

# Dynamic Routing for Delay-Tolerant Networking in Space Flight Operations

Scott C. Burleigh<sup>1</sup>

*Jet Propulsion Laboratory, California Institute of Technology, Pasadena, California, 91109*

**Contact Graph Routing (CGR) is a dynamic routing system that computes routes through a time-varying topology composed of scheduled, bounded communication contacts in a network built on the Delay-Tolerant Networking (DTN) architecture. It is designed to support operations in a space network based on DTN, but it also could be used in terrestrial applications where operation according to a predefined schedule is preferable to opportunistic communication, as in a low-power sensor network. This paper will describe the operation of the CGR system and explain how it can enable data delivery over scheduled transmission opportunities, fully utilizing the available transmission capacity, without knowing the current state of any bundle protocol node (other than the local node itself) and without exhausting processing resources at any bundle router.**

## I. Introduction

ONE of the most challenging research problems in Delay-Tolerant Networking<sup>1</sup> (DTN) has always been the automatic, dynamic computation of efficient forwarding routes between pairs of network nodes separated by one or more links that are characterized by frequent and lengthy outages. This is, in part, because the character of the links prevents the timely exchange of information – routing protocol messages – that enables effective route planning in more benign environments.

Route computation in a delay-tolerant interplanetary network is particularly difficult: not only are the links often interrupted, but when active they also are subject to lengthy signal propagation delay.

Contact Graph Routing (CGR) is a dynamic routing system that computes routes through a time-varying topology composed of scheduled, bounded communication contacts in a DTN network. It is designed to support operations in a space network based on DTN, but it also could be used in terrestrial applications where operation according to a predefined schedule is preferable to opportunistic communication, as in a low-power sensor network.

The basic strategy of CGR is to take advantage of the fact that, since communication opportunities throughout the network are planned in detail, the communication routes between any pair of bundle agents in a population of nodes that have all been informed of one another's plans can be inferred from those plans rather than discovered via dialogue, which is impractical over space links with long one-way light times.

This paper will describe the operation of the CGR system and explain how it can enable data delivery over scheduled transmission opportunities, fully utilizing the available transmission capacity, without knowing the current state of any bundle protocol node (other than the local node itself) and without exhausting processing resources at any bundle router.

## II. Background

Delay-tolerant networking (DTN) is an architecture for automated data networks in which end-to-end communication is reliable and efficient despite possibly sustained and/or frequent interruptions in connectivity and potentially high signal propagation latency. As such it promises to provide an inexpensive and robust medium for returning telemetry relayed from research vehicles in environments that provide meager support for communications: deep space, the surface of Mars, the poles or the sub-Arctic steppes of Earth, and others.

The design of algorithms for computing end-to-end forwarding routes in DTN-based networks, and of protocols for conveying the information required to invoke those algorithms, is a problem that has been the subject of much research beginning in early 2004<sup>2-10</sup>. Ref. 11 offers a thorough survey of the field. More recently, formal theory that contemplates the general solvability of DTN routing problems has appeared<sup>12</sup>.

---

<sup>1</sup> Principal Engineer, Systems Engineering Section, 4800 Oak Grove Drive m/s 301-490, Pasadena CA 91109

To date, the DTN routing algorithms that have been most intensively studied have been those that address routing in networks that may be frequently disrupted but are not subject to lengthy signal propagation delay. That is, they are designed primarily to address challenged networks in terrestrial environments where the distances between nodes are relatively small, i.e., much less than one light second (186,000 miles): bidirectional connectivity may be intermittent in such environments but when established it is conversational, enabling neighboring DTN nodes ("bundle agents") to exchange routing information in timely fashion before and while exchanging application data packaged in "bundles".

In an interplanetary DTN this condition does not hold: distances are so great that the length of a communication contact opportunity between two nodes may be smaller than the time required for a single communication round trip. A DTN node may therefore be required to initiate, perform, and terminate a transmission episode before receiving any responsive information whatsoever from its neighbor. Communication algorithms must function using whatever pre-placed information is available at the beginning of each opportunity. Since it may be physically impossible for nodes to have authoritative current knowledge about their peers, they can only act on the basis of expectation.

Fortunately, the expectations that would be provided to nodes in an interplanetary DTN can be accurate enough to minimize error: interplanetary flight missions have always, of necessity, been operated on this model. Communication opportunities, like all other flight mission activities, are planned, scheduled, and communicated to remote vehicles far in advance of the subject events. These plans may be revised, but the revisions too must be communicated well in advance in order for the cooperating mission elements to coordinate such variables as spacecraft attitude, antenna articulation, and transponder state changes. Without that coordination, no communication is possible at all.

So unlike other DTN routing approaches, CGR proceeds from the assumption that, while it's likely that no node in the network will ever have current knowledge of the routing state of any other, all nodes always have substantially consistent comprehensive knowledge of all scheduled future communication contacts within the network. This information, together with knowledge of the current state of the local node, is sufficient to enable generally successful route computation at any node. Anomalies will of course occur, but mechanisms are provided for mitigating any resulting degradation in routing performance.

### III. Route Computation in CGR

#### A. Concepts and Terminology

CGR concepts in some cases do not correspond to widely used route computation terms of art. The following definitions are offered to help clarify this discussion.

##### 1. Node ID

The DTN *node* concept is well-developed in the specification for the Bundle Protocol<sup>13</sup> (BP), but BP communication endpoints typically do not map one-for-one with nodes: any number of BP endpoints may reside on a single node, and in fact a single BP endpoint may reside on multiple nodes. At the limit, this abstract topology has the potential to become an arbitrarily complex routing fabric. For the purposes of routing in an interplanetary DTN, we believe a simpler model suffices: routing is based not on endpoints but rather on nodes, which, unlike endpoints, are bound to specific links. For this purpose, all BP endpoints in the network are identified by names from which the identities of the nodes on which they reside may be readily inferred. The details of this convention are beyond the scope of the present paper; the key points are that (a) we can route each bundle to its destination endpoint by simply routing it to the node on which the endpoint resides and (b) each node is uniquely identified by an integer greater than zero, termed its *node ID* or node number.

##### 2. Contacts

A *contact* is here defined as an interval during which it is expected that data will be transmitted by DTN node A (the contact's *transmitting node*) and most or all of the transmitted data will be received by node B (the contact's *receiving node*). Implicitly, the transmitting mode will utilize some "convergence-layer" protocol underneath the DTN overlay Bundle Protocol to effect this direct transmission of data to the receiving node. Each contact is characterized by its start time, its end time, the identities of the transmitting and receiving nodes, and the rate at which data are expected to be transmitted by the transmitting node throughout the indicated time period.

##### 3. Range intervals

A range interval is a period of time during which the displacement between two nodes A and B is expected to vary by less than 1 light second from a stated anticipated distance. (We expect this information to be readily computable from the known orbital elements of all nodes.) Each range interval is characterized by its start time, its

end time, the identities of the two nodes to which it pertains, and the anticipated approximate distance between those nodes throughout the indicated time period.

#### 4. *Topology timeline*

The topology timeline at each node in the network is a time-ordered list of scheduled or anticipated changes in the topology of the network. Entries in this list are of two types:

- *Contact entries* characterize scheduled contacts as described above.
- *Range entries* characterize anticipated range intervals as described above.

#### 5. *Contact Graphs*

The *contact graph* constructed locally by each node in the network notionally contains, for every other node D in the network:

- A list of *xmit* structures characterizing data reception opportunities at D, encapsulating contact start time, stop time, transmitting node number, and data transmission rate, ordered by start time. This information is derived from all Contact timeline entries whose receiving node is node D.
- A list of *origin* structures, each encapsulating the presumed **current** distance of some potential transmitting node S from node D. This information is derived from Range timeline entries.

In concept, this information is periodically updated by a background "clock" task that applies stored Range timeline entries to the contact graph as their start times are reached, purging Contact entries and corresponding xmit structures – and Range entries – whose stop times have passed.

#### 6. *Neighbors*

Each node to which, according to the contact graph, the local node will at some time transmit data directly via some convergence-layer protocol is termed a *neighbor* of the local node. Each neighbor must be associated with a single notional outbound bundle transmission queue for the applicable convergence-layer (CL) protocol adapter – termed an *outduct* – so that bundles that are to be transmitted directly to this neighbor can simply be queued for transmission via that CL protocol.

#### 7. *Static routes*

CGR can be configured to forward all bundles that are destined for a given node by enqueueing them to some explicitly specified outduct – a *static route*. Static routing overrides any dynamic route that would be discovered from an examination of the contact graph.

#### 8. *Well-formed routes*

A *well-formed route* for given bundle is defined as sequence of contacts such that the first contact is from the bundle's source to some other node, every subsequent contact in the sequence is from the receiving node of the prior contact to some other node, the last contact in the sequence is from some node to the bundle's final destination, and the route contains no loops – i.e., no two contacts in the sequence involve transmission from the same node and no two contacts in the sequence involve transmission to the same node.

#### 9. *Expiration time*

Every bundle transmitted via DTN has a time-to-live (TTL), the length of time after which the bundle is subject to destruction if it has not yet been delivered to its destination. The *expiration time* of a bundle is computed as its creation time plus its TTL. When computing the next-hop destination for a bundle that the local bundle agent is required to forward, there is no point in selecting a route that can't get the bundle to its final destination prior to the bundle's expiration time.

#### 10. *OWLT margin*

One-way light time (OWLT) – that is, distance – is obviously a factor in delivering a bundle to a node prior to a given time. OWLT can actually change during the time a bundle is en route, but route computation becomes intractably complex if we can't assume an OWLT "safety margin" – a maximum delta by which OWLT between any pair of nodes can change during the time a bundle is in transit between them. We assume that the maximum rate of change in distance between any two nodes in the network is about 150,000 miles per hour, which is about 40 miles per second. (This was the speed of the Helios spacecraft, the fastest man-made object launched to date.) At this speed, the distance between any two nodes that are initially separated by a distance of N light seconds will increase by a maximum of 40 miles per second of transit. This will result in data arrival no later than roughly N + Q seconds after transmission – where the *OWLT margin* value Q is (40 \* N) divided by 186,000 – rather than just N seconds after transmission as would be the case if the two nodes were stationary relative to each other. When computing the expected time of arrival of a transmitted bundle we simply use N + Q, the most pessimistic case, as the anticipated total in-transit time.

#### 11. *Last moment*

The *last moment* for sending a bundle during a given contact such that it will arrive at the receiving node prior to some deadline is computed as the deadline minus the sum of (a) the current one-way light time N between the

contact's transmitting and receiving nodes (which can be obtained from the origin structure for the transmitting node, in the receiving node's list of origins) and (b) the applicable OWLT margin for N, as above. If the contact's start time is after the last moment for the deadline, then clearly no transmission whatsoever that is initiated during that contact can be assured of getting the bundle to the contact's receiving node prior to the deadline.

#### 12. Capacity

The *capacity* of a contact is the product of its data transmission rate (in bytes per second) and its duration (stop time minus start time, in seconds).

#### 13. Estimated capacity consumption

The size of a bundle is simply the sum of its payload size and its header size, but bundle size is not the only lien on the capacity of a contact. The total *estimated capacity consumption* (or "ECC") for a bundle that is queued for transmission via some outduct is a more lengthy computation:

- For each recognized convergence-layer protocol, we can estimate the number of bytes of "overhead" (that is, data that serves the purposes of the protocol itself rather than the user application that is using it) for each frame of convergence-layer protocol transmission. If the convergence layer protocol were UDP/IP over the Internet, for example, we might estimate the convergence layer overhead per frame to be 100 bytes – allowing for the nominal sizes of the UDP, IP, and Ethernet or SONET overhead for each IP packet.
- We can estimate the number of bundle bytes per CL protocol frame as the total size of each frame less the per-frame convergence layer overhead. Continuing the example begun above, we might estimate the number of bundle bytes per frame to be 1400, which is the standard MTU size on the Internet (1500 bytes) less the estimated convergence layer overhead per frame
- We can then estimate the total number of frames required for transmission of a bundle of a given size: this number is the bundle size divided by the estimated number of bundle bytes per CL protocol frame, rounded up.
- The estimated *total convergence layer overhead* for a given bundle is, then, the per-frame convergence layer overhead multiplied by the total number of frames required for transmission of a bundle of that size
- Finally the ECC for that bundle can be computed as the sum of the bundle's size and its estimated total convergence layer overhead.

#### 14. Residual capacity

The residual capacity of a given contact between the local node and one of its neighbors is the sum of the capacities of that contact and all prior scheduled contacts between the local node and that neighbor, less the sum of the ECCs of all bundles currently queued on the outduct for transmission to that neighbor.

#### 15. Plausible opportunity

A *plausible opportunity* for transmitting a given bundle to some neighboring node is defined as a contact whose residual capacity is at least equal to the bundle's ECC. That is, if the capacity of a given contact is already fully subscribed, when computing routes for an additional bundle there is no purpose served by assuming transmission during that contact.

#### 16. Plausible routes

A *plausible route* for a given bundle is a well-formed route whose constituent contacts are all plausible transmission opportunities such that transmission of the bundle during each contact can occur before the last moment for that contact's applicable deadline. The applicable deadline for the last contact in the route is the bundle's expiration time; the applicable deadline for each preceding contact is the end of that contact.

#### 17. Critical bundles

A *Critical bundle* is one that absolutely has got to reach its destination and, moreover, has got to reach that destination as soon as is physically possible. For ordinary non-Critical bundles, the CGR dynamic route computation algorithm uses the contact graph to calculate which of the plausible routes to the bundle's final destination should theoretically get the bundle to that destination in the shortest total elapsed time, assuming no unforeseen delays. It then inserts the bundle into the outbound transmission queue for transmission to the neighboring node that is the first step along that route. It is possible, though, that due to some unforeseen delay the selected route will turn out to be less successful than another route that was not selected: the bundle might arrive later than it would have if another route had been selected, or it might not even arrive at all. For Critical bundles, the CGR dynamic route computation algorithm causes the bundle to be inserted into the outbound transmission queues for transmission to *all* neighboring nodes that are on plausible routes to the bundle's final destination. The bundle is therefore guaranteed to travel over the most successful route, as well as over all other plausible routes. (Note that this may result in multiple copies of a Critical bundle arriving at the final destination. Note too that the procedures for transmission of critical bundles do

not constitute "flood routing": the bundle is forwarded on every plausible route to the final destination but probably not to every node in the network, since many nodes will not be on any plausible route.)

#### 18. Groups

When the size of the network makes it impractical to distribute all Contact and Range information for all nodes to every node, the job of computing dynamic routes to all nodes may be partitioned among multiple gateway nodes. Each gateway is responsible for managing a comprehensive contact graph for some subset of the total network population – a *group*, comprising all nodes whose node numbers fall within the range of node numbers assigned to the gateway. A bundle destined for a node for which no static route is declared and for which no dynamic route can be computed from the local node's contact graph may be routed to the gateway node for the group within whose range the destination's node number falls. Static routes must be declared for all gateway nodes that are not neighbors of the local node. (Note that the group mechanism implements default routing in CGR in addition to improving scalability.)

#### 19. Excluded neighbors

A *custodian* for a bundle is a forwarding node that is required to retain the bundle, possibly re-forwarding it some number of times, until some downstream receiving node "accepts custody" of the bundle by sending a "custody signal" bundle to the custodian affirming that the receiving node has agreed to become the bundle's new custodian. Not every bundle has a custodian: custody transfer procedures are effected only for bundles that are flagged by their source nodes as being "custodial bundles". Custody signals can convey either acceptance or refusal of custody for a given bundle; refusal of custody can be interpreted as an indication that the current custodian should re-forward the bundle along a route that excludes the refusing node. A neighboring node C that refuses custody of a bundle destined for some remote node D is here termed an *excluded neighbor* for (that is, with respect to computing routes to) node D. So long as C remains an excluded neighbor for D, no bundle destined for D will be forwarded to C – except that occasionally (once per lapse of the RTT between the local node and C) a custodial bundle destined for D will be forwarded to C as a "probe". C ceases to be an excluded neighbor for D as soon as it accepts custody of some bundle destined for D.

## B. Route Computation Algorithm

The CGR route computation algorithm encompasses three procedures, presented here in decreasing order of precedence. In every case, the result of successful routing is the insertion of the bundle into the outbound transmission queues for one or more neighboring nodes. (Note that the bundles in an outbound transmission queue should be ordered by decreasing priority, as indicated by the bundle Quality Of Service flags, but procedures for accomplishing this ordering are beyond the scope of this paper.)

#### 1. Static route override

We begin by simply looking for a static route to the bundle's final destination node. If such an overriding static route is found, we insert the bundle into the indicated outbound transmission queue and routing is complete.

#### 2. Dynamic route computation

Otherwise (no static route found), we attempt to compute a dynamic route to the bundle's final destination node.

We start this procedure by setting destination variable *D* to the bundle's final destination node number, setting "deadline" variable *X* to the bundle's expiration time, creating an empty list of *Proximate Nodes* to send to, and creating a list of *Excluded Nodes*, i.e., nodes through which we will *not* compute a route for this bundle. The list of Excluded Nodes is initially populated with:

- the node from which the bundle was directly received, so that we avoid cycling the bundle between that node and the local node (unless the route computation algorithm is being re-applied due to custody refusal as discussed later);
- all excluded neighbors for the bundle's final destination node.

Then we invoke the Contact Review Procedure as described below.

### Contact Review Procedure

First append node *D* to the list of Excluded Nodes, to prevent routing loops. (We don't want to re-compute routes through *D* in the course of computing routes for the intermediate nodes on any path to *D*.)

Then, for each xmit *m* in node *D*'s list of xmits:

If *m*'s start time is after the last moment *T* for deadline *X*, then skip this xmit.

Otherwise:

If *m*'s transmitting node *S* is the local node (that is, *D* is a neighbor of the local node):

- Compute the ECC of the bundle, assuming transmission via the local node's outduct to *D*.

- If  $m$ 's residual capacity is less than the computed ECC, then skip this xmit.
- Otherwise, if  $D$  is already in the list of Proximate Nodes to send this bundle to, then skip this xmit.
- Otherwise, add  $D$  to the list of Proximate Nodes.

Otherwise (that is,  $D$  is not a neighbor of the local node):

If  $S$  is already in the list of Excluded Nodes, then skip this xmit.

Otherwise:

- Compute *estimated forwarding latency*  $L$  as twice the size of the bundle, divided by the data transmission rate for xmit  $m$ . (This value is used to allow for the time needed by node  $S$  simply to receive the bundle from its origin, queue it for transmission, and re-radiate it.)
- Invoke the Contact Review Procedure again, recursively, but with destination variable  $D$  now set to  $S$  and with deadline variable  $X$  set either to  $T$  or to the time that is  $L$  seconds before the stop time of xmit  $m$ , whichever is earlier.

Finally, remove  $D$  from the list of Excluded Nodes (unwinding the recursion stack).

At this point, each member of the Proximate Nodes list is a neighboring node to which we can forward the bundle in the expectation that at least one of that node's planned contacts will enable conveyance of the bundle on a plausible route toward its final destination.

If the list of Proximate Nodes is non-empty:

- If the bundle is Critical, then we now insert the bundle into the outbound transmission queue for every one of those Proximate Nodes.
- Otherwise (the bundle is non-critical, so we must select only a single Proximate Node for transmission):
  - We insert the bundle into the outbound transmission queue of the Proximate node that is on the least constricted plausible route to the final destination. The least constricted plausible route to the final destination is the one with the largest "bottleneck", where the bottleneck on a route is defined as the contact whose capacity is smallest. (Note that this selection criterion biases the algorithm toward minimizing underutilization of large-capacity links, rather than toward minimizing delivery latency. The rationale here is that latency can be minimized by declaring the bundle to be Critical, and absent that declaration the waste of expensive communication opportunities is a greater evil than a few minutes or hours of delay in delivery to the final destination.)
  - We also note the stop time of the earliest plausible opportunity for transmission to the selected Proximate Node, retaining it as the *expected transmit time* for this bundle.
- Routing is now complete.

### 3. *Default route*

Otherwise (no dynamic route computed):

- If the bundle's destination node number is in the range of node numbers assigned to the gateway for some group, then we insert the bundle into the outbound transmission queue (neighbor outduct or static route) for that gateway node and routing is complete.
- Otherwise (no default route found), the bundle cannot be forwarded. If custody transfer is requested for this bundle, we send a custody refusal signal to the bundle's current custodian; in any case, we discard the bundle.

## C. Exception Handling

Conveyance of a bundle from source to destination through a DTN can fail in a number of ways, most of which are beyond the scope of this paper. Failures in Contact Graph Routing, specifically, occur when the expectations on which routing decisions are based prove to be false. These failures of information fall into two general categories: contact failure and custody refusal.

### 1. *Contact failure*

A scheduled contact between some node and its neighbor along the end-to-end route may be initiated later than the originally scheduled start time, or be terminated earlier than the originally scheduled stop time, or be canceled

altogether. Alternatively, the available capacity for a contact might be overestimated due to, for example, diminished link quality resulting in unexpectedly heavy retransmission at the convergence layer. In each of these cases, the anticipated transmission of a given bundle during the affected contact may not occur as planned: the bundle might expire before the contact's start time, or the contact's stop time might be reached before the bundle has been transmitted.

For a non-Critical bundle, we handle this sort of failure by means of a timeout: if the bundle is not transmitted prior to the expected transmit time noted at the moment the bundle was inserted into the outbound transmission queue, then the bundle is removed from its outbound transmission queue and the Route Computation Algorithm is re-applied to the bundle so that an alternate route can be computed.

#### 2. *Custody refusal*

A node that receives a bundle may find it impossible to forward it, for any of several reasons: it may not have enough storage capacity to hold the bundle, it may be unable to compute a forward route (static, dynamic, or default) for the bundle, etc. Such bundles are simply discarded, but discarding any such bundle that is marked for custody transfer will cause a custody refusal signal to be returned to the bundle's current custodian.

When the affected bundle is non-Critical, the node that receives the custody refusal re-applies the Route Computation Algorithm to the bundle so that an alternate route can be computed – except that in this event the node from which the bundle was originally directly received is omitted from the initial list of Excluded Nodes. This enables a bundle that has reached a dead end in the routing tree to be sent back to a point at which an altogether different branch may be selected.

For a Critical bundle no mitigation of either sort of failure is required or indeed possible: the bundle has already been queued for transmission on all plausible routes, so no mechanism that entails re-application of the Route Computation Algorithm could improve its prospects for successful delivery to the final destination.

However, in some environments it may be advisable to re-apply the Route Computation Algorithm to all Critical bundles that are still in local custody whenever a new Contact is added to the contact graph: the new contact may open an additional forwarding opportunity for one or more of those bundles.

### IV. Routing Protocol Messages

In CGR testing to date, the topology timeline at each node has simply been read from a file of timeline entries identifying all contacts and range changes throughout the test interval. For sustained operation in a real network this clearly will not suffice: contacts are planned in advance, but not years in advance. A means of distributing newly developed timeline entries to network nodes that are already operating must be provided. This information distribution mechanism, named "Contact Graph Routing Protocol" (CGRP), has not yet been specified in detail. In broad outline, we expect it to function as follows.

CGRP is an application-layer message protocol that resides immediately above Bundle Protocol (BP) in the DTN stack. That is, CGRP is itself a BP application; each CGRP message is part or all of the application data unit conveyed by a single bundle.

Each CGRP message has the following contents:

- An 8-bit message type, indicating whether the message is a Contact (type = 1) or Range (type = 2) message.
- The starting time of the interval to which the message pertains: the UTC count of seconds since the beginning of the year 2000 ("DTN time") expressed as a Self-Defining Numeric Value (SDNV). Both the SDNV concept and the notion of DTN time are discussed in detail in the Bundle Protocol specification.
- The stop time of this interval, again a DTN time expressed as an SDNV.
- Node number A (which is the Transmitting node for a Contact message), an SDNV.
- Node number B (for a Contact message, the Receiving node), an SDNV.
- Message value, an SDNV. For a Contact message, this value is the planned rate of transmission from node A to node B over this interval, in bytes per second. For a Range message, this value is the anticipated constant distance between A and B throughout this interval, in light seconds.

All CGRP messages are propagated to every node in the interplanetary DTN: rover, orbiter, ground station, etc.

Where communication between a pair of nodes is duplex we propagate two Contact messages, one for A → B transmission and a second for B → A transmission, because the data rates may be different in different directions.

We acknowledge that an alternative to using CGRP to pass all of this information would be local computation of mutual visibility windows and one-way light times based on local knowledge of orbital elements. This would reduce network traffic somewhat. However:

- It would do so at the cost of increased complexity at each node.
- Periods of communication are not always coterminous with periods of mutual visibility: scheduled science operations or other activity may preclude transmission or reception even when the orbital geometry is favorable.
- Scheduled data rates during windows of mutual visibility might not be known *a priori*, i.e., might still have to be propagated somehow.

So it seems likely that some mechanism along the lines of CGRP will be needed in support of interplanetary network operations in any case.

## V. Remarks

The CGR routing procedures respond dynamically to the changes in network topology that the nodes are able to know about, i.e., those changes that are subject to mission operations control and are known in advance rather than discovered in real time. This dynamic responsiveness in route computation should be significantly more effective and less expensive than static routing, increasing total data return while at the same time reducing mission operations cost and risk.

Note that the non-Critical forwarding load across multiple parallel paths should be balanced automatically:

- Initially all traffic will be forwarded to the node(s) on what is computed to be the best path from source to destination.
- At some point, however, a node on that preferred path may have so much outbound traffic queued up that no contacts scheduled within bundles' lifetimes have any residual capacity. This can cause route computation at that node to fail, resulting in custody refusal.
- Custody refusal causes the refusing node to be temporarily added to the current custodian's excluded neighbors list for the affected final destination nodes. If the refusing node is the only one on the path to the destination, then the custodian may end up sending the bundle back to its upstream neighbor. Moreover, that custodian node too may begin refusing custody of bundles subsequently sent to it, since it can no longer compute a forwarding path.
- The upstream propagation of custody refusals directs bundles over alternate paths that would otherwise be considered suboptimal, balancing the queuing load across the parallel paths.
- Eventually, transmission at the oversubscribed node relieves queue pressure at that node and enables acceptance of custody of a "probe" bundle from its upstream neighbor. This eventually returns the routing fabric to its original configuration.

Although the route computation procedures are relatively complex they are not computationally difficult, and the supporting CGRP messages are small and relatively few in number. The impact on both computation and transmission resources at the vehicles should be modest.

Finally, note that the information carried by CGRP messages is useful not only for dynamic route computation but also for the implementation of rate control, congestion forecasting, transmission episode initiation and termination, timeout interval computation, and retransmission timer suspension and resumption. This manifold utility lets us amortize the burden of CGRP traffic across multiple DTN operating capabilities, further improving the cost effectiveness of the protocol.

## VI. Conclusion

One fundamental principle of delay-tolerant networking is computational self-sufficiency, i.e., the making of communication decisions on the basis of locally available information that is already in place, rather than on the basis of information residing at other entities – who might not be able to respond to a query in a timely manner due to link interruption and/or signal propagation latency. The CGR route computation algorithm and supporting protocol are an attempt to apply this principle to the problem of dynamic routing in an interplanetary DTN. The proposed solution is somewhat complex, and ideas for simplifications that would not significantly degrade performance are welcome. In the meantime, CGR as currently conceived continues to be tested for suitability in future DTN-enabled space flight missions.

## Acknowledgments

The research described in this paper was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

## References

- <sup>1</sup>Cerf, V., et al., "Delay-Tolerant Network Architecture," IETF RFC 4838, informational, April 2007, URL: <http://www.ietf.org/rfc/rfc4838.txt>.
- <sup>2</sup>Jain, S., Fall, K., and Patra, R., "Routing in a Delay Tolerant Network," *ACM SIGCOMM*, Portland, OR, USA, 2004.
- <sup>3</sup>Lindgren, A., Doria, A., and Schelen, O., "Probabilistic Routing in Intermittently Connected Networks," *Proceedings of the The First International Workshop on Service Assurance with Partial and Intermittent Resources*, Fortaleza, Brazil, 2004.
- <sup>4</sup>Harras, K., Almeroth, K., and Belding-Royer, E., "Delay Tolerant Mobile Networks (DTMNs): Controlled Flooding Schemes in Sparse Mobile Networks", *IFIP Networking*, Waterloo, Canada, 2005.
- <sup>5</sup>Burns, B., Brock, O., and Levine, B. N., "MV Routing and Capacity Building in Disruption Tolerant Networks," *Proceedings of IEEE INFOCOM*, Miami, FL, USA, 2005, pp. 398-408.
- <sup>6</sup>Li, Y. et al, "DTGR: Disruption-Tolerant Geographic Routing for Wireless Ad Hoc Networks," *Simulation: Transactions of The Society for Modeling and Simulation International*, Vol. 82, No. 6, 2006, pp. 399-411.
- <sup>7</sup>Burgess, J., Gallagher, B., Jensen, D., and Levine, B. N., "MaxProp: Routing for Vehicle-Based Disruption-Tolerant Networks," *Proceedings of IEEE INFOCOM*, Barcelona, Spain, 2006.
- <sup>8</sup>Ott, J., Kutscher, D., and Dwertmann, C., "Integrating DTN and MANET Routing," *Proceedings of the ACM SIGCOMM CHANTS Workshop*, Pisa, Italy, 2006.
- <sup>9</sup>Demmer, M., and Fall, K., "DTLSR: Delay Tolerant Routing for Developing Regions," *ACM SIGCOMM Workshop on Networked Systems for Developing Regions (NSDR)*, Kyoto, Japan, 2007.
- <sup>10</sup>Balasubramanian, A., Levine, B. N., and Venkataramani, A., "DTN Routing as a Resource Allocation Problem," *Proceedings of ACM SIGCOMM*, Kyoto, Japan, 2007.
- <sup>11</sup>Zhang, Z., "Routing in Intermittently Connected Mobile Ad Hoc Networks and Delay Tolerant Networks: Overview and Challenges," *IEEE Communications Society Surveys and Tutorials*, Vol. 8, No. 1, 2006, pp. 24-37.
- <sup>12</sup>Ramanathan, R., Basu, P., and Krishnan, R., "Towards a formalism for routing in challenged networks," *Proceedings of ACM MobiCom workshop on Challenged Networks (CHANTS 2007)*, Montreal, Canada, 2007.
- <sup>13</sup>Scott, K., Burleigh, S., "Bundle Protocol Specification," IETF RFC 5050, experimental, November 2007, URL: <http://www.ietf.org/rfc/rfc5050.txt>.