Distributed Volume Management in Space DTNs: Scoping Schedule-Aware Bundle Routing

Olivier de Jonckère, Juan Andrés A Fraire, Scott Burleigh

Distributed Volume Management in Space DTNs: Scoping Schedule-Aware Bundle Routing

Olivier De Jonckère* Juan A. Fraire†‡, Scott Burleigh§
*LIRMM, Université de Montpellier, France
†Inria, INSA Lyon, CITI, UR3720, 69621 Villeurbanne, France
‡CONICET - Universidad Nacional de Córdoba, Argentina
§ D3TN U.S. Corp., Florida, 444 Brickell Avenue, Miami, FL 33131, USA

***Abstract*—This paper addresses critical improvements in the Schedule-Aware Bundle Routing (SABR) standard, pivotal for distributed space missions based on Delay-Tolerant Networking (DTN). With a focus on volume management, defined as efficiently allocating and utilizing the data transmission capacity of network contacts, we explore enhancements for distributed and scheduled DTNs. Our analysis begins by identifying and scrutinizing existing gaps in volume management within the SABR framework. We then introduce a novel concept coined *contact segmentation*, which streamlines the management of the transmission volumes. Our approach spans all network contacts, initial and subsequent, by unifying previously separate methods such as Effective Volume Limit (EVL), Earliest Transmission Opportunity (ETO), and Queue-Delay (QD) into a single process. Lastly, we propose a refined generic interface for volume management in SABR, enhancing the system's maintainability and flexibility. These advancements rectify current limitations in volume management and lay a foundation for more resilient and adaptable space DTN operations in the future.**

***Index Terms*—Contact Graph Routing, Delay-Tolerant Networks, Schedule-Aware Bundle Routing**

I. Introduction

Integrating Delay-Tolerant Networking (DTN) technologies aligns seamlessly with the projected requirements of future space networks [1], [2]. As highlighted in the Interagency Operations Advisory Group's (IOAG) report on Mars Communications Architecture [3], most nodes, whether on the surface or in orbit, must be equipped with DTN capabilities. The increasing focus on DTN is exemplified by projects such as NASA's Lunar Communication and Navigation System (LCRNS) [4], the European Space Agency's (ESA) Moonlight initiative [5], [6], and the collaborative LunaNet Interoperability Specification development by NASA and ESA [7]–[10]. These initiatives highlight the growing need for DTN capabilities in nodes, whether positioned on planetary surfaces or orbiting in space.

Protocol: DTN necessitates the Bundle Protocol (BP) agents for bundle forwarding [11], [12]. In DTN, the *bundles* (the protocol data units of BP) can be of arbitrarily large lengths (e.g. up to the gigabyte range). Moreover, the BP embodies three fundamental principles of DTN: *i*) it facilitates temporary data storage at intermediate hops, awaiting the availability of subsequent links, *ii*) BP operates completely asynchronously in that it does not assume responses from any downstream nodes, which may be located at interplanetary distances, and *iii*) BP is designed to ease protocol layering, being an overlay that can bridge different network using different protocol stacks. Moreover, selecting appropriate next-hop nodes in DTN hinges on making time-sensitive routing decisions. This is achieved by considering time-bounded contacts, a concept at the core of the Schedule-Aware Bundle Routing (SABR) standard established by the Consultative Committee for Space Data Systems (CCSDS) [13].

Contacts: Anticipated future relay nodes within the Solar System Internet are expected to have multiple interfaces to facilitate improved planet-to-surface and Direct-to-earth communications. However, the sporadic availability of these links, influenced by predictable orbital dynamics and planetary occlusions, presents a challenge in maintaining consistent communication. In DTN, a *contact* is defined as a forthcoming unidirectional transmission opportunity between two nodes, characterized by a start and end time and a mean nominal data rate. A contact's duration and mean data rate determines the *volume* of data a given contact can convey during the period. Meanwhile, a contact's signal propagation *latency* depends on the distance between the nodes and the signal propagation speed.

Routing: Routing over time-dependent and predictable contact topologies differs from classical Internet routing over static graphs. Contact Graph Routing (CGR) [14], [15] is a popular deterministic and distributed routing algorithm for predictable space networks, currently at the core of SABR [13] and implemented in NASA's Interplanetary Overlay Network (ION) stack [16]. In CGR, each node independently initializes and autonomously manages routing decisions based on its local knowledge of future transmission opportunities. Such knowledge is informed by a *contact plan* comprising the list of forthcoming contacts within a time horizon [17], [18]. CGR is powered by a modified version of Dijkstra's algorithm. Since each Dijkstra call yields a single *route* (a sequence of successive contacts from origin to destination), CGR incorporates Yen's K-shortest path (KSP), which generates a list of potential alternative routes [14]. The route with the lowest delivery time among those with sufficient volume is chosen for forwarding. Nevertheless, Yen's KSP encounters computational traceability challenges [19]. The Shortest-Path Tree routing for Space Network (SPSN) [20]–[22] approach replaces the KSP by tracking the contact volumes during

graph exploration for a given bundle, reducing the need for large route lists. Countermeasures to several other known issues were also introduced in the context of SPSN, such as computing trees rather than single destination routes for efficiency and better multicast integration.

Volume Management: While the contact plan informs CGR of future contacts and their maximum volume, it does not plan how such volume is consumed as bundles are sent (the bundle "transmission plan" is unknown at contact plan creation). Volume management is the task of effectively allocating a portion of a contact's volume for routing bundles. Correct volume consultation and updates can ensure availability for future transmissions. At routing time, volume consultation can help select the correct route for a bundle [23]. After queuing a bundle for a given route, the contact volumes of this route can be updated [24]. While volume management reduces to a resource allocation problem in a centralized operation scheme [25], achieving an accurate and synchronized volume vision in a distributed DTN remains an open challenge. The state-of-the-art volume management for distributed DTN involves three approaches typically associated with CGR: *i*) Effective Volume Limit (EVL), which linearly tracks the residual volume of a contact, *ii*) Earliest Transmission Opportunity (ETO), which aims at predicting the bundle transmission time based on local buffer status, and *iii*) Queue-Delay (QD), which extends ETO to consider the estimated remote buffer occupation.

Contribution: This work extends the state-of-the-art of volume management on different fronts. Firstly, we identify the core limitations of EVL, ETO, and QD. We analyze and argue that the effects are exacerbated when handling large bundles and when an end-to-end path presents subsequent overlapping contacts. Secondly, we present the *contact segmentation* volume management approach. Contact segmentation unlocks accurate volume annotations within a contact, integrating and improving EVL, ETO, and QD performance. We quantify the improvement using an extensive simulation campaign in realistic and challenging scenarios. Finally, we discuss and analyze volume management complexity and its role within SABR. We present a new volume management generic interface concept to enhance SABR's maintainability, adaptability, and flexibility.

Organization: The remainder of this paper is organized as follows. Section II presents the details of DTN and existing volume management techniques. Section III identifies and discusses the limitations of EVL, ETO, and QD. Section IV introduces our contact segmentation approach. Section V presents a generic interface for volume management. Section VI analyzes evaluation results obtained from a comprehensive simulation campaign. Section VII concludes this paper.

II. Background

Even though there is growing interest in routing for space DTN, the problem of volume management has received little attention. Nevertheless, such a feature is crucial in ensuring adequate utilization of scarce communication resources.

A. State-of-the-Art in Volume Management

1) Volume Management in CGR: CGR's implementation in the ION protocol stack has been the baseline since its introduction. ION leverages a dry run (defined below) technique to validate the route list in forwarding time. Also, ION incorporates the most sophisticated and validated volume management phases and techniques framed by EVL, ETO, and QD, as discussed below.

The concept of Dry Run: Determining a route's suitability requires calculating the projected bundle arrival at the destination under the conditions imposed by its contacts. These conditions are applied during a so-called "dry run", i.e., the end-to-end transmission of the bundle is played on the route contacts without modifications of the volume annotations (e.g., during CGR's route construction). The result of the dry run may be either a determination that the route cannot successfully deliver the bundle and must be discarded or else the calculation of metrics (projected arrival time, hop count, etc.) used in a tie-break manner as specified by the standard to detect the best candidate route. With the assumption that the volume management technique can vary while still being invoked at specific steps of the algorithm, the dry run can integrate additional conditions depending on the volume management scheme, such as including residual volume checking of each contact along the path for filtering (EVL), the expected transmission start time at the first hop imposed by the length of the transmission backlog queues (ETO), or the predicted transmission start time delay applicable at the nodes after the first hop (QD).

Volume Management aspects in CGR: ION sets the basis for current volume management in DTN, encompassing two integral aspects: (***i***) Volume **consultation**, vital during the constitution of the route candidate list, involves verifying unreserved capacity. It ensures the inclusion of only those contacts in a route where enough volume is available for a bundle transmission. (***ii***) Volume **update**, occurring when scheduling reserves volume, is crucial for keeping the volume data of each contact across routes to different destinations current and accurate.

Volume Management Features in CGR: In ION's implementation of CGR, the consumption of a bundle's capacity is represented through three distinct models:

1) EVL - Effective Volume Limit: It leverages a single value that saves the residual volume available for each contact. Such value is checked for filtering during the route validation in forwarding time and decreased after scheduling if the associated contact is part of the selected route. As a result, EVL enables the exclusion of routes involving contacts with insufficient remaining volume concerning the processed bundle [13]. As a result, EVL ensures that only feasible routes are considered during route selection.

2) ETO - Earliest Transmission Opportunity: It focuses on volume-dependent timing estimations. It calculates the earliest possible moment for the initial byte of a bundle's transmission. This calculation is based on the aggregated volume of all bundles queued for transmission in the outgoing queue, known

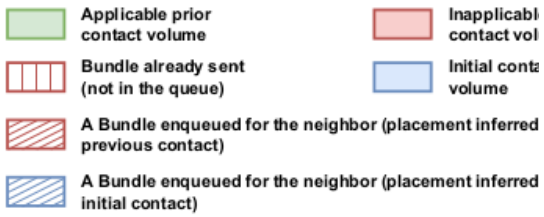

Fig. 1. Legend for figure 2 & 3.

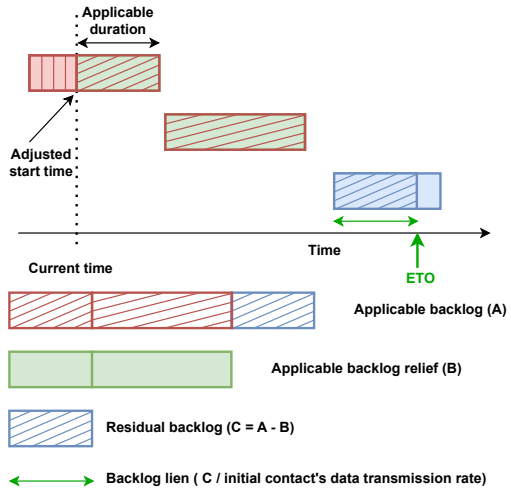

Fig. 2. ETO calculation example. First, the *applicable backlog* (A) is determined by totaling the sizes of bundles queued for the neighbor as per SABR 3.2.6.2.b. Next, the *applicable backlog relief* (B) is computed from the sum of *applicable prior contact volumes*, in line with SABR 3.2.6.2.f. Following this, the *residual backlog*—the expected remaining backlog at the onset of the initial contact—is calculated (C = A - B) according to SABR 3.2.6.2.g. The *backlog lien*, representing the transmission duration for the residual backlog over this contact, is then calculated per SABR 3.2.6.2.h. Finally, the ETO is established by adding the backlog lien to the adjusted start time of the initial contact, as specified by SABR 3.2.6.2.a and SABR 3.2.6.2.i.

as the backlog [26]. In SABR, bundles are enqueued for neighboring nodes rather than contacts. Therefore, the backlog can overlap several consecutive prior contacts with the neighbor before the route's first hop contact (also called initial contact) can take place. The principles of the ETO calculation are depicted in figure 2 and exemplified in its caption.

3) QD - Queue Delay: Expanding upon ETO, QD replicates the backlog computation feature at intermediary nodes down the selected route. In other words, it extends the concept of ETO by considering the time required to transmit the current volume reservation as the expected backlog at remote nodes. This allows for a more nuanced understanding of transmission timings throughout the network, not just at the initial contact point [27]. Notably, QD is not included in the SABR standard, setting it apart from EVL and ETO. In ION's CGR, QD is available in the CGR version developed by the University of Bologna [28].

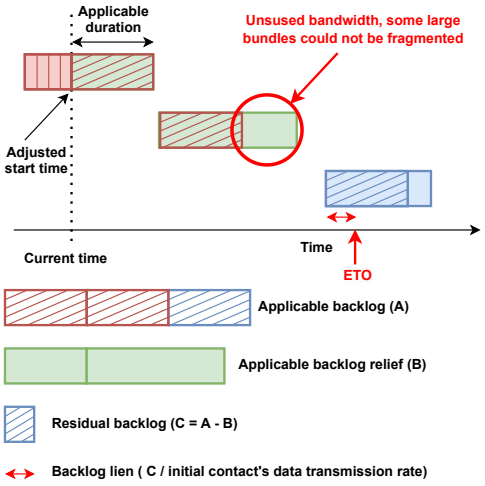

Fig. 3. The bundle depicted by the blue diagonally striped box cannot be fragmented; therefore, the second contact is not fully utilized. The applicable backlog does not cover a portion of its applicable prior volume. This volume portion would still be subtracted from the applicable backlog to calculate the residual backlog needed for ETO calculation.

State-of-the-Art Frontier: Even though ION made progress on volume management and the subsequent extensions championed by EVL, ETO, and QD, the topic is far from being solved. As discussed in the next section, we identified three main volume awareness issues.

III. Limitations

This section describes the limitations of volume management in space DTN.

A. Issue 1: Non-fragmentable bundle backlog

The ETO is derived from the difference between the cumulative volume of pending bundles awaiting transmission and the cumulative volume of all prior contacts with the neighboring node (those contacts that happen before the route's first hop contact and have not yet expired). When the pending volume surpasses the volume of the prior contact, it effectively "spills over" onto the route's initial contact. This is the expected case. Otherwise, one of those prior contacts could be used for transmission.

However, this mechanism's efficacy hinges on the assumption that all prior interactions' cumulative volume is fully utilized. This assumption holds only when all bundles can be fragmented or are sufficiently small to minimize the associated inaccuracies.

Figure 3 depicts a scenario (illustrated by the green portion of a contact) where a portion of the prior volume is not leveraged for transmission due to the inability to fragment the pending bundle (depicted in blue). In this case, the bundle represented by the blue diagonally striped box is too large to accommodate within the second contact's unused volume. Consequently, the expected backlog relief is not fully utilized,

leading to an inaccurate computation of the residual backlog leveraged to calculate the ETO for the bundle. This "backlog hole" thus signifies a limitation in current volume management strategies when dealing with indivisible bundles.

B. Issue 2: Backlog for large bundles at the first hop

Like in the previous issue, we consider large bundles, but this time, we assume they can be fragmented. However, we cover the cases when a bundle's scheduling time can occur during the transmission of another large bundle. The ETO is calculated from the sum of the bundles enqueued for transmission. This approach becomes inaccurate if one of these relatively large bundles requires seconds or minutes to complete transmission.

Suppose the forwarding function is invoked simultaneously for another bundle. In that case, it becomes necessary only to consider the remaining bytes for the transmission of the large bundle to calculate the backlog and then the ETO for the second smaller bundle.

For example, in the scenario depicted in figure 4, the first large bundle is in red and being transmitted (current time is t_0). The second bundle in blue is injected (received or generated) and must be scheduled. If we follow the SABR standard, the possible computed ETO values are:

- $ETO = t_{eto1}$: The red bundle gets out to the queue upon transmission start time, and the node S will assume that the blue bundle can be transmitted early. Transmission has already started for the red bundle, so the queue appears empty.
- $ETO = t_{eto2}$: The red bundle gets out to the queue upon transmission end time, and the node S will assume that the blue bundle can be transmitted late. Transmission has not ended yet, so the red bundle is still in the queue, and its total size is considered for ETO calculation.

However, the correct calculation would be $ETO = t_{eto3}$. In this toy example, the bundles can reach their destination, whatever ETO calculation is processed. However, the induced gap will affect the bundles scheduled down the path, as the incorrect volume booking is unrecoverable per the current SABR standard. The only exception is when the blue bundle has to be rescheduled for some reason (e.g., contact failure). In a rescheduling case, a new opportunity for an ETO calculation can render the correct result (e.g., no concurrent transmission of a long bundle).

C. Issue 3: Backlog at the following hops

Expanding the network's topology, such as adding more contacts, nodes, and node interfaces, makes this scenario increasingly complex regarding intermediary hops. Issue 3 deals with the intricacies of managing the backlog across multiple intermediary hops in a network with overlapping contact times.

When a bundle transmission begins later than the start of the outgoing contact in a subsequent node in the path—possibly because it's queued behind other bundles from different nodes—there's a risk that subsequent transmissions may be

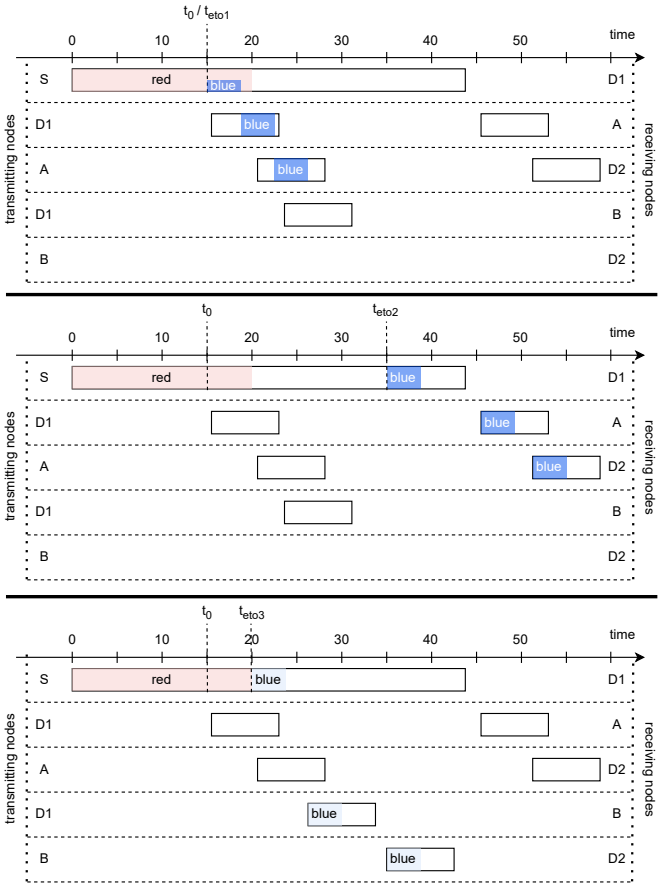

Fig. 4. The node S is the local node, and the first bundle in red was already scheduled for $D1$. The second bundle in blue is injected at t_0 and must be scheduled. The top sub-figure depicts the case where S assumes the red bundle out of the queue once its transmission started, so at t_0, the queue is already empty, and the calculated ETO coincides with t_0. The middle sub-figure depicts the case where S assumes the red bundle out of the queue when its transmission ends, so at t_0, the bundle is still present in the queue, and its whole size is used to calculate the ETO. The bottom sub-figure depicts the expected case if the volume management technique does not suffer from this issue. Faulty transmission plans are in dark color.

inaccurately scheduled. Typically, the QD feature can resolve such discrepancies by pushing forward the ETO for subsequent bundles at intermediary hops based on the volume already reserved on that contact.

However, inaccuracy arises when the contacts following a route overlap and the subsequent contact starts earlier. Under such conditions, the ETO predicted from the reserved volume at an intermediary contact assumes transmission starts earlier, thus not reflecting the actual volume utilization. QD would attempt to compute an ETO by advancing the transmission start time beyond the beginning of the contact, which might result in an ETO that is prematurely set. This miscalculation can lead to scheduling bundles for transmission when the contact is already booked.

A concrete example of a Mars scenario with this issue is visualized in figure 5. Here, a Mars relay orbiter exemplifies the challenge. It uses distinct interfaces for short-range (to

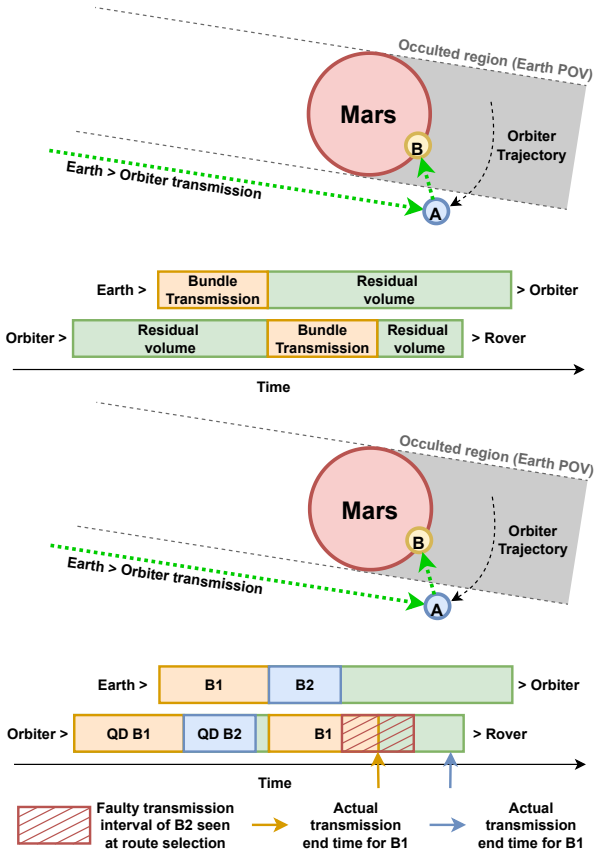

Fig. 5. The top sub-figure depicts the transmission of the first bundle (B1), and the bottom sub-figure presents the associated QD. The bottom sub-figure depicts the planned transmission of a second bundle (B2). The red-striped box depicts the faulty transmission plan, as seen during the route selection for B2. The queue delay induced by B1 (box QD B1) cannot postpone the planned transmission start time of B2 on the second contact because the contact starts too early. For simplicity, the delay induced by the range is omitted.

a rover) and long-range (back to Earth or another planet) communications. The illustration shows how a bundle intended for transmission in a contact that starts later can erroneously be scheduled for an earlier time due to overlapping contacts, highlighting the limitation of current ETO calculation methods in complex topologies.

IV. Contact Segmentation

A. Formalization

To address the volume management issues, we propose an approach that better reflects the physical effect of bundle scheduling through a link by tracking precisely the time intervals of contact utilization. Instead of calculating EVLs or delays, we represent a contact as a list of time splittable *segments*. A time segment represents a period of available bandwidth for this contact. The simple version of contact segmentation can be described as follows. The contact segmentation consists of tracking the intervals of bandwidth availability. If a bundle is transmitted for a given interval, the list of available intervals (or segments) should be updated.

We consider a contact $c = (S, E, R, D)$ with S its start time, E its end time, R its data rate, and D its propagation delay. We define its list of time intervals (the segments) $I_c = \{(s_0, e_0), (s_1, e_1), ..., (s_n, e_n)\}$ that verifies $\forall i \in [0, n]$:

- The interval starts at s_i and ends at e_i
- $S <= s_i < e_i <= E$
- $e_i <= s_{i+1}$ if $n > 1$ and $n \neq i$

This set of rules maintains the temporal coherence of the ordered list of time subintervals (or segments) and is sufficient for the contact segmentation approach in its simple formulation. The initialization of the list of segments for a contact c shall respect the contact's time bounds in this manner: $I_c = \{(S, E)\}$.

A dry run transmission on the contact c is processed as follows for a bundle of size V and a current time T:

1) Find the interval $x = (s_x, e_x) \in I_c$, that satisfies:
   - Condition 1: $max(T, s_x) + \frac{V}{R} <= e_x$
   - Condition 2: $\nexists (s_y, e_y) \in I_c$ that satisfies condition 1 with $s_y < s_x$.
2) If such an interval exists, the arrival time at the receiver node is equal to $max(T, s_x) + \frac{V}{R} + d_c$, where $max(T, s_x)$ is the transmission start time and $\frac{V}{R}$ is the transmission time.

Implementation is trivial and efficient thanks to the enforced ordering, which is done by iterating over the sorted intervals until one verifies condition 1.

Suppose the contact is part of the route elected as the best candidate route. In that case, all contacts of this route need to see their volume representation updated, more precisely, the interval mentioned above $x = (s_x, e_x) \in I_c$ to produce the updated list of segments I'_c:

1) Let $T_s = max(T, s_x)$ and $T_e = T_s + \frac{V}{R}$ and $\beta = \{(s_x, e_x)\}$
2) Let $\alpha = \{(s_x, T_s)\}$ if $T \neq T_s$ otherwise $\alpha = \emptyset$
3) Let $\gamma = \{(T_e, e_x)\}$ if $T_e \neq e_x$ otherwise $\gamma = \emptyset$
4) Let $I'_c = (I_c \setminus \beta) \cup \alpha \cup \gamma$,

where T_s is the transmission start time, T_e the transmission end time, $(I_c \setminus \beta)$ the new segment/interval list I'_c if T_s and T_e are equal to s_x and e_x (the whole segment is consumed), and the segments contained in α and γ the residual segments of availability if transmission starts later than s_x or ends earlier than e_x. The arrival time at the receiver is equal to $T_e + D$, but irrelevant for the segmentation operations. In simple topologies where consecutive contact overlapping does not occur, the contacts are likely to have a single segment (α would always be $\emptyset$ and γ in I'_c would replace β in I_c). In practice, the segment β is not replaced but just stripped by updating its start time from s_x to T_s. Also, $\alpha \neq \emptyset$ means the interval is split.

This stripping approach is depicted in figure 6. This approach is also suitable for first-hop contacts, replacing the Earliest Transmission Opportunity feature. Consequently, the routing algorithm is not required to access transmission queue information, reducing the deployment of SABR in the bundle agent.

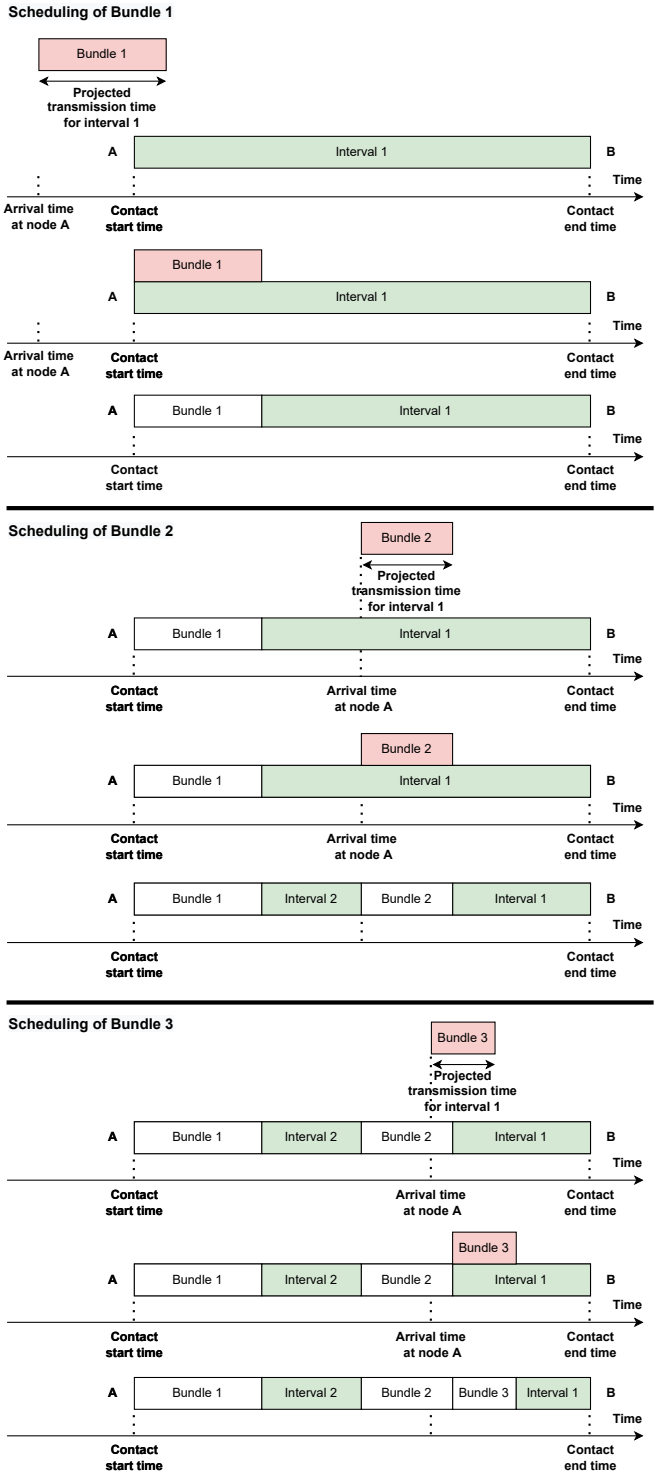

Fig. 6. Example of segment management upon scheduling of 3 different bundles. The segments in white are stored on the concerned bundles. The remaining available segments (in green) are stored on the contact.

V. Volume Management Interface

A. Motivation of a Generic Interface

CGR is an intricate algorithm that has received significant enhancement since its introduction. However, it was not originally architectured in a modular structure to facilitate extensions. In particular, the entanglement of volume management with route construction and route selection makes any research on volume management challenging, as a modification of the volume handling shall be associated with various similar modifications at different locations in the algorithm.

It would be appealing to simultaneously deploy several volume management techniques, as already attested in CGR (ETO for the first hop, queue delay for the next ones). Although contact segmentation can endorse both roles, previous volume management techniques should not be discarded. To this end, we propose further increasing flexibility by extending CGR interfaces to accommodate different volume management techniques for different contacts in the path.

Such flexibility would also help deal with variable link constraints during a contact. For example, the curve of a data rate over time exhibits a characteristic bell shape for Low-Earth Orbit (LEO) satellites, and a polynomial function could approximate this variation linearly. In this case, we could imagine implementing a segmentation flavor that calculates the transmission end time when given a bundle size and transmission start time (a function defined in contact planning). This would be handled by splitting the contact into several more minor contacts of different data rates or with the extended version of contact segmentation. This hypothetical "linear" manager could coexist with contacts using a classic contact segmentation manager for less challenging first-hop contacts.

In summary, we propose allowing the volume management technique to vary depending on the node and contact's physical characteristics. We explain an approach to achieve this in the following section.

B. Interface Definition within SABR

In this section, we describe the construction of a generic volume manager attached to the internal representations of the contact. Our generic volume manager leverages two processes: *Dry Run* and *Volume Update*. These are involved at two different locations within the SABR process:

1) In **route construction**: The *Dry Run* process calculates hop-by-hop distances. In this case, the bundle size can equal 0 if no volume management is needed during route constructions (e.g. when using Yen's algorithm). If volume-aware route constructions, such as those in SPSN, can be used, different bundle sizes can be used. This flexibility can be leveraged to address the volume management issues discussed above.
2) In **route selection**: During selection, each candidate route is evaluated by playing a dry run transmission from the source to the destination for the bundle to transmission. At each hop of a candidate, our *Dry Run* process can be reused. The calculation process

of metrics from one vertex to another during route construction is indistinguishable from a dry run of the route construction phase, with the sole difference that if the route is selected, the volume representation shall be updated afterward.

In contact segmentation, implementing this interface leverages the concepts described in Section IV. The Python prototype of SPSN uses such a principle systematically for simplicity, and an updated version of Rust is under development. The dry run and the volume update function of the Rust version are very similar, and merging them with an input flag is considered an implementation concern. Indeed, they share the same inputs:

- Some contact information like the start and end times.
- The associated volume manager for this contact.
- The current time at the sender.
- A bundle representation to wrap the potential interesting metrics supported by the manager (priority, size, fragmentation policy, etc.).

As well as the same outputs:

- The last byte transmission time.
- The delay (to calculate the last byte arrival time).
- The expiration time.

In some cases, the deployment of the volume management technique becomes trivial. For example, the Rust EVL-only version of the volume update represents two lines of code: a call to the dry run and an update of the internal representation of the residual volume in the EVL if the dry run succeeds. Data rates and delays are delegated to the manager. Storing the data rates and delays directly in the manager is appropriate: their representation depends on the volume management technique (e.g., storing different data rate segments to take into account data rate variation during the course of a contact).

The software layout of a SABR implementation that leverages the volume management interface is depicted in 7.

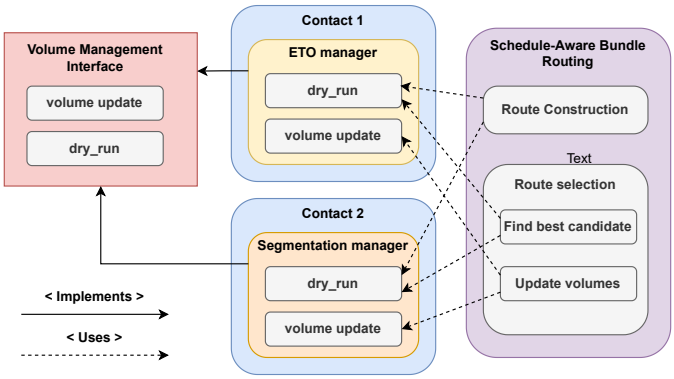

Fig. 7. Interactions between the proposed Volume Management Interface, the volume manager implementations, and SABR specification architecture.

VI. Evaluation

A. Simulation Environment

For our evaluation of diverse routing algorithms across multiple scenarios, we utilized *aiodtnsim*—an open-source simulator developed by Felix Walter, as detailed in [29] and [30], and accessible via [31]. The simulator, written in Python and leveraging the *asyncio* library, is distinguished for its ease of use. It ensures the reproducibility of simulations by enabling the external configuration of bundle injection schedules. It also allows for the straightforward integration of new routing algorithms by extending its node implementation classes. Simulations were run on a robust virtual machine equipped with a 24-core Intel(R) Xeon(R) Gold 6136 CPU @ 3.00GHz, ensuring the reliability and performance of the testing environment.

B. Evaluation Scenarios

For evaluating different routing algorithms, we utilize predictable topologies from the Ring Road Network (RRN) concept [32]. This concept captures the essence of satellite and ground node interactions for delay-tolerant communication with remotely located nodes. Satellites act as data mules, storing, carrying, and forwarding data from so-called isolated "cold spots" to and from "hot spots" connected to the Internet. The cost-effective and high-coverage nature of RRNs is especially suited for IoT on Earth and early planetary exploration phases. Indeed, an RRN-like architecture is envisioned for Martian network development [33]. The predictability of RRN makes them a perfect fit for CGR routing approaches.

Two scenarios for RRNs are dynamically generated using the *tvgutil* toolkit [34], also authored by Felix Walter and available open-source. This utility aids in crafting transmission plans and leverages real Low Earth Orbit (LEO) satellites from CelesTrak data [35] to simulate RRN scenarios. We consider topologies with 30 and 60 nodes and satellite-ground links captured over 48-hour periods.

A terrestrial scenario, sourced from a Public Transportation Network (PTN), is also included to evaluate the volume management variants in more dense and dynamic topologies. The public transportation scenario is created with 206 nodes comprising 89 transit stations and 117 buses over 8 hours with shorter, more frequent contacts that are easier to congest.

In summary, the scenarios we have chosen to evaluate the volume management techniques are:

- **RRN-30n** This RRN scenario envisions a network with 30 nodes, including 15 LEO satellites and 15 ground stations.
- **RRN-60n** An expanded RRN network scenario with 60 nodes: 30 LEO satellites paired with 30 ground stations.
- **PTN-206n** This varied PTN scenario depicts Freiburg's public transport system, comprising 206 nodes: 89 transit stations and 117 buses.

Topology-related metrics for each scenario are summarized in Table I. To evaluate the robustness of the design under stress, a level of congestion is artificially introduced across 20 distinct bundle injection plans per scenario (thus 20 simulations for each scenario). The source and destination are set randomly for each bundle.

TABLE I
SCENARIO PARAMETERS

Scenario Name	# of Nodes	Scenario Dur. (h)	Contact Count	Contact Dur. (s)	Data Rate (bit/s)	Total Bundles	Gen. Int. (s)	Bundle Size (bits)	Bundle TTL (s)	Last Bundle (h)
RRN-30n	30	48	3608	∼442	9600	1445	80	1.5M - 1.7M	86400	32
RRN-60n	60	48	22242	∼447	9600	11568	10	1.5M - 1.7M	86400	32
PTN-206n	206	8	16138	∼48	8000000	28740	1	44M - 46M	24000	8

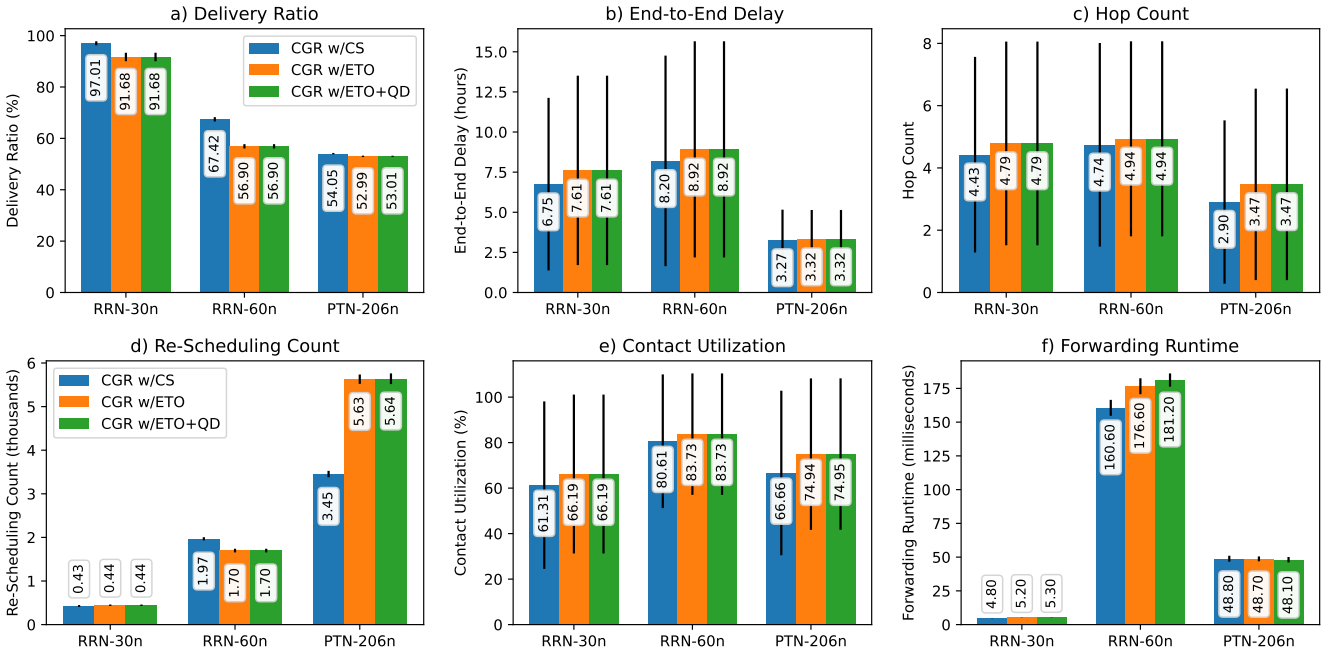

Fig. 8. Evaluation results across three scenarios (RRN-30n, RRN-60n, and PTN-206n) using three volume management techniques (CS, ETO, and ETO+QD). The top figures (a, b, and c) present the benefits in terms of delivery ratio, delay, and path length, respectively. The figures on the bottom (d, e, f) illustrate the costs of achieving the benefits: re-forwarding effort, contact utilization, and computational costs, respectively. In all cases, the bar plots present the averaged values across all nodes in the 20 simulations (bars include the average numerical annotation), and the error bars illustrate the standard deviation.

C. Volume Management Algorithms

The extensive scale of our scenarios and the computational intensity inherent to CGR's Yen algorithm necessitated an optimization technique to control computational overhead. We employed a contact limiting strategy, specifically the "first-ending" heuristic as outlined by Fraire et al. [23], which operates by excluding the earliest terminating contact from the most recently calculated route when determining subsequent ones. This optimization is distinct and separate from the volume management considerations, allowing for a consistent comparison of volume management efficacy across an identical pathfinding framework within CGR.

The **ETO** algorithm exclusively utilizes the volume management feature for the initial hops, adhering to the SABR specifications. ETO includes the abovementioned EVL approach at its core. In contrast, the QD algorithm extends ETO by incorporating the queue delay metric for subsequent hops. Thus, we refer to this variant as **ETO+QD**. On the other hand, our proposed **CS** algorithm integrates CGR with contact segmentation, enhancing the routing process.

It is crucial to highlight that constructing routing tables in these CGR variants does not account for volume consumption. Instead, when routes from these tables are selected, volume is considered only during forwarding. This is in contrast to the SPSN methodology, which incorporates volume considerations directly into the route calculation process. For a comprehensive analysis of the additional benefits the SPSN framework provides, see [22].

D. Assessment of Performance

a) Delivery Ratio: Congestion scenarios were introduced to underscore potential advantages in terms of delivery ratio, as depicted in figure 8 a). The CS method outperforms the alternatives across all test scenarios, showcasing its effective handling of network traffic. In the RRN-30n scenario, contact segmentation achieved a 97.01% delivery rate, surpassing the 91.68% delivery rate of both ETO and ETO+QD by more than 5%. In the more demanding RRN-60n scenario, the delivery rate for CS stood at 67.42%, compared to 56.90% for its counterparts, showing an improvement of approximately 10% in delivery rate. Interestingly, a marginal divergence in delivery rates between ETO and QD is observed in the PTN-206n scenario, with ETO achieving 52.99% and QD slightly edging

out at 53.01%. Despite this, CS maintains a modest lead, at a 54.05% delivery rate. The observed parallels in performance between ETO and QD can be attributed to the predominant influence of volume management on the initial hop in the given scenarios. Although both methodologies employ the ETO feature for the first hop in their routing, CS consistently applies this volume management strategy across all hops, distinguishing its approach from the others.

b) End-to-End Delay: Regarding delay, contact segmentation exhibits a noteworthy reduction, as shown in figure 8 b). The end-to-end delay for contact segmentation is lower across scenarios, clocking in at 6.75 hours for RRN-30n, 8.20 hours for RRN-60n, and 3.37 hours for PTN-206n. In comparison, both ETO and ETO + queue-delay exhibit longer delays of 7.61 hours and 8.92 hours for the first two scenarios, respectively, and a slightly lower delay of 3.32 hours for PTN-206n. As a result, we conclude that CS outperforms state-of-the-art volume management techniques by up to 11 % in terms of delivery delay.

c) Hop Count: Using contact segmentation reduces the hop count, as attested by figure 8 d). The bundles scheduled with contact segmentation require, on average, 4.43, 4.74, and 2.90 hops to reach their destinations, against 4.79, 4.94, and 3.47 hops with ETO and ETO + queue-delay on the RRN-30n, RRN-60n, and PTN-206n scenarios, respectively. These findings demonstrate that the CS method reduces path lengths from 7 to 16 percent, indirectly suggesting reduced energy consumption. This completes a solid benefits analysis across the delivery ratio, delay, and transmission count for our proposed CS scheme.

d) Re-Scheduling Count: These enhancements in delivery rates and delay reductions suggest that contact segmentation facilitates more effective routing decisions. figure 8 c) provides further insights on this aspect. For the PTN-206n scenario, contact segmentation significantly lowers the average number of bundles requiring rescheduling at a node to 3.45k from 5.63k and 5.64k for ETO and ETO + queue-delay respectively. A similar trend, though less pronounced, is observed in the RRN-30n scenario. However, this pattern does not extend to the RRN-60n scenario, where rescheduling instances are more frequent with contact segmentation, averaging 1.97k compared to 1.70 for the other algorithms. The observed gain in delivery ratio and delay suggest that contact segmentation implements an efficient dynamic re-forwarding scheme, effectively mitigating congestion by adapting bundle re-routing (thus avoiding dropping them) as the scenario demands.

e) Contact Utilization: figure 8 e) illustrates the contact utilization efficiency across the evaluated scenarios. In the RRN-30n and RRN-60n scenarios, contact utilization remains relatively consistent, hovering around the mid-60 percentiles for all algorithms. Notably, contact segmentation outperforms ETO and ETO + queue-delay by up to approximately 7.96% in contact utilization, showing its maximal gain for the PTN-206n scenario. These results indicate an appreciable gain in utilization efficiency as the network scales up. Like with hop count, contact utilization metrics also indicate a reduced energy usage for CS, as fewer transmissions are needed to achieve a higher delivery ratio and a lower delay.

f) Forwarding Runtime: The Forwarding Runtime is another critical system performance metric that indicates the computational cost of implementing volume management techniques. As presented in figure 8 f), the simulation runtimes for each scenario include inherent simulation overheads. Although more transmissions could skew the results, observable trends still provide valuable insights. With contact segmentation demonstrating marginally improved runtimes despite a reduced transmission count—which would typically result in lower runtimes—it can be inferred that, in this developmental phase, the differences are not substantial. We conclude that the more accurate volume management in CS does not come at increased computational costs concerning other schemes.

The computational cost findings might initially appear paradoxical, given that the contact segmentation approach amplifies the complexity of the graph model, which could be expected to increase computational demands. Yet, the results indicating reasonable computational pressure can be explained by recognizing that while a contact may be divided into multiple segments, navigating through these segments often incurs minimal cost. Commonly, the first segment is the optimal choice for forwarding a message, meaning the subsequent segments do not require additional consideration. As a result, the process does not typically involve a significant increase in computational effort compared to the methods that only employ ETO or ETO+QD. Essentially, the design of the contact segmentation algorithm allows it to operate efficiently despite the theoretically increased complexity.

We hypothesize that the observed computational efficiency might diminish in scenarios where all contacts along a path overlap, necessitating more interaction across segments as more bundles are injected and potentially escalating the total processing cost. Such circumstances could introduce additional computational load due to the increased number of segments that must be evaluated. This effect warrants further investigation, which we propose as future research to explore the impact of volume management on routing algorithm performance within scheduled space DTN environments.

VII. Conclusion

In this paper, we presented Contact Segmentation (CS) as a significant advancement in Volume Management for Delay-Tolerant Networking (DTN) within the Schedule-Aware Bundle Routing (SABR) framework. We provided an in-depth critical analysis of the state-of-the-art, which comprises Effective Volume Limit (EVL), Earliest Transmission Opportunity (ETO), and Queue-Delay (QD), and derived CS as an outperforming volume management technique. Also, we provided a generic interface to implement volume management in practical DTN protocol stack implementations.

Results validate an appealing cost-benefit ratio of our CS approach in realistic scenarios. CS delivers up to a 10% improvement in delivery ratios, ensuring reliable message completion. Furthermore, with reductions in delivery delay

of up to 11%, it substantially improves network latency. Notably, CS also achieves up to a 16% decrease in hop counts, which directly correlates to a reduction in energy consumption, which is paramount in the power-constrained realm of space networks. On the cost side, CS delivers adaptive rescheduling while reducing contact utilization penalties by, at most, an 8%, aligning with the algorithm's energy-saving benefits. Crucially, these performance improvements do not translate to a heightened computational cost within the tested scenarios.

Future work includes an extended evaluation campaign of CS in more scenarios, including deep-space links. Also, we envision integrating and evaluating EVO, ETO, and QD into the novel SPSN routing and comparing their performance in the context of SPSN and against their CGR counterparts evaluated in this paper.

VIII. Acknowledgment

We would like to sincerely thank Prof. Carlo Caini for his insightful and remarkably accurate review, which enhanced the quality of the manuscript. This research has received support from the European Union's Horizon 2020 R&D program under the Marie Skłodowska-Curie grant agreement No 101008233 (MISSION project) and the French National Research Agency (ANR) under the STEREO project ANR-22-CE25-0014-01.

References

[1] V. Cerf, S. Burleigh, A. Hooke, L. Torgerson, R. Durst, K. Scott, K. Fall, and H. Weiss, "Rfc 4838," *Delay-Tolerant Networking Architecture, IRTF DTN Research Group, April*, 2007.

[2] ——, "Delay-tolerant networking architecture," Internet Requests for Comments, RFC Editor, RFC 4838, April 2007. [Online]. Available: http://www.rfc-editor.org/rfc/rfc4838.txt

[3] "The future mars communications architecture," *Interagency Operations Advisory Group*, 2022. [Online]. Available: https://www.ioag.org/Public%20Documents/MBC%20architecture%20report%20final%20version%20PDF.pdf

[4] "LCRNS Project - TEMPO," https://tempo.gsfc.nasa.gov/projects/LCRNS/, 2023, accessed: 2024-03-15.

[5] P. Giordano, A. Grenier, P. Zoccarato, L. Bucci, A. Cropp, R. Swinden, D. Gomez Otero, W. El-Dali, W. Carey, L. Duvet *et al.*, "Moonlight navigation service-how to land on peaks of eternal light," in *Proceedings of the 72nd International Astronautical Congress (IAC), Dubai, United Arab Emirates*, 2021, pp. 25–29.

[6] "ESA's Moonlight: Connectivity and Secure Communications," https://www.esa.int/Applications/Connectivity_and_Secure_Communications/Moonlight, 2023, accessed: 2024-03-15.

[7] D. J. Israel, K. D. Mauldin, C. J. Roberts, J. W. Mitchell, A. A. Pulkkinen, D. C. La Vida, M. A. Johnson, S. D. Christe, and C. J. Gramling, "Lunanet: a flexible and extensible lunar exploration communications and navigation infrastructure," in *2020 IEEE Aerospace Conference*. IEEE, 2020, pp. 1–14.

[8] P. Giordano, R. Swinden, C. Gramling, J. Crenshaw, and J. Ventura-Traveset, "Lunanet position, navigation, and timing services and signals, enabling the future of lunar exploration," in *Proceedings of the 36th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2023)*, 2023, pp. 3577–3588.

[9] "NASA's Lunanet: Empowering Artemis with Communications and Navigation Interoperability," https://www.nasa.gov/humans-in-space/lunanet-empowering-artemis-with-communications-and-navigation-interoperability/, 2023, accessed: 2024-03-15.

[10] "Lunanet Interoperability Specification Version 5 - Draft," https://www.nasa.gov/wp-content/uploads/2023/09/lunanet-interoperability-specification-v5-draft.pdf?emrc=6f4483, 2023, accessed: 2024-03-15.

[11] K. Scott, S. Burleigh, and E. Birrane, "Bundle protocol version 7," Internet Requests for Comments, RFC Editor, RFC 9171, January 2022, http://www.rfc-editor.org/rfc/rfc9171.txt. [Online]. Available: http://www.rfc-editor.org/rfc/rfc9171.txt

[12] Consultative Committee for Space Data Systems (CCSDS), "Ccsds bundle protocol specification (blue book, recommended standard CCSDS 734.2-B-1," https://public.ccsds.org/Pubs/734x2b1.pdf, September 2015.

[13] B. Book, "Schedule-aware bundle routing," *Consultative Committee for Space Data Systems*, 2019.

[14] J. A. Fraire, O. De Jonckère, and S. C. Burleigh, "Routing in the space internet: A contact graph routing tutorial," *Journal of Network and Computer Applications*, vol. 174, p. 102884, 2021.

[15] S. Burleigh, "Contact graph routing," 2009. [Online]. Available: http://tools.ietf.org/html/draft-burleigh-dtnrg-cgr-00

[16] ——, "Interplanetary overlay network: An implementation of the dtn bundle protocol," 2007.

[17] F. D. Raverta, J. A. Fraire, P. G. Madoery, R. A. Demasi, J. M. Finochietto, and P. R. D'argenio, "Routing in delay-tolerant networks under uncertain contact plans," *Ad Hoc Networks*, vol. 123, p. 102663, 2021.

[18] J. A. Fraire, "Introducing contact plan designer: A planning tool for dtn-based space-terrestrial networks," in *2017 6th International Conference on Space Mission Challenges for Information Technology (SMC-IT)*. IEEE, 2017, pp. 124–127.

[19] O. De Jonckère, J. A. Fraire, and S. Burleigh, "On the tractability of yen's algorithm and contact graph modeling in contact graph routing," in *2023 IEEE International Conference on Wireless for Space and Extreme Environments (WiSEE)*, 2023, pp. 80–86.

[20] O. De Jonckère and J. A. Fraire, "A shortest-path tree approach for routing in space networks," *China Communications*, vol. 17, no. 7, pp. 52–66, 2020.

[21] O. De Jonckère, "Scalable schedule-aware bundle routing," 2023.

[22] O. De Jonckère, J. A. Fraire, and S. Burleigh, "Enhanced pathfinding and scalability with shortest-path tree routing for space networks," in *ICC 2023-IEEE International Conference on Communications*. IEEE, 2023, pp. 4082–4088.

[23] J. A. Fraire, P. G. Madoery, A. Charif, and J. M. Finochietto, "On route table computation strategies in delay-tolerant satellite networks," *Ad Hoc Networks*, vol. 80, pp. 31–40, 2018.

[24] P. G. Madoery, J. A. Fraire, and J. M. Finochietto, "Congestion management techniques for disruption-tolerant satellite networks," *International Journal of Satellite Communications and Networking*, vol. 36, no. 2, pp. 165–178, 2018.

[25] J. A. Fraire and E. L. Gasparini, "Centralized and decentralized routing solutions for present and future space information networks," *IEEE Network*, vol. 35, no. 4, pp. 110–117, 2021.

[26] N. Bezirgiannidis, C. Caini, D. P. Montenero, M. Ruggieri, and V. Tsaoussidis, "Contact graph routing enhancements for delay tolerant space communications," in *2014 7th advanced satellite multimedia systems conference and the 13th signal processing for space communications workshop (ASMS/SPSC)*. IEEE, 2014, pp. 17–23.

[27] C. Caini, G. M. De Cola, and L. Persampieri, "Schedule-aware bundle routing: Analysis and enhancements," *International Journal of Satellite Communications and Networking*, vol. 39, no. 3, pp. 237–249, 2021.

[28] https://gitlab.com/unibo-dtn/unibo-cgr.

[29] F. Walter and M. Feldmann, "Leveraging Probabilistic Contacts in Contact Graph Routing," in *IEEE Global Communications Conference (GLOBECOM)*, Waikoloa, HI, USA, 2019.

[30] F. Walter, "Prediction-enhanced Routing in Disruption-tolerant Satellite Networks," Doctoral Dissertation, Technische Universität Dresden, 2020. [Online]. Available: https://nbn-resolving.org/urn:nbn:de:bsz:14-qucosa2-721622

[31] https://gitlab.com/d3tn/aiodtnsim.

[32] M. Feldmann, J. A. Fraire, F. Walter, and S. C. Burleigh, "Ring road networks: Access for anyone," *IEEE Communications Magazine*, vol. 60, no. 4, pp. 38–44, 2022.

[33] O. Ledesma, P. Lamo, J. A. Fraire, M. Ruiz, and M. A. Sánchez, "Architectural framework and feasibility of internet of things-driven mars exploration via satellite constellations," *Electronics*, vol. 13, no. 7, p. 1289, 2024.

[34] https://gitlab.com/d3tn/dtn-tvg-util.

[35] http://www.celestrak.com.