at the University of Massachusetts Amherst and was deployed as part of the DieselNet project. It attempts to conserve resources such as bandwidth, storage space, and power by only replicating bundles that optimize a specified routing metric [7]. The RAPID algorithm can be configured to optimize average delay, worst-case delay, or number of bundles delivered before they expire. This is done using a per-packet utility function specific to the desired routing metric. When two nodes encounter one another they exchange metadata about what bundles they have buffered, as well as information from past meetings. Bundles that can be directly delivered to their destination are transferred in order of creation time. Bundles that are destined for another node in the network are replicated if they do not already exist in the neighbor's buffer. The utility function is calculated for each bundle and they are then selected for transfer in decreasing order of their marginal utility.

The functionality of RAPID is broken into three main elements. A selection algorithm determines what packets to replicate based on their contribution to the optimization of the desired metric. An inference algorithm estimates the bundle's contribution to the selected routing metric. A control channel is used to exchange information about bundles in the network with other nodes [7].

Table 1 shows the routing metrics used by RAPID. Here U_i is the packet's utility, or the packet's expected contribution to a given routing metric, *D(i)* is the packet's expected delay, and *S* is the set of all packets in a given node's buffer [7].

Metric	Per-packet Utility Function	Explanation
Minimize Average Delay	$U_i = -D(i)$	Replicate packets which reduce the delay most
Minimize Expired Bundles	$U_i = \begin{cases} P\big(a(i) < L(i) - T(i)\big), L(i) > T(i) \\ 0 \qquad otherwise \end{cases}$	L(i) is the bundle time to live and T(i) is the time since creation. A bundle that has expired has a utility of 0.
Minimize Maximum Delay	$U_i = \begin{cases} -D(i), \quad D(i) \geq D(j) \quad \forall j \in S \\ 0 \quad otherwise \end{cases}$	Replicate the packet which is causing the maximum delay

Table 1. RAPID Routing Metrics

RAPID estimates the delay in a three-step process. Each node maintains a queue of bundles for each destination in decreasing order of the time they were created. The delivery delay distribution is computed for each bundle as if it is to be delivered directly, based on the number of bytes ahead of it in the queue and the size in bytes of the expected transfer opportunity. This is done for each node possessing a copy of the bundle. The minimum is then found among all delay distributions for each replicated bundle [7].

III. Experimental Setup

The NASA DTNBone test bed was used to evaluate the several current state-of-the-art routing algorithms for delay tolerant networks. The NASA DTNBone [4] consists of thirteen virtual machines running Ubuntu 14.04.5 LTS. Each virtual machine runs the current version of DTN2 (version 2.9.0). The nodes are networked together in a mesh topology and link delays and disruptions are simulated using channel-emulating software. Link disruptions are simulated hourly, with each link following its own schedule. The configuration of the network for initial testing is shown in Fig. 3. Table 2 shows the data rates and one-way delays associated with each link. TCP was used as the convergence layer for this initial testing.

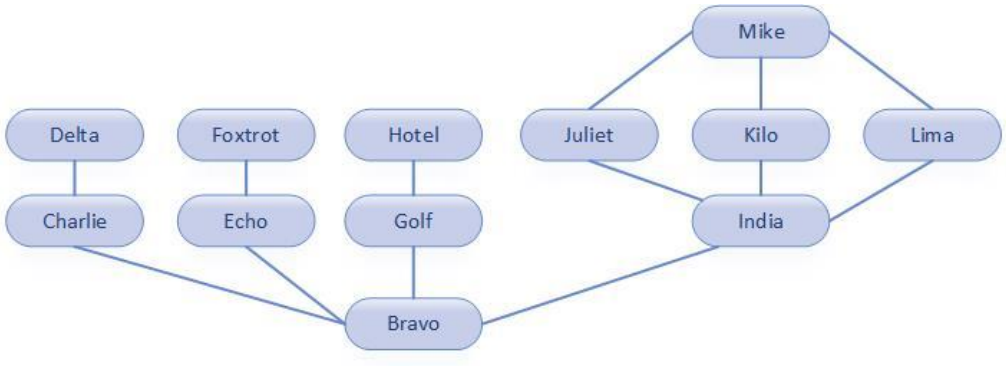

Figure 3. Network Topology of the NASA DTNBone

GRC DTNbone Configuration			
Link	**Delay**	**Rate Limit**	**Availability**
bravo-charlie	None	None	Toggles every 30 minutes
bravo-echo	None	None	Toggles every 3 minutes
bravo-golf	5 s	128 Kb/s	Always available
bravo-india	None	None	Always available
charlie-delta	None	None	Toggles every 3 minutes
echo-foxtrot	None	None	Toggles every 30 minutes
golf-hotel	200 ms	256 Kb/s	Always available
india-juliet	1250ms	512 Kb/s	Up for 20 minutes at beginning of the hour
india-kilo	1250 ms	512 Kb/s	Up for minutes starting at 20 minutes past the hour
india-lima	1250 ms	512 Kb/s	Up for minutes starting at 40 minutes past the hour
juliet-mike	200ms	1544 Kb/s	Toggles every 6 minutes
kilo-mike	200 ms	1544 Kb/s	Toggles every 5 minutes
lima-mike	200 ms	1544 Kb/s	Toggles every 2 minutes

Table 2. DTNbone Availability Schedule

DTN2 provides an ideal framework for DTN software research and development as it includes a bundle protocol implementation as well as DTLSR, flooding, and PRoPHET routing implementations. In addition, it provides an interface for external routers to communicate with and control the DTN2 daemon, allowing developers to easily integrate custom software with the existing bundle protocol implementation. This is accomplished by sending XML message to a port used by the DTN2 daemon. The RAPID protocol was implemented using this approach and as such can be used as an example for further software development.

For purposes of exercising each algorithm, the dtnperf tool included in DTN2 was used to send a series of bundles to a specified node in the network. The dtnperf tool allows the user to configure the bundle size, number of bundles and a destination server node in the network to send the bundles to. It generates a time stamped log of bundle forwarding and delivery status to allow the user to analyze network performance. Most testing was done sending bundles from node Bravo to node Mike since this is the most complex path for the algorithm to navigate as it has the most hops, possible paths and intermittent disruptions. The algorithms selected were the DTLSR, PRoPHET and flooding implementations provided by DTN2, as well as the RAPID implementation developed by University of Massachusetts Amherst as an external router to DTN2.

Of the three routing protocols internal to DTN2 that were tested, DTLSR performed the best, followed by flooding. The results from initial testing a summarized in Table 3.

Algorithm	Average Delay (s)	Average # Replications
DTLSR	78.14661576	2.24
Flooding	99.1532053	5.22
RAPID	511.6900912	5.384285714

Table 3. Results for 50 1 KB Bundles

To study the effects of bundle size on the routing algorithms, a simpler destination to reach in the network was chosen. The path from node Bravo to node Hotel consists of only two hops and has links which are always available. DTLSR continued to perform better than RAPID as shown in Fig. 4. In addition to having a lower average delay, it also did not replicate unnecessary packets. In the case of RAPID, there were still typically an average of 4 bundles replicated per delivery, even though there was a direct path to the destination.

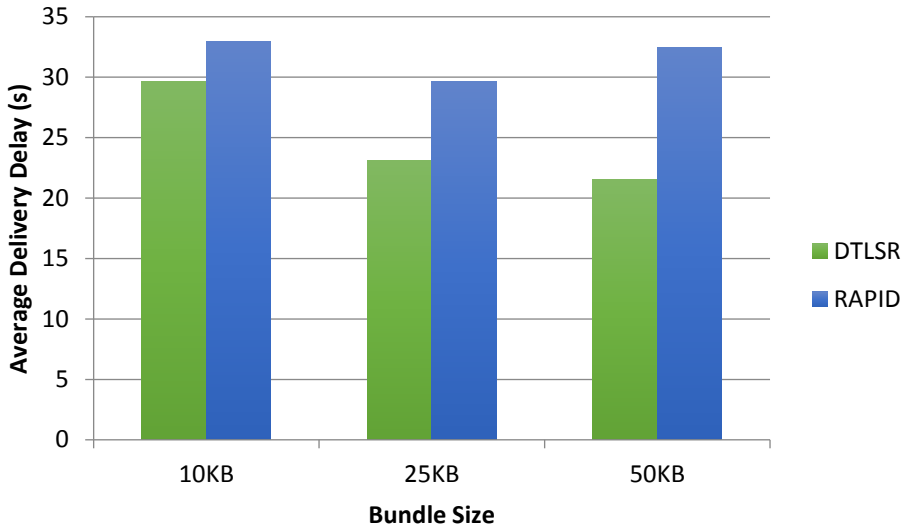

Figure 4. Average Delay for 2 Hop Path

The PRoPHET routing algorithm is not included in the preliminary results as its performance was quite unstable and it was difficult to send any number of bundles to even directly connected nodes. Forwarding bundles to a destination requiring multiple hops was even less successful. The initial parameters used were configured as recommended in the PRoPHET Internet-Draft and also several attempts at adjusting them to improve performance were made. This is not to say that further testing could not be done to determine the cause of the poor performance, whether it be due to configuration parameters needing to be tuned for each node or some other factor in the PRoPHET implementation or DTN2.

There are a number of studies of characterizing DTN routing algorithm performance that have shown PRoPHET's performance to be inferior to MaxProp, Spray and Wait, Epidemic, and RAPID [7] [12] [10] in some cases, particularly depending on the mobility scenario used, as well as the amount of time given to allow the algorithm to converge. For the case of this initial testing, two problems were noted that impacted performance. The first was that links with a delay associated with them (bravo-golf: 5 seconds, india-juliet, india-kilo, india-lima: 1250ms) seemed completely missed in the exchange of PRoPHET routing bundles containing the delivery predictability information. This is no doubt due to the expiration of some timeout value, however increasing the value of the hello_interval (20 s) and hello_dead value (allow up to 10 hello_intervals before a node is

considered unreachable) did not help to solve the problem. In addition, the algorithm seemed to have a difficult time reacting to availability changes in links that toggle frequently (bravo-echo, toggle link availability every 3 minutes). When the link was available, the delivery predictability would approach 1, however the link would become unavailable and this would not be reflected in the delivery predictability, causing the algorithm to continue to repeatedly attempt to contact the unavailable node. These types of problems are noted in [10], where the authors discuss improvements for a second revision of the PRoPHET protocol. They note that when the frequency of encounters is disproportionate throughout the network and encounters occur frequently enough that the delivery predictability is not reduced quickly enough by the aging procedure, the algorithm can fail to produce an accurate representation of the current network state. It is possible that performance could be improved for this test case by further investigating the recommendations of PRoPHETv2, as well as further refining the PRoPHET configuration parameters for each node. In the initial test case, all nodes were configured with the same parameters, but it would likely be beneficial to customize the parameters based on the link characteristics of each node, essentially to make nodes with links that change frequently adjusts their delivery predictabilities more aggressively.

IV. Conclusions and Future Work

There is a large body of research regarding opportunistic and adaptive routing algorithms of delay tolerant and challenged networks. Many of the principles used in such work can be applied to future missions to take advantage of contact opportunities between multiple surface and space assets. As the deep space and space networks expand, the increasing complexity of network management will require more sophisticated techniques.

Further testing can be done to better tune the parameters for PRoPHET to specific network scenarios. In addition, the impact of bundle size and the amount of time allowed for the algorithm to learn the network behavior can also be investigated more fully. Other techniques such as History Based Scheduling and Drop (HBSD) and Distributed Composite Multiple Criteria Routing can also be explored.

Acknowledgments

The authors would like to thank the NASA Space Communications and Navigation (SCaN) program, and in particular Dr. Don Cornwell for supporting this research. In addition, we would like to thank Dr. Christos Papachristou of Case Western Reserve University and Alan Hylton and Dennis Iannicca of NASA Glenn Research Center for their help and support.

References

[1] Chan, V., “Optical Satellite Networks”, *Journal of Lightwave Technology*, 21(11), November 2003.

[2] Clark, P. And Sengers A., Wireless Optical Networking Challenges and Solutions. Military Communications Conference IEEE, 2004.

[3] Raible, D. And Hylton, A., “Integrated RF/Optical Interplanetary Networking Preliminary Explorations and Empirical Results”. Ottawa, Canada, 2012. 30th AIAA International Communications Satellite Systems Conference.

[4] Hylton, A. And Raible, D., “Networked Operations of Hybrid Radio Optical Communications Satellites”, San Diego, California, 2014. 32nd AIAA International Communications Satellite Systems Conference.

[5] Schoolcraft, J. and Wilson, K., “Experimental Characterization of Space Optical Communications with Disruption-Tolerant Network Protocols”, Space Optical Systems and Applications (ICSOS), 2011.

[6] Juergens, J., Hylton, A., Raible, D. And Iannica, D., "On Applications of Disruption Tolerant Networking to Optical Networking in Space", Ottawa, Canada, 2012. 30th AIAA International Communications Satellite Systems Conference.

[7] Balasubramanian, A., Levine, B. N. and Venkataramani, A., "Replication Routing in DTNs: A Resource Allocation Approach", *IEEE/ACM Transactions on Networking*, 18(2):596–609, April 2010.

[8] Spyropoulos, T., Vasilakos, A. , Zhang, Y., editor. *Delay Tolerant Networks :Protocols and Applications*. CRC Press, 2012.

[9] Lindgren, A. And Doria, A., "Probabilistic Routing Protocol for Intermittently Connected Networks", 2006, https://tools.ietf.org/html/draft-irtf-dtnrg-prophet-09

[10] Grasic, S., Davies, E., Lindgren, A., and Doria, A., "The Evolution of a DTN Routing Protocol - PRoPHETv2", In *Proceedings of the 6th ACM Workshop on Challenged Networks*, CHANTS '11, pages 27–30, New York, NY, USA, 2011. ACM.

[11] Demmer, M. and Fall, K., "DTLSR: Delay Tolerant Routing for Developing Regions", In *Proceedings of the 2007 Workshop on Networked Systems for Developing Regions*, NSDR '07, pages 5:1–5:6, New York, NY, USA, 2007. ACM.

[12] Almeida, V., Oliveira, A., Macedo, D. F. and Nogueira J. M. S., "Performance Evaluation of MANET and DTN Routing Protocols", In *Proc. IFIP Wireless Days (WD)*, pages 1–6, November 2012.

[13] Malakooti, B., Thomas, I., "A Distributed Composite Multiple Criteria Routing Using Distance Vector", Proceedings of the 2006 IEEE International Conference on Networking, Sensing, and Control, 2006.