CGR*: A* Search for Contact Graph Routing

Simon Rink*, Olivier De Jonckère†, Juan A. Fraire‡*, Holger Hermanns*
*Saarland University, Saarland Informatics Campus, Saarbrücken, Germany
†LIRMM, Université de Montpellier, France
‡Inria, INSA Lyon, CITI, UR3720, 69621 Villeurbanne, France

***Abstract*—Routing in space Delay-Tolerant Networks relies on Contact Graph Routing (CGR), standardized within the CCSDS Schedule-Aware Bundle Routing (SABR) framework. Existing SABR implementations use Dijkstra's algorithm, which ignores any information about the remaining cost to the destination and therefore explores many unnecessary states. We propose CGR*, an A*-based variant that exploits domain-specific heuristics to prune this search. Two heuristics are designed: h_{Delay}, an admissible heuristic suited to delay-dominated deep-space networks, and h_{Geo}, a greedy geographic heuristic for near-Earth networks, where contact availability rather than propagation delay drives arrival time. Experiments on realistic orbital scenarios show that h_{Delay} accelerates routing on deep-space networks by up to an order of magnitude with no loss of optimality, while h_{Geo} speeds up near-Earth routing at the cost of arrival times up to $\sim$2.5$\times$ the optimum — revealing cost-model design, rather than heuristic design alone, as the open challenge for A*-based routing in near-Earth scenarios. CGR* and its heuristics are released as part of the open-source A-SABR library.**

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

I. Introduction

Unlike terrestrial networks, space networks do not rely on the assumption that end-to-end communication is continuous and has low latency. Instead, they are subject to long *delays* (e.g., due to light-speed signal propagation in deep space networks (DSNs) [1], [2]) and frequent *disruptions* (e.g., caused by orbital mechanics in near-Earth networks (NENs) [3]). These characteristics necessitated the development of network protocols capable of addressing such challenges, now standardized in the Delay-Tolerant Networking (DTN) architecture [1].

Unlike TCP/IP-based networks, DTNs enable nodes to store, carry, and forward data as so-called *bundles*, which are retained locally until transmission becomes possible. This functionality is provided by the *Bundle Protocol (BP)*, deployed as an overlay on top of one or more underlying protocol stacks [4], [5].

Due to the disruptive yet partially predictable nature of space networks, communication between nodes is commonly modeled as episodic time intervals, referred to as *contacts*. These intervals are characterized by their transmitting and receiving nodes, start and end times, signal delay, and data rate. Predicted contacts are aggregated into a so-called *contact plan*.

These characteristics also require specialized routing mechanisms. In particular, the contact plan can be used to derive a time-varying graph representation, known as a *contact graph*. Based on this representation, *Contact Graph Routing (CGR)* [2], [3] has been proposed. CGR employs a time-dependent variant of Dijkstra's algorithm [6] to compute routing paths and has been standardized by the *Consultative Committee for Space Data Systems* (CCSDS) within the *Schedule-Aware Bundle Routing (SABR)* framework [7], where earliest delivery time serves as the primary optimization objective.

A key limitation of CGR is its reliance on Dijkstra's algorithm, which performs an uninformed search over the graph. Consequently, only the cost of reaching already discovered nodes is considered, while information about the remaining cost to the destination is ignored until the destination is reached. This can lead to the exploration of suboptimal paths.

Heuristic search algorithms such as A* [8] address this limitation by incorporating an estimate of the remaining cost to the destination. Specifically, A* employs a heuristic function that guides the search toward promising regions of the graph, thereby reducing the number of explored states. When the heuristic is efficient to compute, this can significantly improve computational performance. Based on these observations, integrating A*-based search into the CGR framework has the potential to substantially improve routing efficiency.

Contribution: We present a first approach to integrate A* search into the SABR framework and discuss the associated challenges. Because DSNs and NENs are governed by different cost regimes, distinct heuristic functions are required. We therefore introduce an admissible, consistent heuristic tailored to DSNs and a greedy, non-admissible heuristic designed for NENs. The resulting algorithm and both heuristics are implemented in the A-SABR library, a Rust-based state-of-the-art implementation of SABR-related algorithms [9], and evaluated on scenarios based on realistic orbital data derived from public Two-Line Element (TLE) datasets [10]. Our experiments confirm that the admissible delay-based heuristic delivers an order-of-magnitude reduction in state expansions on deep-space networks while preserving optimality, whereas the geographic heuristic trades route quality for speed on near-Earth networks — exposing cost-model design as the open challenge for A*-based routing in this regime.

Organization: The remainder of the paper is structured as follows: Section II discusses the required preliminaries for CGR, A-SABR, and A*. Section III presents the new approach together with the newly derived heuristic functions. Section IV presents the benchmark results, and Section V concludes the paper and outlines possibilities for future work.

II. Preliminaries

A. CGR

At its core, CGR comprises algorithms for DTN-specific routing, aimed at determining efficient source-to-destination routes. For this paper it suffices to distinguish two phases [2]: the *Route Computation Procedure (RCP)* and the *Route Selection Procedure (RSP)*.

The RCP generates candidate routes optimized primarily for bundle delivery time, as defined in the SABR standard [7], but does not account for practical constraints such as the bundle's volume or priority.

The RSP then evaluates these routes for feasibility under current network conditions, e.g., whether the required transmission volume exceeds the available capacity of any contact along the route. If no feasible route is found, the RCP is reinvoked for additional candidates, repeating until a suitable route is found, at which point the bundle is scheduled for transmission to the first hop of the selected path.

The RCP employs Dijkstra's algorithm to determine an optimal route with respect to the *time of arrival* at the destination. This differs from classical variants of Dijkstra's algorithm, where the objective typically consists of minimizing an accumulated edge cost. When the RSP rejects a route, the RCP must be reinvoked to compute alternatives. Early CGR implementations used Yen's K-shortest-path algorithm [11], [12] for this purpose; however, as identified in [13], this can incur significant computational overhead, as many of the generated alternatives are infeasible due to volume constraints.

To address these limitations, several enhancements have been proposed. Alternative pathfinding strategies, as presented in [14], exclude certain contacts from the Dijkstra search based on their anticipated end time or volume exhaustion. While this reduces complexity, it may also eliminate viable routes due to the complete exclusion of specific contacts [9].

A more fundamental approach consists of considering the bundle characteristics (including volume and priority) directly during the RCP. As a result, any route constructed for a specific bundle is guaranteed to be feasible, instead of relying on the RSP for validation. This bundle-aware construction was first introduced with the Shortest Path Tree Routing for Space Networks (SPSN) [15], [16], alongside other optimizations such as computing shortest-path trees rather than single destination paths. Building on this idea, Volume-CGR (Vol-CGR) [9] integrated bundle-aware construction into CGR's single-destination framework. By directly accounting for volume constraints, Vol-CGR eliminates the need for Yen's algorithm or alternative pathfinding. As demonstrated in [9], Vol-CGR achieves network performance comparable to SPSN while significantly improving computational efficiency. Consequently, CGR* is based on the Vol-CGR framework.

B. A-SABR

The Adaptive library for Schedule-Aware Bundle Routing (A-SABR) [9] is a modular Rust library implementing multiple CGR and SPSN variants. Its architecture separates routing into exchangeable components: pathfinding algorithms, distance metrics (defining the priority queue ordering), and contact resource managers.

A-SABR supports three *parenting classes* for its Dijkstra implementations, each representing a different graph abstraction [15]: *NodeParenting*, where nodes serve as vertices in the priority queue; *ContactParenting*, where contacts serve as vertices (the classical CGR approach); and *HybridParenting*, which uses contact-level tracking with node-based filtering. NodeParenting is computationally cheapest but may miss routes that ContactParenting discovers; HybridParenting balances both aspects. All three classes are evaluated in this work.

C. A*

The A* search algorithm [8] is closely related to Dijkstra's algorithm, but incorporates a heuristic function h estimating the remaining cost from a node n to the destination d, guiding the search toward the destination rather than merely away from the source. Its priority function is

$$p(n) = c(n) + h(n),$$

where $c(n)$ is the cost of reaching n from the start node. An informative and cheap-to-compute heuristic can significantly reduce the search effort [17].

To guarantee optimality with respect to the chosen cost metric, the heuristic function must be *admissible*, i.e., it must never overestimate the true cost from a node n to the destination d. Formally, this requirement is expressed as

$$\forall n \in V : \; h(n) \leq h^*(n),$$

where $h^*(n)$ denotes the true minimal cost from n to d. If this condition is violated, the algorithm may produce suboptimal solutions.

In addition to admissibility, informative heuristics should satisfy the property of *consistency* (or monotonicity), which is defined as

$$h(d) = 0 \;\wedge\; \forall (n_1, n_2) \in E : \; h(n_1) \leq c(n_1, n_2) + h(n_2),$$

where $c(n_1, n_2)$ is the cost of edge (n_1, n_2). Intuitively, the estimate must not drop faster than the actual transition cost between adjacent nodes, and the destination is assigned heuristic value 0.

Consistency is important in practice, as it guarantees that each node is expanded at most once; the optimal path to a node is therefore known as soon as it is first visited, avoiding re-expansion.

III. CGR*

In this section, we present the general concept underlying the proposed CGR* algorithm, followed by detailed descriptions of the heuristics designed for different network scenarios.

Algorithm 1 CGR* using NodeParenting

```
Require: Graph G, source s, bundle b w. dest. d, heuristic h
Q ← min-priority queue ordered by f(r)
Q.push(RouteStage(s, t_now, f = h(s, d)))
while Q not empty do
    r ← Q.pop()
    if r.node = d then
        return r
    end if
    for each neighbor v of r.node do
        r′ ← BestHop(r, b, v) {bundle-aware}
        if r′ exists and improves known cost to v then
            r′.f ← c(r′) + h(v, d)
            Q.push(r′)
        end if
    end for
end while
```

A. General Idea

According to the SABR standard, the primary optimization objective is the arrival time of a bundle at its destination. Consequently, the heuristics proposed in this work are exclusively designed to optimize bundle arrival time. Other optimization criteria, such as hop count, are left for future investigation. Since CGR* builds on the Vol-CGR framework (cf. Section II), we use the terms CGR* and VolCGR* interchangeably throughout this paper.

Algorithm 1 shows how A* search integrates into Vol-CGR's main loop, using Node Parenting as an example. The key modification is the priority function: instead of ordering by the actual cost metric $c(r)$ alone, the queue orders by $f(r) = c(r)+h(r.\text{node}, d)$. The bundle-aware hop construction (BestHop) ensures that only volume-feasible contacts are considered. Here, a *route stage* r is a partial route ending at node r.node with arrival time r.arrival; $c(r)$ is its accumulated cost and $h(r.\text{node}, d)$ the estimated remaining cost to the destination d. The stage priority $f(r)$ is thus the route-stage analogue of the node priority $p(n) = c(n) + h(n)$ from Section II.

In principle, integrating A* into CGR is straightforward, as it requires modifying only the priority function. The challenge therefore stems not from the integration itself, but from the design of suitable heuristics for the time-varying contact graph.

One of the most prominent applications of A* is shortest-path computation in terrestrial road networks [18], where the Euclidean distance between two nodes commonly serves as the heuristic function. This heuristic is admissible, since Euclidean distance constitutes the minimum possible distance between two points, and consistent due to the triangle inequality.

Motivated by these prior results, we similarly aim to construct distance-based heuristics, as such heuristics can be precomputed prior to route computation and retrieved during search with minimal computational and memory overhead. However, directly transferring this concept to space networks is nontrivial for two primary reasons:

1) In SABR, the optimization objective is bundle arrival time rather than physical distance between satellite nodes. While certain network classes, such as DSNs, exhibit a direct correlation between propagation delay and inter-node distance, other network types, such as NENs, may experience negligible transmission delay.
2) In terrestrial road networks, distances are typically assumed to remain constant, despite this being an idealization [18]. In contrast, space networks are inherently time-varying, causing inter-node distances to change significantly over time and thereby complicating the estimation of future contact opportunities.

To address these challenges, we derive two heuristics: an admissible heuristic based on communication delays, denoted h_{Delay}, intended for DSNs, and a greedy, non-admissible heuristic based on geographical distances, denoted h_{Geo}, designed for NENs. The following subsections describe both heuristics in detail.

B. Delay-Based Heuristic for DSNs

Deep Space Networks are characterized less by disruptions than by substantial propagation delays from the vast interplanetary distances. Transmission delay is therefore the dominant factor in routing decisions and derives directly from inter-node distance. As in road networks, any bundle must incur at least the minimum propagation delay between its current node and the destination, by the triangle inequality.

This motivates using the minimum delay between a node and the destination as the DSN heuristic. Because inter-node distances—and thus propagation delays—vary over time, it must use the *minimum possible* delay over the planning horizon to ensure admissibility. Formally, h_{Delay} for a node n is defined as:

$$h_{\text{Delay}}(n) = \text{min\_delay}(n, d),$$

where $\text{min\_delay}(n_1, n_2)$ denotes the minimum delay between nodes n_1 and n_2 over the planning horizon.

Since $\text{min\_delay}(n, d)$ lower-bounds the propagation delay of any route from n to d by the triangle inequality, and a bundle's arrival time never decreases along a route, h_{Delay} never overestimates the true cost and is therefore both admissible and consistent.

Note that in practice, interplanetary propagation delays are often assumed to be (nearly) constant over multiple contacts, with only contact availability varying dynamically over time [10]. For example, the one-way light time between Earth and Mars drifts by just a few minutes over hundreds of days [19] and is thus essentially constant over horizons of days to weeks.

Precomputing min_delay requires $\mathcal{O}(|V|^2)$ storage, where each entry is the minimum delay observed for the corresponding node pair across all contacts in the plan. This matrix is computed once during contact-plan parsing and remains valid for the plan's entire horizon, amortizing its cost across all subsequent routing queries.

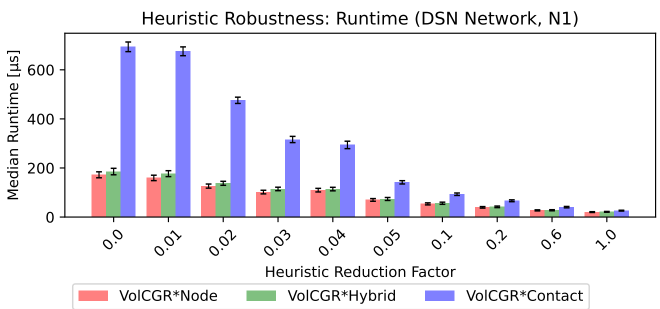

Fig. 1. Heuristic robustness on N1: median runtime of VolCGR* with h_{Delay} scaled by a reduction factor, routing from node 0 to all destinations. Factor 1.0 is the full heuristic; 0.0 disables guidance.

C. Geographical Distance-Based Heuristic for NENs

While h_{Delay} exploits a direct relationship between propagation delay and arrival time, NENs present a fundamentally different challenge: delays are often negligible (assumed to be 0 [10]), and arrival time is instead dominated by *contact availability*, which depends on orbital dynamics. This means h_{Delay} provides no useful guidance in NENs, and constructing an admissible arrival-time heuristic likely requires a deeper reformulation of the cost model, revisited in Section V.

To test whether A* offers value in NENs without such a reformulation, we propose h_{Geo}, a deliberately simplistic greedy heuristic based on geographical distance: since contacts in NENs require proximity, minimizing geographical distance may correlate with shorter arrival times, in line with procedures like DQN [20] that forward bundles to the geographically closest neighbor(s). Treating h_{Geo} as a stand-in for a future admissible heuristic lets us quantify the quality/efficiency trade-off A* can offer in this regime.

This heuristic, h_{Geo}, requires replacing the arrival-time cost model with one based on cumulative geographical distance, since no direct relationship between distance and arrival time exists. The heuristic is defined as the *current* geographical distance between the node and the destination:

$$h_{\text{Geo}}(n,t) = \text{geo\_distance}(n,d,t),$$

where $\text{geo\_distance}(n_1, n_2, t)$ denotes the distance between n_1 and n_2 at time t.

This approach has two important limitations:

1) **Non-admissibility.** The current distance can overestimate the distance a bundle must traverse (e.g., if nodes are moving toward each other), and contacts do not require zero distance. This violates both admissibility and consistency, so optimality is not guaranteed.
2) **Greedy cost model.** The assumption that minimizing cumulative geographical distance also minimizes arrival time does not hold in general, and can lead to suboptimal routing plans.

IV. Evaluation

A. Setup

Hardware. All experiments were conducted on an Apple MacBook Pro with an M4 Max 16-Core Processor and 48 GB main memory.

Library. We implemented CGR* and the two heuristics introduced in Section III within A-SABR [9][1], a Rust-based library implementing state-of-the-art SABR architectures. Given A-SABR's modular architecture (cf. Section II), the implementation required adding heuristic modules and a heuristic-augmented distance metric, without modifying the pathfinding core. In all experiments, we run both VolCGR and VolCGR* for all three parenting classes.

Baseline. We compare exclusively against Vol-CGR, the state-of-the-art exact schedule-aware router that dominates Yen-based CGR and SPSN in efficiency at comparable network performance [9] and shares CGR*'s earliest-arrival objective; as CGR* differs from it only in the priority function, any observed speedup is attributable solely to the heuristic. Replication- or probability-based DTN schemes (e.g., DQN [20]) instead optimize delivery ratio under uncertain topologies rather than exact schedule-aware routes, and are therefore not comparable here.

Utilized Networks. We evaluate on three networks derived from realistic orbital data generated with dtn-tvg-util [10]. Their characteristics are summarized in Table I.

TABLE I
NETWORK CHARACTERISTICS.

	$\|V\|$	$\|E\|$	Horizon	Type
N1	63	3,356	1 week	DSN
N2	40	900	1 day	Hybrid
N3	45	750	1 day	Ring-road

Since h_{Delay} assumes a delay-dominated regime and h_{Geo} assumes a geometry-dominated regime, we evaluate the former on N1 (pure DSN) and N2 (hybrid, with delay-dominated inter-planetary links bridging short-delay planetary subnetworks), and the latter on N3 (near-Earth ring-road).

Methodology. To characterize scalability, we randomly remove nodes from each network until only $10\%, 20\%, \ldots, 100\%$ of nodes remain, ensuring the resulting graph stays connected. For each network size, we compute the route from node 0 (an arbitrary source; spot-checks with other sources gave qualitatively similar trends) to every reachable destination and report the median across destinations. Each configuration is repeated 10 times; bar heights show the average across repetitions and error bars one standard deviation.

B. h_{Delay}

As shown in the left panel of Figure 2, VolCGR* substantially reduces runtime over VolCGR across all network sizes on N1, the gap widening as the network grows, driven by the reduction in state expansions (right panel). The relative ordering

[1] https://github.com/SimRi99/A-SABR/tree/astar

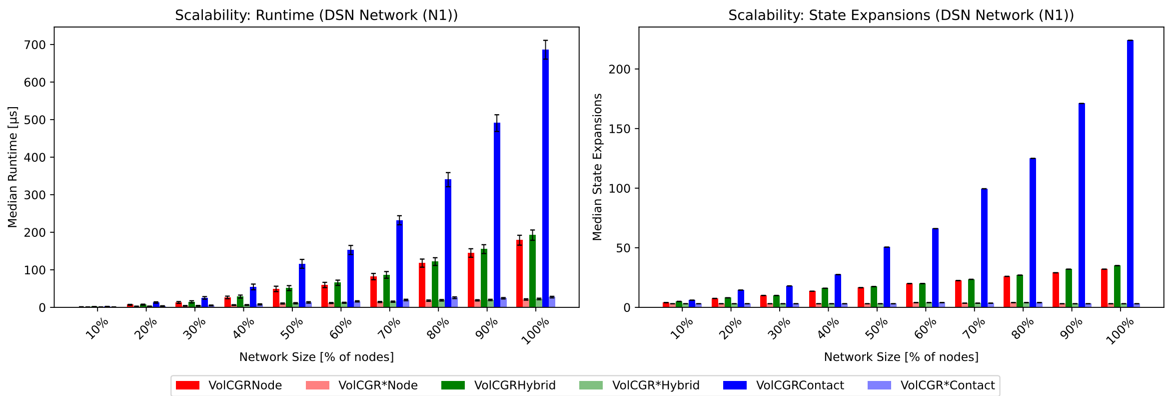

Fig. 2. Scaling of runtime (left) and number of state expansions (right) with network size, for VolCGR and VolCGR* under h_{Delay} on network N1. Error bars show one standard deviation across 10 repetitions.

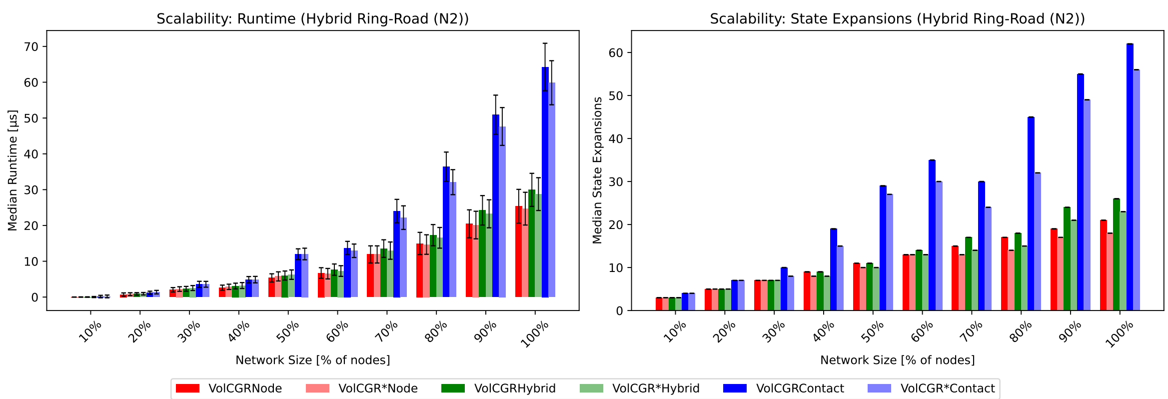

Fig. 3. Scaling of runtime (left) and number of state expansions (right) with network size, for VolCGR and VolCGR* under h_{Delay} on network N2.

of the parenting classes (NodeParenting, then HybridParenting, then ContactParenting) is preserved. Since h_{Delay} is admissible and consistent, VolCGR* returns the same optimal routes — the speed-up comes at no quality cost.

In contrast, on the hybrid network N2 (Figure 3), the benefits of VolCGR* are much smaller. Most of N2's traffic stays within planetary subnetworks, where propagation delays are short and h_{Delay} provides little guidance. The reduction in state expansions is therefore concentrated in ContactParenting, where the search space is largest and even a weak heuristic prunes substantially; for NodeParenting and HybridParenting the gain is marginal and is partly offset by the heuristic-evaluation overhead.

C. Heuristic Robustness

To estimate how much the speed-up depends on the precision of h_{Delay}, we scale it by a uniform reduction factor ≤ 1 (affecting expressiveness, not admissibility) and route from node 0 to all destinations in N1. As seen in Figure 1, the benefit degrades gracefully: even a small factor (≥ 0.03) prunes most of the search and performance is nearly unchanged down to 0.05; only for factors ≤ 0.01 does it collapse to VolCGR levels, where the heuristic adds overhead without guidance. Thus h_{Delay} is robust to imprecise estimates, losing guidance only very close to 0.

-

D. h_{Geo}

As seen in the left panel of Figure 4, VolCGR* is able to outperform VolCGR in terms of computational efficiency for all parenting classes. However, the right panel reveals a substantial cost: while the greedy cost model yields competitive routing plans at small network sizes, for larger networks the arrival time rises to roughly 2–2.5$\times$ the optimal VolCGR baseline. These results illustrate that A* can accelerate routing in NENs even with a simplistic heuristic, but they also expose a clear quality/efficiency trade-off and motivate the more sophisticated cost models discussed in Section V.

V. Conclusion and Future Work

In this paper, we have presented CGR*, an initial approach to integrating A* search into the state-of-the-art CGR variant Vol-CGR. We introduced an admissible and consistent heuristic, h_{Delay}, whose estimates are derived from transmission delays and are therefore primarily suited to DSNs. In addition, we proposed a greedy, non-admissible heuristic, h_{Geo}, designed for NENs, in which estimates of future arrival times are

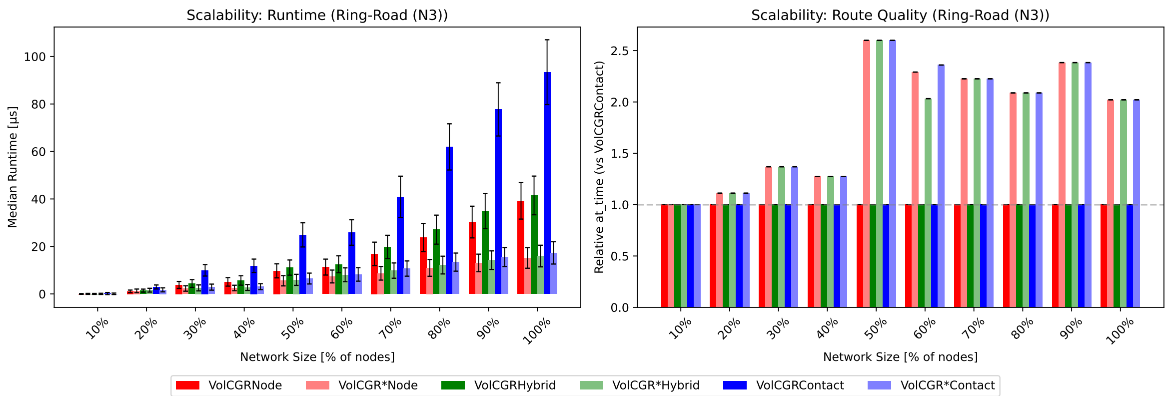

Fig. 4. Scaling of runtime (left) and route quality (right, arrival time relative to the optimal VolCGR baseline) with network size, for VolCGR and VolCGR* under h_{Geo} on network N3.

non-trivial to obtain; this heuristic greedily minimizes the geographical distance that satellites must traverse in order to forward a bundle to its destination.

Our results show that h_{Delay} accelerates route computation on DSNs and (more modestly) on hybrid networks while preserving optimality, whereas h_{Geo} reduces computation time on NENs at the cost of arrival times up to roughly $2.5\times$ the optimal baseline.

The integration of A* into CGR nonetheless remains in its early stages. The core challenge for NENs is that arrival time depends on contact availability rather than propagation delay, making it difficult to construct admissible heuristics without a deeper reformulation of the cost model.

Promising directions for future work include: (i) incorporating orbital dynamics directly into the heuristic to derive coarse lower bounds on future link availability; (ii) time-bucketed reachability estimates, where the contact plan horizon is partitioned into intervals and per-interval reachability is precomputed; and (iii) data-driven heuristics that regress an arrival-time estimate from past exact queries on recurring contact plans, guaranteeing admissibility, e.g., by flooring the learned value with a provable lower bound such as h_{Delay}.

References

[1] S. Burleigh, A. Hooke, J. Torgerson, K. Fall, V. Cerf, B. Durst, K. Scott, and H. Weiss, "Delay-tolerant networking: an approach to interplanetary internet," *Comms. Magazine, IEEE*, vol. 41, pp. 128 – 136, 07 2003.

[2] J. Fraire, O. Jonckère, and S. Burleigh, "Routing in the space internet: A contact graph routing tutorial," *Journal of Network and Computer Applications*, vol. 174, p. 102884, 01 2021.

[3] S. Burleigh, "Contact graph routing," NASA Jet Propulsion Laboratory, Tech. Rep., 2009. [Online]. Available: https://ntrs.nasa.gov/citations/20120006508

[4] S. Burleigh, K. Fall, and E. Birrane, III, "Bundle protocol version 7," Internet Engineering Task Force (IETF), RFC 9171, 2022.

[5] Consultative Committee for Space Data Systems (CCSDS), "Ccsds bundle protocol specification," CCSDS Secretariat, Tech. Rep. CCSDS 734.2-B-1, 2015.

[6] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[7] Consultative Committee for Space Data Systems (CCSDS), "Schedule-aware bundle routing," CCSDS Secretariat, Tech. Rep. CCSDS 734.3-B-1, 2009. [Online]. Available: https://ccsds.org/Pubs/734x3b1.pdf

[8] P. Hart, N. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968. [Online]. Available: https://doi.org/10.1109/tssc.1968.300136

[9] O. de Jonckère, L. Ma, and J. A. Fraire, "A-sabr: The adaptive library for schedule-aware bundle routing," in *2025 IEEE International Conference on Wireless for Space and Extreme Environments (WiSEE)*, 2025, pp. 1–6.

[10] "Dtn-tvg-util." [Online]. Available: https://gitlab.com/d3tn/dtn-tvg-util

[11] J. Y. Yen, "An algorithm for finding shortest routes from all source nodes to a given destination in general networks," *Quarterly of Applied Mathematics*, vol. 27, pp. 526–530, 1970.

[12] E. L. Lawler, "A procedure for computing the k best solutions to discrete optimization problems and its application to the shortest path problem," *Manage. Sci.*, vol. 18, no. 7, p. 401–405, Mar. 1972. [Online]. Available: https://doi.org/10.1287/mnsc.18.7.401

[13] O. Jonckère, J. 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)*, 09 2023, pp. 80–86.

[14] J. Fraire, P. Madoery, A. Charif, and J. Finochietto, "On route table computation strategies in delay-tolerant satellite networks," *Ad Hoc Networks*, vol. 80, 07 2018.

[15] 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.

[16] O. Jonckère, J. Fraire, and S. Burleigh, "Enhanced pathfinding and scalability with shortest-path tree routing for space networks," in *2023 IEEE International Conference on Communications (ICC)*, 05 2023, pp. 4082–4088.

[17] D. Rachmawati and L. Gustin, "Analysis of dijkstra's algorithm and a* algorithm in shortest path problem," *Journal of Physics: Conference Series*, vol. 1566, p. 012061, 06 2020.

[18] M. Alam and O. Faruq, "Finding shortest path for road network using dijkstra's algorithm," *Bangladesh Journal of Multidisciplinary Scientific Research*, vol. 1, pp. 41–45, 07 2019.

[19] K. T. McBrayer, P. R. Chai, and E. L. Judd, "Communication delays, disruptions, and blackouts for crewed mars missions," in *ASCEND 2022*. American Institute of Aeronautics and Astronautics, 2022, nASA Langley Research Center, NTRS 20220013418. [Online]. Available: https://ntrs.nasa.gov/citations/20220013418

[20] R. Diana, E. Lochin, L. Franck, C. Baudoin, E. Dubois, and P. Gelard, "Dtn routing for quasi-deterministic networks with application to leo constellations," *International Journal of Satellite Communications and Networking*, vol. 35, no. 2, pp. 91–108, 2017. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/sat.1159