



HAL
open science

A Stability-First Approach to Running TCP Over Starlink

Gregory Stock, Juan A Fraire, Santiago Henn, Holger Hermanns, Andreas Schmidt

► **To cite this version:**

Gregory Stock, Juan A Fraire, Santiago Henn, Holger Hermanns, Andreas Schmidt. A Stability-First Approach to Running TCP Over Starlink. 2024 IEEE International Conference on Communications Workshops (ICC Workshops), Jun 2024, Denver, France. pp.1708-1713, 10.1109/ICCWorkshops59551.2024.10615714. hal-04711313

HAL Id: hal-04711313

<https://hal.science/hal-04711313v1>

Submitted on 26 Sep 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution 4.0 International License

A Stability-first Approach to Running TCP over Starlink

Gregory Stock* , Juan A. Fraire*^{†‡} , Santiago Henn[†] , Holger Hermanns* , and Andreas Schmidt* 

*Saarland University – Computer Science, Saarland Informatics Campus, Saarbrücken, Germany

[†]CONICET – Universidad Nacional de Córdoba, Córdoba, Argentina

[‡]Inria, INSA Lyon, CITI, UR3720, 69621 Villeurbanne, France

Abstract—The end-to-end connectivity patterns between two points on Earth are highly volatile if mediated via a Low-Earth orbit (LEO) satellite constellation. This is rooted in the enormous speeds at which satellites in LEO must travel relative to the Earth’s surface. While changes in end-to-end routes are rare events in stationary and terrestrial applications, they are a dominating factor for connection-oriented services running over LEO constellations and mega-constellations. This paper discusses how TCP-over-constellations is affected by the need for rerouting and how orbital route selection algorithms impact the end-to-end performance of communication. In contrast to the state of the art that primarily optimizes for instantaneous shortest routes (i.e. lowest delay), we propose several algorithms that have route stability and longevity in their focus. We show that this shift in focus comes with vastly improved end-to-end communication performance, and we discuss peculiar effects of the typical TCP-like implementations, taking inspiration from the Starlink constellation in our empirical investigations. The spectrum of algorithms proposed provides a basis for co-designing suitable orbital route selection algorithms and tailored transport control algorithms.

I. INTRODUCTION

Low-Earth orbit (LEO) mega-constellations are thriving. Over half of the orbiting satellites are part of large-scale networked fleets (e.g. Starlink, Project Kuiper) aiming at global high-throughput internet connectivity and potentially other services. This development poses new challenges in network management to achieve both *low delay* and *high throughput*.

Traditional routing algorithms have predominantly honed in on optimizing the instantaneous state of the network, aligning with the stochastic nature of the internet. There is a plethora of works on the stability of routes on the internet [1]–[4], as well as the interaction of this stability with transport layer performance [5], [6].

The dynamics in LEO however induces rapidly changing end-to-end routes, characterized by highly volatile round-trip times (RTTs) when traversing through diverse sets of inter-satellite links (ISLs) [7]. Furthermore, ground stations are forced to continually switch access satellites as visibility periods are generally short-lived [8], albeit being nearly periodic. As a consequence, cross-orbit routes exhibit a high degree of variability and are of limited stability over time.

Thus, it becomes obvious that route stability within a satellite mega-constellation is an important factor in ensuring the efficacy of end-to-end protocols. Furthermore, unavoidable route changes need to be set up in such a way that they mitigate prevalent issues. One important such issue is the reordering of packets caused by overtaking, which can happen if the new

route is shorter in delay. This can have a detrimental impact on communication reliability and efficiency [9], rooted in the fact that in every standard Transmission Control Protocol (TCP) implementation such a reordering is likely to lead to *duplicate acknowledgements (ACKs)*. But also changes to considerably slower routes can trigger adverse effects due to *timeouts* [10]. While the individual handling is partly up to the TCP variant, both effects tend to lead to a substantial reduction in the in-flight packet window size and hence a reduction in achievable data rate.

On the other hand, routing solutions tailored for satellite constellations can leverage the predictably periodic nature of orbital networks to address stability issues adeptly. This is in contrast to terrestrial mobile networks that also have dynamics, but which are less predictable as space constellations.

To the authors’ knowledge, a route selection procedure that judiciously considers the instability inherent in routes within mega-constellations remains an open research topic [9]. In response, this paper introduces a family of algorithms, offering a spectrum of solutions to navigate the trade-offs between *end-to-end delay* and *route stability* in satellite communication. Despite their varied computational complexities, these algorithms are unified in their objective to enhance route stability and longevity, thereby elevating overall communication performance within in-orbit networks. The contributions of this work are as follows:

- We provide a background on route stability and its consequential impact on transport-layer network functions.
- We introduce a new evaluation framework to compute route stability figures and apply them to TCP-like communication models within Walker Delta constellations.
- We propose a suite of route selection algorithms, each with distinct computational complexities, and articulate the trade-offs between end-to-end delay and route stability.
- We comprehensively evaluate the well-known Starlink constellation, providing insights into stability metrics and end-to-end performance in TCP-like communication.

The remainder of this paper starts off with detailing our methodological framework in Section II, followed by a description of the different route selection algorithms in Section III. Empirical results are summarized in Section IV, and Section V discusses the findings and draws final conclusions.

II. CONTEXT & METHODOLOGICAL FRAMEWORK

In this section, we present the contextual background together with the methodological details of our evaluation framework.

arXiv:2408.07460v1 [cs.NI] 14 Aug 2024

A. Walker Delta Constellations & Orbital Dynamics

This study focuses on Walker Delta constellations due to the popularity of this constellation type for upcoming mega-constellations. All satellites in a Walker Delta constellation follow circular orbits and share the same altitude h and inclination α . They thus move at the same speed. The satellites are distributed across P evenly-spaced orbital planes, where each plane contains Q evenly-spaced satellites. Formally, we describe Walker Delta constellations by $\alpha: P \cdot Q/P/F$, where F specifies the relative phase shift between adjacent planes. Each satellite can be distinguished at any time as either ascending or descending, depending on whether they move from South to North or from North to South, respectively.

We assume that each satellite maintains four permanent inter-satellite links: two intra-plane links (to the successor and predecessor in same orbital plane) and two inter-plane links (to the left and right neighbour on the adjacent planes) [11]. So, from the perspective of an individual satellite, the underlying connection topology resembles that of a Manhattan Street Network that gets distorted at the most polar positions, where satellites switch from ascending to descending and vice versa.

For the routing algorithmics, future satellite positions need to be predicted. We will use a simple two-body propagator, i.e. assuming ideal conditions and unperturbed central force motion. This means that all orbital elements (except the argument of latitude) are assumed constant over time. More sophisticated propagators, such as SGP4, could be used, but the additional precision comes with a computation time penalty without having a significant effect on the analysis results.

B. Efficiency and Stability Metrics

We work with the following metrics to measure different aspects of efficiency and stability.

Route Delay: The route delay is determined by transmission delay, propagation delay, and queuing delays. We define the one-way delay (OWD) of a route as $d_{ow} = d_p + d_t + d_q$, where d_p is the propagation delay (i.e. the time the signal needs to travel, depending only on the physical distance), d_t is the transmission delay (i.e. the time to send the whole packet, depending only on the packet size and the data rate of the link), and d_q the queueing delay, an additional per-hop overhead for on-board processing and packet queueing. We assume store-and-forward-switching and consider a packet size of 1.5 kB (standard Ethernet frame) and 1 Gbit/s links for all satellites and ground stations. This results in a transmission delay of $d_t = \frac{1.5 \text{ kB}}{1 \text{ Gbit/s}} = 12 \mu\text{s}$. Further, we assume $d_q = 1 \text{ ms}$ for queueing and on-board processing. Assuming that the signals travel with the speed of light c , the propagation delay for some distance l is given by $d_p = \frac{l}{c}$.

Route Validity: Route validity is defined as how long a particular route remains accessible. Disregarding node failures, any route between two satellites in a Walker Delta constellation is valid ad infinitum since neighbourhood remains unchanged (while link distances oscillate across planes). However, end-to-end routes between two ground stations are always limited by the visibility periods of the two access satellites (i.e.

the first and last satellite of the route that directly communicate with the ground). Generally, the aim is maximizing end-to-end route validity to obtain less frequent route changes.

Delay Delta and Bad Changes: The delay delta is defined as $d_{ow}^{new} - d_{ow}^{old}$ and specifies by how much the delay changes at the points in time in which route changes occur. Negative delay delta values (switching from high to low delay) cause out-of-order arrivals, disrupting the expected sequential delivery of packets at the transport layer, particularly impacting protocols like TCP that rely on ordered delivery. If three duplicate acknowledgements are received, this scenario can trigger mechanisms such as TCP fast retransmission. Conversely, positive delta values (switching from low to high delay) can provoke unnecessary retransmissions at the sender, as the sender might interpret the increased delay as packet loss if acknowledgements from the receiver are not received within the expected time frame. Based on recent RTT measurements, TCP dynamically adjusts its retransmission timeout (RTO). If the delay suddenly increases due to a route change, the sender might not receive acknowledgements within the expected RTO, triggering unnecessary retransmissions. However, RFC6298 recommends (“SHOULD”) a minimum RTO of 1 s [10]. In our evaluation, the delays and deltas were never close to this bound, so we ignored retransmits and the corresponding in-flight window reductions. The RFC also suggests that this minimum value is up to further research—hence we plan to reinvestigate this in future work.

TCP Data Rate: The average data rate is a tangible value allowing for a performance-oriented comparison of different route selection algorithms. In general, we want to maximize this quantity that is derived from the route delays and validities using our abstract TCP implementation (Section IV-A).

III. ROUTE SELECTION ALGORITHMS

This section introduces a family of route selection algorithms that cover the full spectrum between minimizing delay and minimizing the number of route changes.

Each algorithm takes as input the position of two points on Earth (ground stations) some time interval $[t_0, t_h)$ (typically one orbital period), and a time granularity (typically 1 s). The output is a mapping of each time point in the interval to a specific end-to-end route connecting the two ground stations via the constellation.

Dijkstra (DIJKSTRA): Dijkstra’s algorithm is the state-of-the-art solution. It optimizes for the shortest route at a given time instant. In the following, we denote a route as shortest if it is minimal with respect to one-way delay, i.e. the “fastest” route. (By setting $d_t = d_q = 0 \text{ ms}$, the classical shortest route w.r.t. distance results.) As a baseline, our algorithm DIJKSTRA runs Dijkstra’s algorithm for each time point in $[t_0, t_h)$. While this minimizes the overall delay all along the interval, it has no concept of route stability.

Stubborn Dijkstra (STUBBORN): As a variation of DIJKSTRA, we introduce and analyse a “stubborn” variant of it. Here, the current shortest route is selected at t_0 . This route is then used for as long as it is valid, i.e. until one of the two access satellites

loses visibility to its respective ground station. Then, a new shortest route is computed, and this procedure is repeated until the end of the time interval.

Tenacious Routing (TENACIOUS): Since visibility periods are decisive for route validity, another alternative for maximizing route validity is to identify, for both ground stations, the satellite with the longest remaining visibility period. This is done at t_0 , and the resulting pair is then used as access satellites for a route that remains unchanged across the entire period in which both satellites have visibility. Once the current route becomes invalid, two new satellites are selected with the now longest remaining visibility time and the procedure is repeated.

This algorithm yields a list of access satellite pairs together with the time interval during which they are to be used. In a second step, a route is computed per entry. Assume two access satellites and an interval $[t_s, t_e)$. As the route length (and hence incurred delay) changes over time, there is no obvious optimal in-orbit route for the whole time interval. One possibility would be to compute the route with the lowest average delay during $[t_s, t_e)$. However, as this approach is compute-intensive, we just compute the shortest route at some point within $[t_s, t_e)$. There is a trade-off between minimizing average route delay (typically best if we pick a point in the middle) and having a low delay at t_e to increase the chances of avoiding reordering events when switching to another route. Our extensive empirical studies have shown that—compared to the risk of timeouts if changing to a fast route—reordering is clearly the crucial obstruction to performance, and that $t_s + \frac{3}{4}(t_e - t_s)$ provides overall good performance.

While experimenting with this algorithm, we noticed exorbitantly high delays in some cases. This is due to the algorithm not distinguishing between ascending and descending satellites. This movement direction, however, has a considerable effect on the route length. Assume, for example, two ground stations close to each other on the Equator. Now it may be that these stations can be connected by two ascending satellites with just one intermediate ISL hop. On the contrary, if one of the satellites is exchanged for a nearby descending satellite, these satellites can no longer communicate directly with each other, and multiple hops are required. Therefore, instead of selecting only two access satellites with the longest remaining visibility, we select at each station the ascending and the descending satellite with the longest remaining visibility (if it exists). In the second step, we consider all combinations (at most four) to select the pair with the lowest delay. This optimization significantly reduces the overall delay at the cost of a few more route changes.

(Weighted) Set Cover (SETCOVER): The previous algorithms are all greedy at specific time points, in one way or another. Now, we introduce a solution that considers the entire time interval $[t_0, t_h)$ to find some global optimum. For this, we have formulated the route selection problem as a (weighted) set cover problem. First, all satellite visibility intervals at both ground stations are computed. This information is used to compute all possible combinations of valid access satellite pairs, together with the maximal intervals $[t_s, t_e)$ during which

this pair can be used. The goal is now to select as few of these combinations as possible so that the whole time interval is covered. We solve this problem by modelling it as an integer linear program (ILP). For each combination of access satellite pair (src, dst) and interval $[t_s, t_e)$ during which it is valid, we introduce a binary variable $x_{[t_s, t_e)}^{src, dst} \in \{0, 1\}$. A value of 1 means that this element is part of the overall solution. Let I be the set of all such variables. The objective function $\min \sum_{x \in I} w_x \cdot x$ now minimizes the weighted sum of variables, where w_x is some non-negative weight for x . In this paper, our algorithm minimizes the total number of route changes, i.e. $w_x = 1$ for all $x \in I$. It is, however, straightforward to tweak the optimization objective by assigning different weights, such as the hop count in between the two satellites.

Let $I|_t = \{x_{[t_s, t_e)}^{src, dst} \in I \mid t_s \leq t < t_e\}$ be the subset of I that contains all binary decision variables whose interval includes time point t . Now, we add constraints to ensure that every time point in the time interval is covered by (at least) one element:

$$\forall t \in \{t_0, t_1, \dots, t_h\}. \sum_{x \in I|_t} x \geq 1$$

A solution of this ILP specifies which access satellites are best to use and at what intervals they should be used. However, the resulting intervals usually have a small overlap. To get disjoint intervals, we cut each interval in the middle of the overlapping part. More precisely, two overlapping intervals $[t_s, t_e)$ and $[t'_s, t'_e)$ are mapped to $[t_s, m)$ and $[m, t'_e)$ where $m = \frac{1}{2}(t_e + t'_s)$. Finally, we proceed the same way as described in the TENACIOUS algorithm to compute good routes given the access satellite pairs.

IV. EVALUATION AND RESULTS

In this section, we present our analysis results of an empirical performance evaluation using simulations. First, we introduce a framework for transport layer simulations, our tool ARTCP, which is used to evaluate the performance of the algorithms with respect to the transport layer. We then report some detailed findings for an exemplary pair of ground stations using the Starlink constellation. Finally, we provide some aggregated results obtained by placing ground stations on all possible locations on a grid, followed by an analysis of the computational costs of the algorithms. All benchmarks were run on a Linux machine equipped with an Intel Core™ i7-6700 CPU running at 3.40 GHz and 32 GB of main memory. Our toolchain is implemented in the Rust programming language, and `lp_solve` is used to solve the ILP problems.

A. Transport Layer Performance Evaluation Framework

To study the transport-layer effects of routing decisions, we have developed ARTCP, an abstract simulation of TCP variants. Comparing figures of delay and route validity times is a non-obvious task, as they impact the communication differently. Instead of doing multivariate comparisons of these two parameters, we decided to use them as inputs for a transport layer simulation that works as follows: As input, it takes both time series of RTT and reordering events (i.e. when a route change led to subsequent packets overtaking preceding ones).

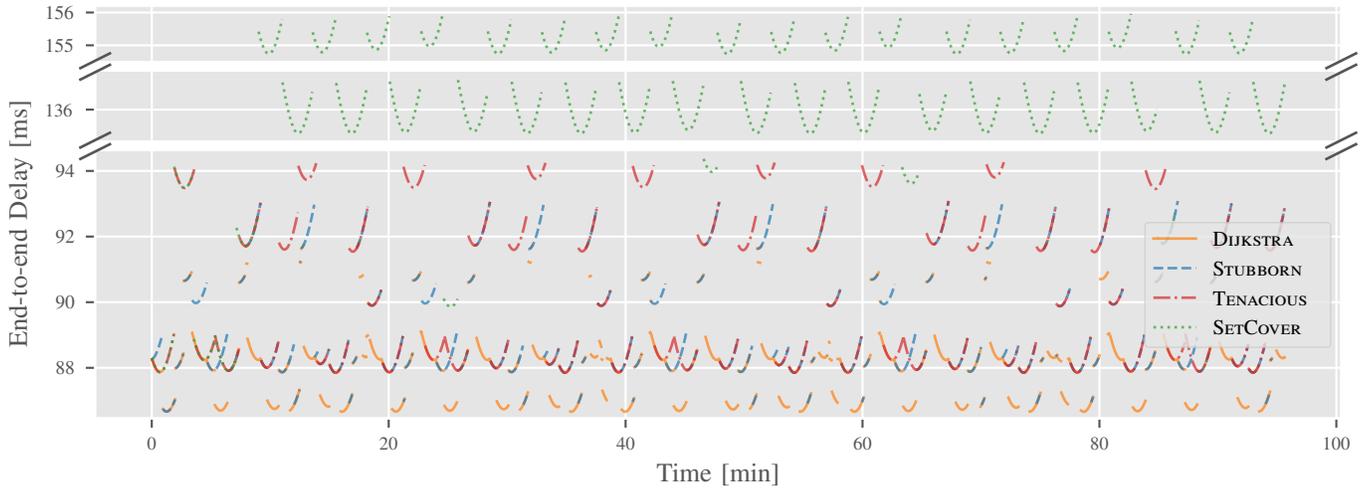


Figure 1. Comparison of end-to-end route delay and validity for the ground station pair Bariloche and Beijing on Starlink.

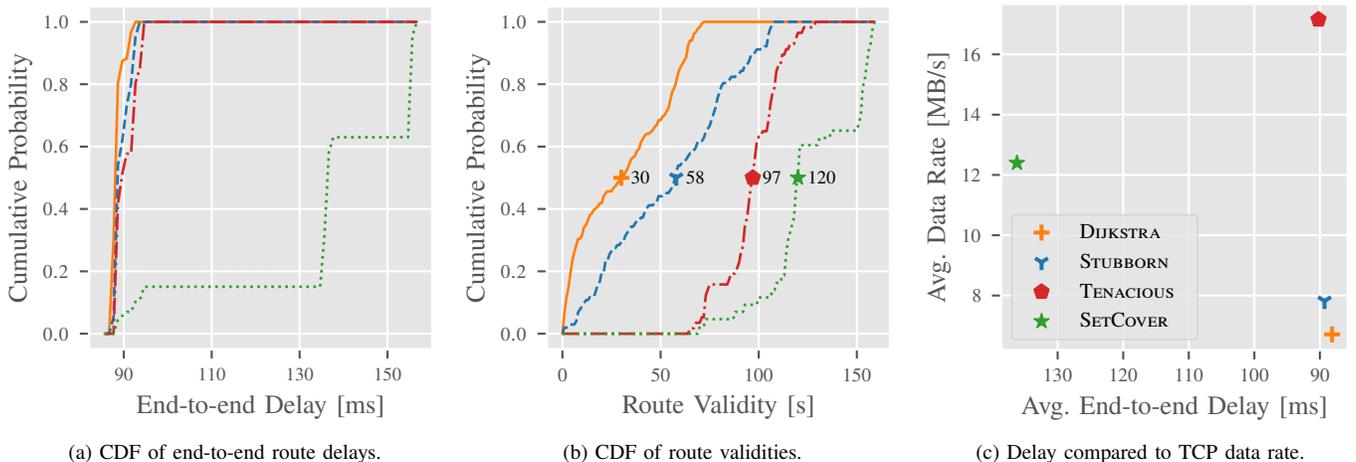


Figure 2. Cumulative statistics of the four algorithms on the ground station pair Bariloche and Beijing on Starlink.

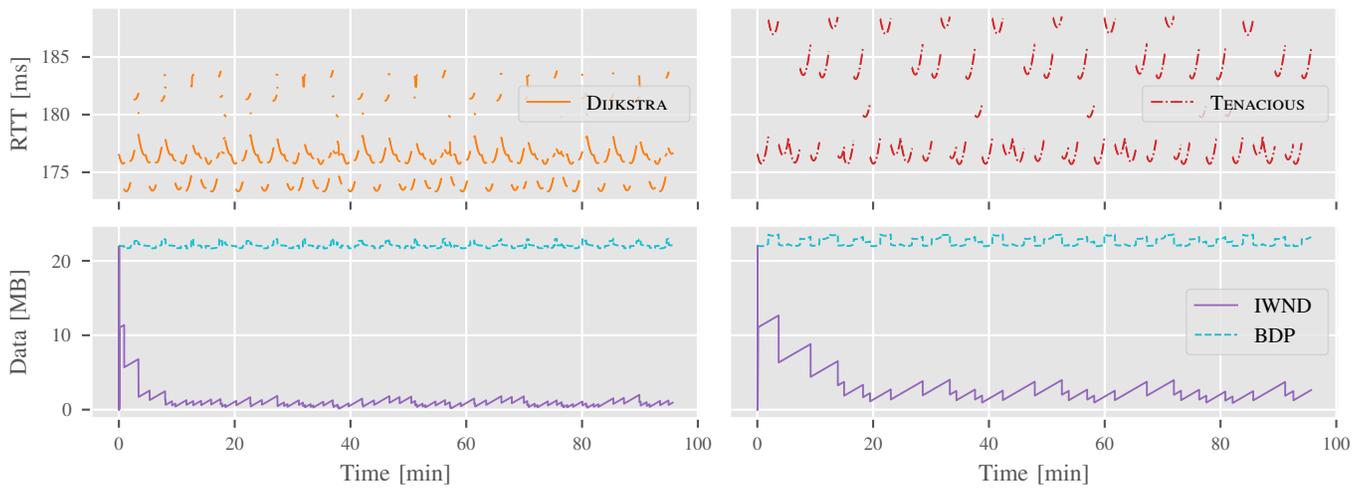


Figure 3. ARTCP results showing the round-trip time and in-flight window (IWND) for the algorithms DIJKSTRA and TENACIOUS.

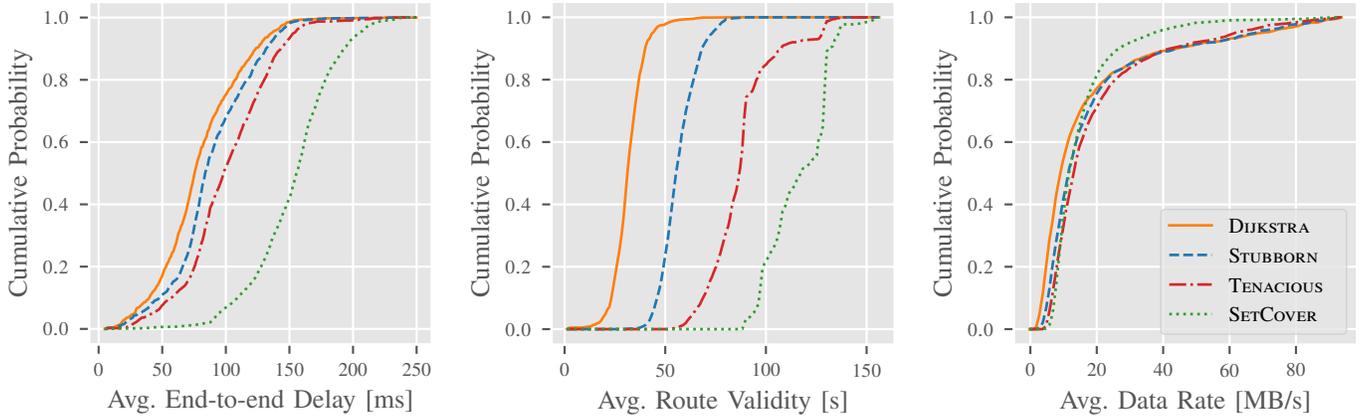


Figure 4. CDF plots over aggregated data for points on a latitude/longitude grid for Starlink.

During simulation, the evolution of an in-flight window is modelled, using the dynamics of well-known TCP variants (e.g. Reno). This means that we do not model individual packets via discrete event simulation, but rather assume ideal window progression under reordering events and RTT changes. Finally, the in-flight window over time is reported, together with aggregate metrics (e.g. average throughput).

In ARTCP, we assume a) the sender always has packets to send, i.e. infinite maximum data rate (in practice still limited by transmission algorithms) and b) the receiver can always accept packets, i.e. infinite receive buffer and application read rate (no flow control effects on in-flight window are considered).

B. Example: Bariloche and Beijing on Starlink

This example serves to demonstrate the different analyses we perform. For this benchmark, we assume two ground stations in Bariloche, Argentina and Beijing, China. For the constellation, we selected the (first) orbital shell of the initial deployment phase of SpaceX Starlink. The reason is that Starlink is well known and one of the few already operational Walker Delta constellations. Based on the publicly available information [11], [12], we model Starlink as a Walker Delta $53^\circ: 1584/72/39$ at 550 km. We consider satellites with a minimum elevation angle of 40° to be visible from the ground station. The great-circle distance between Bariloche and Beijing is 19 350 km and the shortest route has an average ISL hop count of 11. Specifics aside, the results shown here are representative for the general behaviour of the algorithms, regardless of whether the ground stations are a) on the same or different Earth hemispheres, and b) rather close or rather far apart.

Figure 1 shows the dynamics of end-to-end route delays over one orbital period for the four algorithms. It can be seen that DIJKSTRA is indeed the algorithm with the lowest delays while the other algorithms have higher delays, SETCOVER almost twice as high as DIJKSTRA. On the other side, DIJKSTRA induces 185 (101) route changes, STUBBORN 103 (61), TENACIOUS 58 (29), and SETCOVER just 44 (21), where the number in parentheses indicates the number of “bad” route changes, i.e. changes

from high to low delay. This illustrates the trade-off between minimizing delay and maximizing route validity.

The trade-off can be seen more clearly by looking at the two cumulative distribution function (CDF) plots showing the delay (Figure 2a) and validity (Figure 2b). The median delay of SETCOVER is 136 ms while it is just below 89 ms for the other three algorithms. In terms of route validity, SETCOVER is clearly superior to the DIJKSTRA or STUBBORN approaches.

Finally, we ran the resulting routes through ARTCP. The average TCP data rate for all algorithms is shown in Figure 2c. The plot shows that DIJKSTRA and STUBBORN have the lowest performance. The average data rate for TENACIOUS is more than doubled, while still being competitive in terms of average delay. Even that SETCOVER minimizes the number of route changes, we observe that the ARTCP performance falls behind that of the TENACIOUS algorithm. The reason is that the higher delay for SETCOVER prevents a good performance.

The full ARTCP comparison between DIJKSTRA and TENACIOUS is in Figure 3, as these provide the worst and best performance. The plot shows that both algorithms feature rather poor performance in general, as neither of them gets close to the bandwidth-delay product (BDP). The two key takeaways are: a) not just the number of route changes is important but rather how many of them are from high to low delay (i.e. cause reordering events), and b) other, optimized TCP congestion control algorithms (e.g. CUBIC) should be explored.

C. Aggregated Grid Analysis

The previous evaluation results were all specific to the pair of ground stations at Bariloche and Beijing. Now, we explore how the different algorithms perform when other ground stations are chosen. For this, we first create a grid covering the entire globe between $-\alpha$ and α degrees latitude, where α is the inclination of the constellation, e.g. $\alpha = 53^\circ$ for Starlink. In this case, we use a grid granularity of 10° for longitude and $\frac{53^\circ}{5} = 10.6^\circ$ for latitude. Initially, one ground station is fixed at 0° E, 0° N. For the other ground station, every possible point on the grid is considered, and all algorithms are executed on that pair. Then,

Table I
STATISTICS OF RUN TIMES (IN SECONDS) TO EXECUTE THE ALGORITHMS ON A
GROUND STATION PAIR FOR ONE ORBITAL PERIOD.

Algorithm	Min	Q1	Median	Q3	Max
DIJKSTRA	6.637	9.087	12.537	18.684	92.503
STUBBORN	2.104	2.186	2.243	2.351	4.829
TENACIOUS	2.060	2.110	2.118	2.127	2.167
SETCOVER	2.310	2.415	2.566	2.833	4.196

the first ground station is relocated to 0° E, 10.6° N and again all grid points are considered for the second ground station. This whole process is repeated until the first ground station is at 0° E, 53° N. This considers all relevant combinations of two ground station positions. Due to the symmetries of a Walker Delta constellation, it is not necessary to consider negative latitudes for the first ground station nor different longitudes than 0° E.

In each run, the average end-to-end delay and average route validity is stored. Each trace is also run through ARTCP to get the average data rate. Figure 4 shows the respective CDF plots.

To evaluate the effect of the constellation topology on the results, we repeated the same analysis on two other Walker Delta constellations (60°: 779/41/5 at 500 km and 60°: 399/21/5 at 1000 km, both with a minimum elevation angle of 25° and a grid granularity of 10° in each direction). While the individual figures changed, the overall picture remained the same.

D. Run Time / Computational Complexity

As a final metric, we evaluate the computational cost of the four algorithms. For every ground station pair of the Starlink grid analysis, we measured the run time of each algorithm. The results are shown in Table I. The first observation is that STUBBORN, TENACIOUS, and SETCOVER typically require about two seconds for each run while the run time of DIJKSTRA is significantly higher. However, this comparison is biased. Since DIJKSTRA should minimize the overall delay, it needs to perform a shortest-route computation every second. This time granularity is a factor chosen by us that punishes the performance of DIJKSTRA. The other algorithms only sporadically have to compute shortest routes because they stick to a route longer and, in the case of STUBBORN and TENACIOUS, only need to check whether the current route is still valid, i.e. whether the two access satellites are still visible from the ground. Since the median run time of DIJKSTRA is six times higher than the other algorithms, we could increase the time granularity to six seconds to obtain a median run time that can compete with the other algorithms. On the other side, this would increase the overall delay of DIJKSTRA slightly.

V. CONCLUSION

This paper has introduced dedicated route selection algorithms for LEO satellite networks that aim to significantly increase overall network performance while being fully compatible with standard transport layer management.

We have presented empirical studies that highlight the potential of routing algorithms optimized for stability, as

opposed to the state of the art, which primarily optimizes for minimum delays. Our evaluation results are encouraging but suggest that in addition to specialized route selection algorithms, specialized congestion control algorithms (CCAs) may also be necessary. We are currently analysing other TCP CCAs (e.g. CUBIC and BBR) and are using these findings to design a route selection algorithm that maximizes network performance. Using our simulator ARTCP, we are making first steps in co-designing route selection and transport control approaches for in-orbit communication. This can be combined with performance-enhancing proxies and related approaches (e.g. Yuan et al. [13]) to further improve connection stability.

This work has so far focused on Walker Delta constellations such as Starlink, but we will extend this work to other types of constellations. Further investigation will include packet-level simulation and/or real-world measurements to prove the validity of the results in less idealized environments.

ACKNOWLEDGEMENTS

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 101008233 – MISSION, see <https://mission-project.eu>, and by DFG grant 389792660 as part of TRR 248 – CPEC, see <https://perspicuous-computing.science>.

REFERENCES

- [1] R. Govindan and A. Reddy, “An analysis of internet inter-domain topology and route stability,” in *IEEE INFOCOM '97*, IEEE, 1997, pp. 850–857. doi: 10.1109/INFCOM.1997.644557.
- [2] L. Gao and J. Rexford, “Stable internet routing without global coordination,” *IEEE/ACM Trans. Netw.*, vol. 9, no. 6, pp. 681–692, 2001. doi: 10.1109/90.974523.
- [3] M. A. Canbaz, K. Bakhshaliyev and M. H. Gunes, “Analysis of path stability within autonomous systems,” in *IEEE M&N 2017*, IEEE, 2017. doi: 10.1109/IWMN.2017.8078364.
- [4] M. Iodice, M. Candela and G. D. Battista, “Periodic path changes in RIPE atlas,” *IEEE Access*, vol. 7, pp. 65 518–65 526, 2019. doi: 10.1109/ACCESS.2019.2917804.
- [5] K. N. Srijith, L. Jacob and A. L. Ananda, “TCP Vegas-A: Solving the fairness and rerouting issues of TCP vegas,” in *IPCCC 2003*, IEEE, 2003, pp. 309–316. doi: 10.1109/PCCC.2003.1203713.
- [6] F. Y. Yan, J. Ma, G. D. Hill *et al.*, “Pantheon: The training ground for internet congestion-control research,” in *USENIX ATC 2018*, USENIX Association, 2018, pp. 731–743.
- [7] Y. Hauri, D. Bhattacharjee, M. Grossmann and A. Singla, “‘Internet from space’ without inter-satellite links,” in *HotNets '20*, ACM, 2020, pp. 205–211. doi: 10.1145/3422604.3425938.
- [8] D. Vasisht and R. Chandra, “A distributed and hybrid ground station network for low earth orbit satellites,” in *HotNets '20*, ACM, 2020, pp. 190–196. doi: 10.1145/3422604.3425926.
- [9] V. Bhosale, A. Saeed, K. Bhardwaj and A. Gavrilovska, “A characterization of route variability in LEO satellite networks,” in *PAM 2023*, Springer, 2023, pp. 313–342. doi: 10.1007/978-3-031-28486-1_14.
- [10] M. Sargent, J. Chu, D. V. Paxson and M. Allman, *Computing TCP’s retransmission timer*, RFC 6298, 2011. doi: 10.17487/RFC6298.
- [11] G. Stock, J. A. Fraire and H. Hermanns, “Distributed on-demand routing for LEO mega-constellations: A starlink case study,” in *ASMS/SPSC 2022*, IEEE, 2022. doi: 10.1109/ASMS/SPSC55670.2022.9914716.
- [12] Space Exploration Holdings, LLC, *SpaceX non-geostationary satellite system: Attachment A*, FCC IBFS SAT-MOD-20190830-00087, 2019.
- [13] G. Yuan, D. K. Zhang, M. Sotoudeh, M. Welzl and K. Winstein, “Sidecar: In-network performance enhancements in the age of paranoid transport protocols,” in *HotNets 2022*, ACM, 2022, pp. 221–227. doi: 10.1145/3563766.3564113.