Routing Algorithms for VLEO Constellations

Supervisors

Prof. Roberto GARELLO

Prof. Juan Andres FRAIRE

Dr. Alessandro COMPAGNONI

Candidate

Nicolò BENSO

Abstract

Very Low Earth Orbit (VLEO) satellite constellations are gaining attention for enabling low-latency, high-throughput non-terrestrial networks and high-resolution Earth observation. Operating at 300-500 km, they offer reduced propagation delays and stronger signals but also face challenges such as increased atmospheric drag, limited satellite lifespans, high Doppler shifts, and fast-changing topologies that complicate routing. Efficient routing over Inter-Satellite Links (ISLs) is critical in these dynamic environments. DisCoRoute, a recently proposed heuristic, demonstrates strong performance under such conditions. This work builds on it, with three primary objectives. The first is to assess its performance under time-varying conditions. Starting from an existing MATLAB implementation, the code is adapted to reflect the dynamic positions of satellites over time. The analysis shows that the latency between a fixed source and destination varies over time due to orbital geometry or route changes. Additionally, the frequency of path changes is evaluated across different sampling intervals to identify the time step that ensures route stability. The second objective focuses on adapting DisCoRoute to a revised satellite topology in which each node has three communication terminals: two for intraplane links and one for interplane links. This deviates from the original four-terminal setup, which includes two interplane links. Several DisCoRoute variants are developed for this three-terminal configuration, and the most effective is selected and named DisCo3T. Its performance is benchmarked against a modified Dijkstra algorithm adapted to the same network structure. To further improve routing efficiency, a novel method is devised to compute the minimum number of ISL hops between any two nodes in the new topology. A new mathematical model, called Interleaved Scenario - MinHopCount (IS-MHC), is formulated and implemented, allowing direct comparison with Dijkstra-based results under evenly distributed planes and satellites. The method consistently matches the theoretical minimum hop counts obtained through benchmarking. In the final phase of the work, the optimized hop-count method is integrated into an enhanced version of the DisCo3T algorithm. While preserving the original structure, this new version leverages the refined hop-count calculation to approximate optimal path selection. The resulting routing method outperforms traditional shortest-path algorithms such as Dijkstra in terms of computational efficiency, requiring only 3.56% of the execution time needed by Dijkstra-based approaches, while maintaining near-optimal latency performance with a gap of just 2.5%. The third and final objective concerns the formulation of a novel, modified hop-count model specifically tailored to Walker-Star constellation configurations, called Walker-Star-MinHopCount (WS-MHC). Upon integrating the WS-MHC model into the DisCoRoute routing algorithm, the enhanced algorithm once again demonstrates near-optimal performance when compared to the Dijkstra benchmark, with an average latency gap of only 1.20 %, and maintaining its computational efficiency, requiring just 0.75 % of the computational time needed by Dijkstra-based methods. The results obtained position the newly developed algorithms as strong candidates for practical deployment in large-scale VLEO satellite networks.

Acknowledgements

I would like to express my sincere gratitude to my supervisors, Prof. Roberto Garello, Prof. Juan Fraire, and Dr. Alessandro Compagnoni, for their guidance throughout these six months and for the trust they placed in my abilities. Their support has made this thesis not only an opportunity for professional growth but also a deeply enriching personal experience.
I also wish to thank all the members of the AstraCOM Group for their valuable advice and for the way they welcomed me into a team of established professionals.

A heartfelt thank you goes to my family, especially my parents and grandparents, for their love and support throughout all these years of study, through the highs and lows, both inside and outside the university.

To all my friends, those I see often and those I speak with less frequently, those I've known for twenty years and those I've just met, THANK YOU. Thank you for the experiences, the laughter, the arguments, and all the emotions we've shared. You have made these years even more memorable and have played a part in shaping the person I am today. May this be just the beginning.
Per Aspera Ad Astra

Table of Contents

List of Tables

List of Figures

Acronyms

3T Three-Terminal.

4T Four-Terminal.

A2A Ascending-to-Ascending.

A2D Ascending-to-Descending.

D2A Descending-to-Ascending.

D2D Descending-to-Descending.

DisCo3T Distributed On-Demand Routing for Mega-Constellations for 3T Topologies.

DisCoRoute Distributed On-Demand Routing for Mega-Constellations.

E2E End-to-End.

ECEF Earth-Centered, Earth-Fixed.

ESS Expand Search Space.

IS-MHC Interleaved-Scenario MinHopCount.

ISL Inter-Satellite Link.

LEO Low Earth Orbit.

MinHopCount Minimum Hop Count.

NTN Non-Terrestrial Network.

RAAN Right Ascension of the Ascending Node.

VLEO Very Low Earth Orbit.

WS-MHC Walker-Star-MinHopCount.

Chapter 1

Introduction

The growing interest in Very Low Earth Orbit (VLEO) satellite constellations is driven by their potential to deliver high-throughput, low-latency, and resilient non-terrestrial network (NTN) connectivity, as well as to support high-resolution Earth observation missions.

Operating at altitudes between 300 and 500 km, VLEO satellites offer significantly reduced propagation delays and improved link budgets. However, they also pose technical challenges, including stronger atmospheric drag, shorter satellite lifespans, increased Doppler shifts, reduced visibility durations, and the need for denser, more dynamic constellations to guarantee continuous global coverage [1, 2].

Routing in VLEO constellations is particularly challenging due to the rapid satellite motion, which causes frequent and unpredictable topology changes, complicating the design of reliable and timely data delivery mechanisms. To address this, the heuristic routing algorithm DisCoRoute [3] has recently been proposed and demonstrated strong performance under dynamic VLEO conditions [4]. It efficiently selects routes by leveraging two key geometric properties of Walker-Delta constellations: (i) intra-plane hops are constant and time-invariant in length, and (ii) inter-plane hops tend to be shorter near the poles and longer near the Equator. DisCoRoute was originally developed for satellites equipped with four terminals, a configuration feasible for many LEO/VLEO constellations.

However, the number of available terminals per satellite plays a critical role in determining overall network performance, cost, and architectural complexity [5]. A promising alternative for future LEO/VLEO systems is the use of a three-terminal (3T) architecture, currently adopted in commercial constellations like Starlink [6]. In this design, two terminals support along-track (intra-plane) communication, while a third hemispherical terminal dynamically switches between left and right cross-plane neighbors, but not both simultaneously [7].

A key limitation of the 3T configuration is the reduced cross-plane connectivity, which imposes topological constraints that traditional routing algorithms, designed for fully-connected or stable mesh topologies, may not accommodate efficiently. To date, the only known work that addresses routing under the 3-ISL topology is [8], which presents a theoretical comparison with the 4-terminal (4T) case, analyzing metrics such as hop count, delay, and capacity. However, a comprehensive evaluation of in-space routing algorithms in terms of latency and computational cost under 3T constraints remains an open research question.

1.1 Contributions

This thesis, developed within the framework of the European Space Agency (ESA) project **Hand**over, Data Rout**ing** and Radio Resource Management f**or** **Ver**y Low Earth Orbit (VLEO) Broadband Constellations (**HANDING-OVER**) [9], addresses these challenges by extending and adapting state-of-the-art routing algorithms to operate effectively in 3T VLEO constellations. Specifically:

- We propose a novel analytical model to accurately compute the minimum hop count under sparse 3T topologies, ensuring guaranteed reachability.

- We introduce a modified version of the DisCoRoute heuristic tailored to support interleaved topologies under 3T constraints.

- We propose a new minimum hop-count formulation for Walker-Star constellations under 4T, accounting for their distinct Right Ascension of the Ascending Node (RAAN) and inclination configuration.

- We incorporate a time-varying topology analysis to evaluate algorithm performance under realistic orbital dynamics, capturing the impact of satellite mobility on connectivity and routing stability.

- We perform extensive simulations with realistic VLEO parameters, evaluating key performance indicators including latency, hop count, and algorithmic complexity.

Thesis Organization. The remainder of this thesis is structured as follows: Section 1.2 introduces the system and network topology models. Chapter 2 reviews the state-of-the-art algorithms employed in this work. Chapter 3 presents the proposed models and algorithms for the different tasks. In particular, Section 3.3.1 introduces the analytical hop-count model developed for 3T scenarios, while Section 3.4 details the adaptation of the DisCoRoute heuristic. Evaluation results are presented and analyzed in Chapter 4, and final conclusions are drawn in Chapter 5.

1.2 System Model

This section introduces the adopted orbital configuration, based on the Walker-Delta constellation model.

1.2.1 Walker-Delta Constellations

Walker-Delta constellations, also known as *Ballard-Rosette* configurations, are characterized by circular orbits arranged in a flower-like pattern and evenly spaced along the Equator. These constellations are described using the notation $\alpha\colon T/P/F$, where:

- α denotes the inclination of each orbital plane relative to the Earth's equatorial plane;
- T is the total number of satellites;
- P is the number of orbital planes;
- F is the phasing factor, which defines the relative angular displacement of satellites in adjacent planes.

All orbits in a Walker-Delta configuration have the same altitude, inclination, and are circular. Each orbital plane contains $Q = T/P$ satellites, uniformly distributed along the orbit.

Each satellite within the constellation is defined by [3]:

- Its *longitude of the ascending node*, denoted as Ω. However, in this thesis, we refer to the *initial longitude of the ascending node*, represented as $L_0 \in [-\pi, \pi)$, which is a time-invariant constant specific to each satellite. The actual (i.e., time-dependent) longitude of the ascending node Ω at a given time instant t is computed as: $\Omega = L_0 - \omega_E \cdot t$, where ω_E denotes the angular velocity of the Earth's rotation.
- The *true anomaly*, denoted as ν, which in general defines the angle between the direction of the periapsis and the current position of the satellite along its orbit. However, in this case, since the orbits are circular (i.e., eccentricity $e = 0$), the periapsis is not uniquely defined, rendering the true anomaly ν undefined and thus unsuitable for determining the satellite's position.
- The *argument of latitude* (also referred to as the phase angle), denoted as $u \in [-\pi, \pi)$, is used in place of ν to represent the angular position of the satellite within its orbit. It measures the angle between the ascending node and the satellite's position along the orbital path. A satellite is said to be

in an *ascending* phase, meaning it is moving in the north-east direction, if $u \in [-\frac{\pi}{2}, \frac{\pi}{2}]$. Conversely, it is considered *descending*, that is, moving towards the south-east, if u lies outside this range.

- The satellite's *altitude*, denoted as h, which is defined as its vertical distance relative to the surface of the Earth.

The angular separation in right ascension between adjacent planes, referred to as the *RAAN (Right Ascension of the Ascending Node) difference*, is given by:

$$\Delta\Omega = \frac{2\pi}{P} \in [0, 2\pi) \tag{1.1}$$

This value determines how far neighboring planes are spaced in longitude and depends solely on the number of planes P.

Within each plane, satellites are separated in argument of latitude. The angular separation between adjacent satellites in the same plane, known as the *phase difference*, is:

$$\Delta\Phi = \frac{2\pi}{Q} \in [0, 2\pi) \tag{1.2}$$

This value is a function of the number of satellites Q per plane.

Between adjacent planes, the relative angular shift between satellites is defined by the *phase offset*:

$$\Delta f = \frac{2\pi F}{PQ} \in [0, 2\pi) \tag{1.3}$$

This offset specifies the difference in argument of latitude between two horizontally neighboring satellites, and it depends on the phasing factor $F \in \{0, 1, \ldots, P-1\}$, which controls the angular offset between satellites in adjacent planes. When $F = 0$, satellites in different planes are aligned; for $F \neq 0$, satellites in successive planes are progressively shifted forward or backward along the orbit.

Unlike arbitrary phase offsets, the structured form of Δf in the Walker Delta model guarantees that the product $P \cdot \Delta f$ is always a multiple of $\Delta\Phi$. This ensures that all satellites experience the same angular offset across planes, a property essential for constellation symmetry and regularity.

Each satellite is uniquely identified by a pair of indices (o, i), where $o \in \{0, \ldots, P-1\}$ indicates the orbital plane, and $i \in \{0, \ldots, Q-1\}$ denotes the satellite's position within that plane. Alternatively, satellites can be indexed using a global ID in the range 1 to T.

In terms of connectivity, each satellite typically maintains four ISLs:

- Two *intra-plane* links, connecting to the previous and next satellites in the same orbit: $(o, (i-1) \bmod Q)$ and $(o, (i+1) \bmod Q)$;

- Two *inter-plane* links, connecting to satellites in adjacent planes. The left neighbor is $(o{-}1, i)$ if $o > 0$, or $(P{-}1, (i{-}F) \bmod Q)$ if $o = 0$. The right neighbor is $(o{+}1, i)$ if $o < P{-}1$, or $(0, (i{+}F) \bmod Q)$ if $o = P{-}1$.

In the literature, links between satellites in the same orbital plane are referred to as *vertical hops* or *intra-plane links*, while links between satellites in adjacent planes are referred to as *horizontal hops* or *inter-plane links*.

1.2.2 ISL Terminal Configurations

This thesis considers two distinct configurations of ISL terminals:

- **4T**: The original DisCoRoute algorithm [3] was designed for a 4T satellite configuration. In this architecture, each satellite is equipped with four communication terminals: two are dedicated to intra-plane communication, while the remaining two hemispherical terminals enable simultaneous inter-plane communication with both the left and right cross-plane neighbors. In this configuration, each satellite maintains simultaneous communication with all four immediate neighbors: two within the same orbital plane and two in adjacent planes.

- **3T**: This thesis mainly focuses on adapting the DisCoRoute algorithm to operate within a 3T architecture. In this setting, two terminals are dedicated to along-track (intra-plane) communication, while a third, hemispherical terminal enables switching to either the left or right cross-plane neighbor, but not both simultaneously.

  In this context, selecting an ISL topology that ensures full network reachability is of paramount importance. Accordingly, we adopt an interleaved topology characterized as follows:

  Given a satellite with an even index $i = 2m$, if it is located on an even-numbered orbital plane $o = 2k$, it is connected to its right-hand neighbor in the adjacent plane. Conversely, if it lies on an odd-numbered plane $o = 2k + 1$, it is connected to its left-hand neighbor. For satellites with an odd index $i = 2m + 1$, the connection direction is reversed: satellites on even-numbered planes connect to the left, and those on odd-numbered planes to the right, where $k, m \in \mathbb{N}$. In this configuration, each satellite maintains simultaneous communication with only three immediate neighbors: two within the same orbital plane and one in adjacent planes. A representative segment of this final topology is depicted in Figure 1.1.

As one can deduce, the 3T topology presents greater challenges than the 4T configuration when it comes to routing algorithms. While Dijkstra's algorithm can

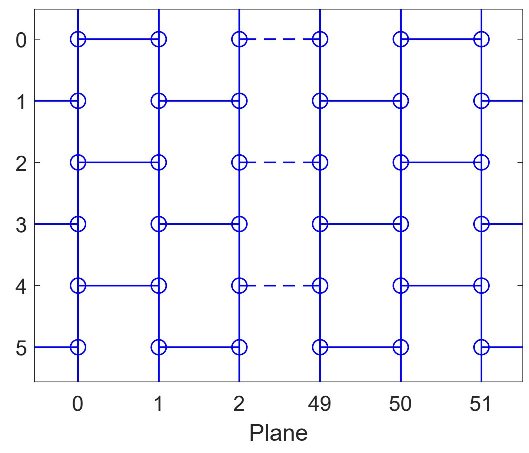


Figure 1.1: 3 ISL topology of 65°: 2080/52/10 with interleaved inter-plane links.

be relatively easily adapted as discussed in Section 2.1, other algorithms such as DisCoRoute encounter more significant difficulties due to their structural reliance on persistent cross-links, which are limited or unavailable in a 3T scenario.

Nevertheless, given the promising results obtained in [3], we argue that adapting DisCoRoute to operate under 3T constraints is a crucial step toward enabling future satellite communication systems that require low-latency routing while maintaining low execution time. The algorithm has demonstrated both low computational complexity and near-optimal performance when compared to Dijkstra's algorithm, making its extension to the 3T case both relevant and impactful.

Chapter 2

State of Art

In this chapter, we present the two main algorithms employed in this thesis: Dijkstra's algorithm and the DisCoRoute algorithm. The former serves as a baseline for comparison, while the latter provides the foundation upon which we develop the novel routing algorithm tailored for the 3T scenario.

2.1 Dijkstra

Dijkstra's algorithm is a classical and widely adopted method for solving shortest path problems and is used here as a baseline. The algorithm solves the single-pair shortest path problem by modeling the satellite constellation as a graph and using the Euclidean distance between satellites as the edge cost. A min-heap is used to store unvisited nodes, prioritizing those with the smallest tentative distance.

The pseudocode is shown in Algorithm 1. To compute the shortest path from the source to all nodes, Line 12 to Line 14 can be omitted; however, when only the path to a specific destination is needed, these lines provide an early-termination condition once the destination node is reached, saving computational time.

Upon completion, the shortest path length from source to destination is stored in `dist[dst]`. Since the full path is typically of interest, the algorithm also records, for each visited node, its predecessor along the shortest path. The route reconstruction begins at the destination and follows the `prev` pointers backward until the source is reached, yielding the complete path. We employed two versions of the algorithm:

Dijkstra-Distance: It finds the shortest path in terms of shortest distance, i.e., minimizing the overall route propagation delay. The metric used is the Euclidean distance between satellites. In the remainder of this work, we shall refer to this version interchangeably as *Dijkstra-Distance*, or more concisely, simply as *Dijkstra*.

Dijkstra-Hop: It finds one of the shortest path in terms of minimum number of hops to connect source and destination satellites, i.e., minimizing the overall number of traversed nodes in the route. The metric is a constant, 1, instead of the Euclidean distance, since we are interested in the number of hops. Throughout the following discussion, this variant will be referred to either as *Dijkstra-Hop* or as *DijkstraHop*.

The two algorithms share a similar structure, differing solely in the objective of the minimization: latency in the case of Dijkstra-Distance, and the number of hops for Dijkstra-Hop. As such, both variants can be represented by Algorithm 1, with the only distinction being the metric used in Line 17. The adaptation of these algorithms to the 3T topology is based on a common strategy, which consists in modifying the mechanism used to identify the neighboring nodes of the current satellite. In the original implementation, it was assumed by default that each node had exactly four neighbors and they were determined using the constellation indices (o, i); however, this assumption no longer holds under the 3T configuration and must therefore be explicitly reconsidered. Specifically, this is accomplished by determining the neighboring satellites based on the adjacency matrix of the inter-satellite links. The 3T adaptations of the two algorithms will henceforth be referred to as *Dijkstra3T*, for the Dijkstra-Distance algorithm, and *DijkstraHop3T*, for Dijkstra-Hop.

2.2 DisCoRoute

This section presents *DisCoRoute*, a lightweight, topology-aware routing algorithm designed for large-scale polar satellite constellations, such as those based on the Walker-Delta architecture. DisCoRoute is specifically tailored for 4T inter-satellite link (ISL) topologies and exploits the deterministic structure of the constellation to construct low-latency, minimum-hop routes between any source and destination pair. The key innovation lies in the use of a geometric *Minimum Hop Count* (MinHopCount) model, which analytically estimates the minimum number of horizontal and vertical hops required for a given path. By relying solely on the known constellation configuration, without requiring global link-state knowledge, DisCoRoute achieves high scalability and routing efficiency. Simulation results demonstrate that DisCoRoute achieves over a 98% reduction in computational time compared to Dijkstra-based algorithms, while maintaining nearly equivalent end-to-end latency, within a worst-case deviation of only -2%. These findings underscore the suitability of DisCoRoute for onboard execution in dynamic LEO and VLEO large constellations, where routing efficiency, scalability, and low computational overhead are essential.

Algorithm 1 Dijkstra's Shortest Path

```
procedure DIJKSTRA($Const., src, dst$)
    for all $satellites\ v \in Constellation$ do
        $dist[v] \leftarrow \infty$                                   ▷ unknown distance to $v$
        $prev[v] \leftarrow \perp$                                    ▷ predecessor of $v$
        $visited[v] \leftarrow false$
    end for
    $dist[src] \leftarrow 0$
    $Q \leftarrow$ HEAPIFY($\{(v, dist[v]) \mid v \in Const.\}$)        ▷ Initialize priority queue
    while $Q \neq \emptyset$ do
        $(u, d) \leftarrow Q$.POP( )                                  ▷ pop sat. u with minimal distance $d$
        $visited[u] \leftarrow true$
        if $u = dst$ then
            break                                                     ▷ shortest path found
        end if
        for all neighbors $v$ of $u$ do
            if $\neg\ visited[v]$ then
                $alt \leftarrow d +$ EUCLIDEAN($u, v$)
                if $alt < dist[v]$ then
                    $dist[v] \leftarrow alt$                          ▷ found shorter alternative
                    $prev[v] \leftarrow u$
                    $Q$.DECREASEKEY($v, alt$)                         ▷ update priority of sat. v
                end if
            end if
        end for
    end while
    return ($dist[\ ], prev[\ ]$)
end procedure
```

2.2.1 MinHopCount Model

To estimate the minimum number of inter-satellite link (ISL) hops between two satellites in a Walker Delta constellation, we adopt the hop-count model presented in [3]. The model computes both inter-plane and intra-plane hops required to connect any two satellites, accounting for the directional geometry of the constellation. Let ΔL_0 denote the angular difference between the initial longitude of the ascending node of the destination, denoted as $L_{0,2}$, and of the source, $L_{0,1}$:

$$\Delta L_0 = (L_{0,2} - L_{0,1}) \bmod 2\pi$$

From this, the number of inter-plane hops in east and west directions can be determined by dividing the angular separation by the RAAN spacing $\Delta\Omega$:

$$H_h^{\leftarrow} = \left\lfloor \frac{2\pi - \Delta L_0}{\Delta\Omega} \right\rceil, \qquad H_h^{\rightarrow} = \left\lfloor \frac{\Delta L_0}{\Delta\Omega} \right\rceil$$

The symbol $\lfloor \cdot \rceil$ denotes *commercial rounding*, where a given real number is rounded to the nearest integer, according to $\lfloor x \rceil = \texttt{sgn}(x) \lfloor |x| + 0.5 \rfloor$. Next, we compute the residual phase difference (argument of latitude) after accounting for horizontal hops. This is given by:

$$\Delta\vec{u} = (u_2 - u_1 - H_h^{\rightarrow} \cdot \Delta f) \bmod 2\pi, \qquad \Delta\overleftarrow{u} = (u_2 - u_1 + H_h^{\leftarrow} \cdot \Delta f) \bmod 2\pi$$

where u_1 and u_2 are the arguments of latitude (or phase angles) of the source and the destination respectively, with $u_n \in [-\pi, \pi)$ for $n = 1{,}2$. These residual angular differences are then used to compute vertical hop counts (intra-plane), depending on direction:

$$H_v^{\nearrow} = \left\lfloor \frac{\Delta\vec{u}}{\Delta\Phi} \right\rceil, \quad H_v^{\nwarrow} = \left\lfloor \frac{\Delta\overleftarrow{u}}{\Delta\Phi} \right\rceil, \quad H_v^{\searrow} = \left\lfloor \frac{2\pi - \Delta\vec{u}}{\Delta\Phi} \right\rceil, \quad H_v^{\swarrow} = \left\lfloor \frac{2\pi - \Delta\overleftarrow{u}}{\Delta\Phi} \right\rceil$$

Finally, the minimum hop count is the smallest among the four directionally-combined options:

$$\min \left\{ H_h^{\rightarrow} + H_v^{\nearrow},\, H_h^{\rightarrow} + H_v^{\searrow},\, H_h^{\leftarrow} + H_v^{\nwarrow},\, H_h^{\leftarrow} + H_v^{\swarrow} \right\}$$

This formulation ensures that both inter-plane and intra-plane contributions are minimized jointly, and also yields directional indicators for use in path construction. It advances upon previous approaches, such as the one proposed by Chen et al. [10], by eliminating the assumption that "*if the path in a given direction is too long (e.g., $H_h > P/2$), packets will be routed in the opposite direction*," which can lead to suboptimal routing decisions in specific constellation configurations.

2.2.2 Routing Algorithm

Based on the results obtained from the minimum hop count analysis, the algorithm produces near-optimal solutions while maintaining a significantly lower computational complexity compared to Dijkstra's algorithm. This property constitutes its principal advantage and underpins the fundamental motivation for our proposed methodology. DisCoRoute is built upon two key observations:

- The length of an intra-plane hop (i.e., a connection between satellites within the same orbital plane) remains constant throughout the constellation, regardless of the satellites' positions.
- The length of an inter-plane hop (i.e., a connection between satellites in adjacent orbital planes) decreases as the satellites approach the polar regions, moving away from the Equator.

The core idea is to distribute inter-plane hops so that they occur as close to the poles as possible. To achieve this, the algorithm first determines the minimum number of hops, along with the corresponding propagation directions, required to connect the source and destination satellites. Based on this information, the algorithm distinguishes between two distinct routing scenarios, depending on the respective flight directions of the source and destination satellites:

- **Ascending-to-Ascending/Descending-to-Descending (A2A/D2D)**: All inter-plane hops are placed either at the beginning or the end of the route, while all intra-plane hops are positioned in the middle. The algorithm selects the configuration that maximizes the aggregate distance of the inter-plane hops from the Equator. To construct the route, the algorithm operates bidirectionally, starting simultaneously from both the source and the destination satellites. At each iteration, it computes the absolute value of the sum of the latitudes of the potential inter-plane hop from the source side and the corresponding hop from the destination side. The hop associated with the *larger* value is incorporated into its respective route segment. This selection process is repeated until all required inter-plane hops have been allocated. Once completed, the two partial routes are joined by the necessary number of intra-plane hops, if any, to form the final path. The procedure is reported in Algorithm 2.
- **Ascending-to-Descending/Descending-to-Ascending (A2D/D2A)**: These types of routes must always pass near the poles, as this is the only region where links exist between ascending and descending nodes. Consequently, all intra-plane hops are placed either at the beginning or the end of the route, while all inter-plane hops are concentrated in the middle, opposite to the

Algorithm 2 DisCoRoute (Cases A2A & D2D)

```
procedure DisCoRouteA2A($Const$, $src$, $dst$)
    $H_h, H_v \leftarrow$ MinHopCount($Const, src, dst$)
                                  ▷ w.l.o.g. $s_{0,0} = src$, $s_{H_h,H_v} = dst$, $dst$ is north east of $src$
    $route_s \leftarrow [src]$
    $route_t \leftarrow [dst]$
    $i \leftarrow 0$
    $j \leftarrow H_h$
for $H_h$ many times do
    $reward_s \leftarrow |\varphi_{i,0} + \varphi_{i+1,0}|$
    $reward_t \leftarrow |\varphi_{j,H_v} + \varphi_{j-1,H_v}|$
    if $reward_s < reward_t$ then
        $route_t \leftarrow s_{j-1,H_v} :: route_t$
        $j \leftarrow j - 1$
    else
        $route_s \leftarrow route_s :: s_{i+1,0}$
        $i \leftarrow i + 1$
    end if
end for
assert $i = j$                    ▷ $s_{i,0}$ and $t_{j,H_v}$ are on same orbital plane
if $H_v = 0$ then                 ▷ is $route_t[0]$ also last element of $route_s$?
    $route_t \leftarrow route_t[1 :]$                  ▷ remove first element
else
    $route_s \leftarrow route_s +\!\!+ [s_{i,1}, \ldots, s_{i,H_v-1}]$
end if
return $route_s +\!\!+ route_t$
End procedure
```

strategy used in the previous case. The primary distinction from the earlier algorithm lies in the selection criterion. Here, an intra-plane hop is chosen on the side (source or destination) where the absolute value of the sum of the latitudes of the involved satellites is *smaller*, thereby prioritizing hops that occur closer to the Equator. This procedure is formally described in Algorithm 3.

Throughout the following discussion, this algorithm will be referred to either as *DisCoRoute* or as *DisCo4T*.

Algorithm 3 DisCoRoute (Cases A2D & D2A)

```
procedure DisCoRouteA2D($Const$, $src$, $dst$)
    $H_h, H_v \leftarrow$ MinHopCount($Const, src, dst$)
                          ▷ w.l.o.g. $s_{0,0} = src$, $s_{H_h,H_v} = dst$, $dst$ is north east of $src$
    $route_s \leftarrow [src]$
    $route_t \leftarrow [dst]$
    $i \leftarrow 0$
    $j \leftarrow H_v$
    for $H_v$ many times do
        $reward_s \leftarrow |\varphi_{0,i} + \varphi_{0,i+1}|$
        $reward_t \leftarrow |\varphi_{H_h,j} + \varphi_{H_h,j-1}|$
        if $reward_s < reward_t$ then
            $route_s \leftarrow route_s :: s_{0,i+1}$
            $i \leftarrow i + 1$
        else
            $route_t \leftarrow s_{H_h,j-1} :: route_t$
            $j \leftarrow j - 1$
        end if
    end for
    assert $i = j$                   ▷ $s_{0,i}$ and $t_{H_v,j}$ are reachable via horiz. hops
    if $H_h = 0$ then                ▷ is $route_t[0]$ also last element of $route_s$?
        $route_t \leftarrow route_t[1:]$              ▷ remove first element
    else
        $route_s \leftarrow route_s \mathbin{+\!\!+} [s_{1,i}, \ldots, s_{H_h-1,i}]$
    end if
    return $route_s \mathbin{+\!\!+} route_t$
end procedure
```

Chapter 3

Methodology and New Algorithms

In this chapter the different methodologies employed during the research are reported, describing the main characteristics, formulations and procedures. In particular, this chapter will represent the logical thread followed during the research, touching the main findings in the attempt to test DisCoRoute in a time-varying scenario and to adapt it to both 3T topology and Walker-Star costellation.

3.1 Time varying DisCoRoute

This section describes the MATLAB-based implementation developed to simulate and evaluate DisCoRoute algorithm within a dynamic satellite network environment. Notably, in [3], the algorithm was evaluated solely in static scenarios, i.e. at a single fixed time instant, where the positions of the satellites remain unchanged. In contrast, our analysis considers a dynamic framework, evaluating the algorithm over a time interval during which satellite positions evolve. Within this context, the objective is to analyze routing behavior over time, assess End-to-End (E2E) delay performance, and visualize route stability under various time instants, with the results reported in Section 4.2.

3.1.1 MATLAB implementation

The simulation framework has been meticulously designed to operate within a scenario in which satellites are organized according to a constellation-based topology. The constellation considered in this study is the Walker-Delta $65° : 2080/52/10$ [11]. The initialization of such a constellation is performed through the `walkerDelta` function [12], wherein all relevant parameters defining the constellation's configuration are explicitly specified, and a `constellation` object is instantiated. The parameters are detailed as follows:

- **Scenario**: The simulation environment is instantiated using the `satelliteScenario` function [13], which introduces a a scenario controller (`sc`) and enables the modeling of satellite constellations, the visualization of satellite dynamics, and, most crucially for this analysis, the specification of temporal parameters including the initial time, stop time, and the sample time of the simulation. For this specific case study, the simulation is initialized at `11:00:00.000` (UTC) on March 28th, 2025, and proceeds for a duration of 12 hours. The sample time, representing the time interval between successive sampled positions of the satellites, is also defined within this framework and set to 15 minutes.

- **Radius**: This parameter represents the orbital radius of the satellites and is computed as the sum of the Earth's radius (6378.137 kilometers as specifed in the WGS84 reference system [14]) and the orbital altitude, which has been set to 310 kilometers. Thus, the total radius is 6688.137 kilometers.

- **Inclination**: This denotes the angle between the orbital planes and the equatorial plane. In this configuration, the inclination α is fixed at 65°.

- **Total Number of Satellites**: The total number of satellites in the constellation is given by the product $P \cdot Q$, yielding a total of 2080 satellites.

- **Number of Orbital Planes**: This corresponds to the value of P and is set to 52, indicating the number of distinct orbital planes in the constellation.

- **Phasing Factor**: Denoted by F, this parameter determines the angular displacement between satellites in adjacent planes and is set to 10 in this configuration.

The developed simulator is specifically designed to support the DisCoRoute routing algorithm, and it facilitates a comprehensive temporal analysis of path selection dynamics and communication delay metrics. The implementation achieves the following objectives:

- Executes routing computations at each discrete time step throughout the simulation duration.

- Assesses the stability and consistency of routing paths over time.

- Calculates E2E communication delays.

- Provides a visual representation of the evolution of routing decisions and associated performance indicators across the simulation timeline.

The simulation advances in a step-by-step manner, orchestrated by the scenario controller. At each simulation step, the current simulation time is recorded and systematically stored within a vector of strings. Intermediate timestamps are formatted into standardized `HH:mm:ss` strings, thereby ensuring consistency and clarity during post-simulation visualization. This methodology enables precise alignment between the routing outcomes and their corresponding temporal instances within the simulation timeline.

The simulation is conducted over a total of 10,000 distinct source-destination satellite pairs within the constellation. These pairs are randomly selected to ensure statistical representativeness and avoid any structural bias.

For each satellite in the constellation, positional data are retrieved from the pre-defined `constellation` object. These positions are extracted in both Earth-Centered Earth-Fixed (ECEF) and geographic coordinate systems. The geographic coordinates (latitude, longitude, altitude) are utilized by the DisCoRoute routing algorithm to determine the optimal communication path between the source and destination satellites. Conversely, the ECEF coordinates are employed to compute the E2E latency of the various paths corresponding to each simulated pair and timestamp.

As previously discussed, the DisCoRoute routing procedure is executed at each simulation time step for every selected source-destination satellite pair. During each simulation step, the following operations are performed in sequential order:

- **Minimum Hop Count Calculation:** Determine the minimum number of hops required to connect the source and destination nodes within the current network topology.

- **Routing Execution:** Invoke the routing algorithm to compute the optimal communication path based on the current satellite positions.

- **End-to-End Delay Computation:** Evaluate the E2E delay by considering the physical positions of the satellites, the speed of light for signal propagation, and a transition time. The transition time accounts for the processing delay incurred at each satellite while forwarding a packet toward the next node along the computed route. In this simulation framework, the transition time is treated as a fixed cost and is set to 1 millisecond per node.

- **Data Logging:** Store the resulting routing paths and corresponding latency values for subsequent analysis and performance evaluation.

Upon completion of all routing computations, the script proceeds to evaluate the temporal evolution of both the routing paths and the corresponding latency values. The first aspect is analyzed by comparing the routing solution for each source-destination satellite pair at a given simulation time step with that obtained at the subsequent time instant. A change flag is raised whenever a discrepancy is detected between the two paths. It is important to emphasize that the number of hops required to complete a route remains constant over time, as the MinHopCount model is based on the geometric configuration of the constellation rather than on the real-time positions of the satellites. What varies instead are the specific satellites traversed along the route. Indeed, the latitudes of the satellites vary over time, and DisCoRoute adapts the routing path at each time step by selecting satellites that are geographically closer to the polar regions, guided by the hop count and propagation direction inferred from the MinHopCount model.

The second phase of the analysis involves computing the standard deviation of the latency evaluated for each satellite pair across the entire simulation time window. Additionally, the overall distribution of E2E delay is evaluated for each individual simulation time step, providing insight into the temporal evolution and variability of communication performance throughout the simulation.

These two evaluations are essential to assess the robustness and temporal consistency of the routing algorithm in the context of highly dynamic satellite network environments.

Finally, three primary visualizations are generated to facilitate a comprehensive analysis of the collected simulation data:

1. **Latency Plot**: A series of violin plots representing the distribution of E2E delay at each simulation time step. These plots are based on the latency results corresponding to all computed routes at the respective time instants, thereby offering a detailed evaluation of the routing algorithm's performance in terms of latency.

2. **Standard Deviation Plot**: A graphical representation showing the evolution of the standard deviation in latency for each source-destination satellite pair over the entire simulation period. This plot highlights the temporal variability and consistency of routing performance across the constellation.

3. **Path Change Frequency**: A bar chart illustrating the percentage of routing paths that either remained unchanged or underwent modifications at each simulation time step. This visualization enables an intuitive assessment of the algorithm's responsiveness and adaptability within a dynamic orbital environment.

Modularity and Extensibility

The implementation is deliberately structured to promote modularity and extensibility. Additional routing algorithms can be seamlessly integrated by introducing corresponding conditional branches, provided that they conform to the standardized input/output interface employed by the existing routing functions. Moreover, the clear separation between data acquisition, computational procedures, and visualization routines enhances both the maintainability of the codebase and the potential for reuse in future developments or related research endeavors.

3.2 Adaptation of DisCoROute to 3T scenario

In this section, we introduce a preliminary adaptation of the DisCoRoute algorithm to the 3T scenario. The original DisCoRoute algorithm was conceptually thought for exclusively the 4T scenario, thus, when used in a 3T topology, it fails to converge to a solution. From these findings, we conceived a new heuristic, still based on the major characteristics of the original algorithm, which would be able to find the path from a source satellite to a destination in the 3T scenario, within certain acceptable values of latency, computational time and number of hops needed to complete a path.

3.2.1 Missing Horizontal Links Analysis

Before introducing the new routing algorithm, it is important to assess one of the issues that might rise from a rough adaptation of DisCoRoute without changing the mathematical model to compute the minimum hop path: routes with only horizontal hops are the most critical ones for DisCoRoute in an interleaved scenario, since, from MinHopCount, no vertical hops are expected to be used (H_v =0). Then, when DisCoRoute tries to compute a route between two satellites in this scenario, it will fail to complete the path as it will not be able to move in the vertical direction. To measure the impact of this scenario over the routing algorithm, we estimate the probability and occurrences of horizontal-only paths over a certain number of iterations. By *iterations*, we refer to the number of randomly selected source-destination satellite pairs simulated. These measurements indicate the number of paths with a routing failure probability of 100% if the algorithm is not adapted to the case of horizontal-only paths.

The results presented in Figure 3.1 clearly illustrate that the number of occurrences increases progressively with the number of iterations. In particular, for each increment in the iteration count, the number of occurrences tends to grow by approximately an order of magnitude. The probability associated with horizontal-only paths is approximately 2.72%, based on empirical estimation over 10^6 iterations. Despite this value appearing numerically low, its significance becomes more apparent when considered in conjunction with the absolute number of occurrences. From this perspective, it becomes clear that the presence of such horizontal-only paths is far from negligible. Therefore, it is imperative to adapt the DisCo3T framework in order to effectively mitigate this issue and enhance the robustness of the system.

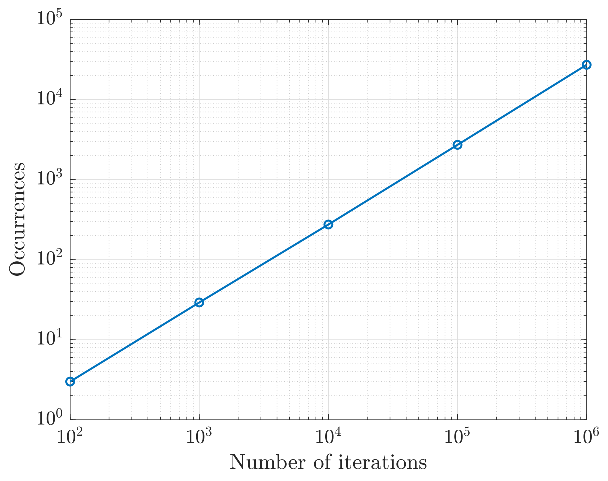


Figure 3.1: Only horizontal paths occurrences

3.2.2 DisCoRoute heuristic adaptation

In this section, we present the initial attempt to extend the DisCoRoute framework to the 3T scenario. The details of the corresponding procedure are outlined in Algorithm 4. The first noteworthy aspect is that, unlike the 4T scenario, the algorithm does not necessitate the use of two distinct procedures to handle the A2A/D2D and A2D/D2A cases. The algorithm takes as input a set of parameters that define the constellation, denoted as $Const.$, which includes values such as the number of orbital planes (P), the number of satellites per plane (Q), and the phasing factor (F). Additionally, the input includes the source satellites, represented by src, from which the routing process originates, and the destination satellite, denoted by dst, which marks the termination point of the route. Another crucial input is the adjacency matrix, $adjMatrix$, which is a $PQ \times PQ$ matrix where each element indicates the connectivity between satellite pairs: entries with a value of 1 represent a direct link between two satellites, while entries with a value of 0 denote the absence of such a connection. Furthermore, the algorithm continues to rely on the MinHopCount model, which determines the number of horizontal and vertical hops necessary to establish a connection between any pair of satellites within the constellation. Subsequently, the routing structures originating from the source and destination are initialized, starting respectively from the designated source satellite and the specified destination satellite.

Of particular importance is Line 5 in Algorithm 4, which implements the mitigation strategy for the horizontal-only path issue, thoroughly discussed in Section 3.2.1. To address this problem, the algorithm increases the number of required vertical hops by an amount equal to twice the number of horizontal hops.

This adjustment is based on the characteristic behavior of the 3T scenario, wherein the absence of a single horizontal link necessitates two additional vertical hops to effectively bypass the missing connection.

At this stage of development, in the absence of a refined model, other than the MinHopCount framework, that accurately estimates the total number of hops required for a given path, we adopt a conservative, worst-case assumption. Specifically, we presume that none of the horizontal links along the intended path are available. As a result, the total vertical hop budget is augmented by a factor of $2 \cdot H_h$, where H_h denotes the number of horizontal hops originally required according to the MinHopCount model in 4T. This precautionary measure ensures that the routing algorithm avoids premature failures due to underestimated connectivity constraints. Subsequently, the algorithm initializes the horizontal hop counter, denoted by h, to zero and enters the main iterative loop. This `while` loop continues execution as long as there remain either horizontal or vertical hops available for use. It is important to emphasize that, in order to prevent routing failure, the horizontal hops must be exhausted prior to the depletion of vertical hops. This ordering ensures that the routing process can still leverage vertical transitions to compensate for any unavailable horizontal links, thereby maintaining connectivity throughout the path construction.

At each iteration of the loop, the algorithm identifies the next satellites to be visited in the horizontal direction, proceeding according to the propagation direction defined by the MinHopCount model, both from the source side (denoted as $next_s$) and from the destination side (denoted as $next_t$). Once these candidate satellites are determined, the algorithm consults the adjacency matrix, $adjMatrix$, to verify the presence or absence of the corresponding horizontal cross-links. Depending on the outcome of this verification, four distinct scenarios may arise:

1. **Both horizontal hops are available**: In this scenario, direct horizontal links are available from both src to $next_s$ and from dst to $next_t$, allowing the algorithm to proceed analogously to the procedure described in Algorithm 2. Specifically, two reward values are computed, one for each direction, by taking the absolute value of the sum of the latitudes (denoted with φ) of the current node (either source or destination) and its corresponding next satellite. These reward values are used to guide the decision on which path to extend in the current iteration. Let $reward_s$ denote the reward associated with the path propagating from the source, and $reward_t$ the reward from the destination direction. If $reward_t > reward_s$, the algorithm appends the next destination-side satellite to the partial path $route_t$ and updates the current destination node to $next_t$. Conversely, if $reward_s \geq reward_t$, it appends the next source-side satellite to $route_s$ and updates the current source node to $next_s$. After completing this comparison and update, the horizontal hop

counter is incremented to reflect the progress made in the current iteration.

2. **Horizontal hop available only from source**: Here, the horizontal link from *src* to $next_s$ is present, while the corresponding link from *dst* to $next_t$ is absent. The algorithm updates the path from the source side and updates the current source node to $next_s$ and the counter.

3. **Horizontal hop available only from destination**: This is the symmetric case to the previous one, in which the link from *dst* to $next_t$ is available, but the source-side horizontal link is missing. The algorithm updates the path from the destination side accordingly, the current source node to $next_t$ and the counter.

4. **Both horizontal hops are NOT available**: If cross-links are unavailable in both directions, that is, neither from *src* to $next_s$ nor from *dst* to $next_t$, the algorithm resorts to vertical movement, while still adhering to the propagation direction prescribed by the MinHopCount model. The decision regarding whether to update the source or destination path is based on a comparison of the latitudes of the current source and destination satellites. Specifically, if the latitude of the current source satellite is lower than that of the destination, the algorithm appends the next satellite in the vertical direction to the source-side path and updates the source node to $next_s$. Conversely, if the latitude of the destination satellite is less than or equal to that of the source, the vertical move is performed from the destination side, updating $route_t$ and assigning $next_t$ to the *dst* variable. This heuristic mirrors the strategy employed in Algorithm 3 of the original DisCoRoute algorithm, where satellites at lower latitudes are preferentially selected for vertical transitions. The rationale behind this preference is to have horizontal hops at higher latitudes, where inter-satellite distances are reduced. This, in turn, has the potential to minimize the overall latency of the computed path. Finally, the number of vertical hops is decreased.

Once the `while` loop concludes, the values of the last satellites appended to the source and destination partial paths, $route_s$ and $route_t$, are assigned to the variables *src* and *dst*, respectively. The algorithm then verifies whether these two satellites reside on the same orbital plane. This is determined by comparing their orbital indices, denoted as o_{src} for the source-side satellite and o_{dst} for the destination-side satellite. If the satellites are not located on the same plane ($o_{src} \neq o_{dst}$), the algorithm declares a routing failure, as it would be impossible to connect them directly using vertical links. However, if they do belong to the same plane, the routing process continues by establishing the vertical connection between them.

To compute the correct number of vertical hops required for this final connection, the algorithm invokes the MinHopCount model once more. This model outputs both the number of vertical hops, H_v, and the direction of propagation. Based on the value of H_v, the algorithm proceeds as follows:

- If $H_v = 0$, it implies that the source and destination nodes coincide. In this case, the first element of $route_t$ is removed to avoid duplication of the common node in the final path.
- If $H_v = 1$, it indicates that the next satellite required to complete the path from src is exactly dst. Since the destination node will already be included in the merging of the two sub-paths, there is no need to append it again.
- If $H_v > 1$, the necessary vertical hops are appended to $route_s$ to complete the connection from src to dst.

Finally, the two partial paths, $route_s$ and $route_t$, are concatenated to form the complete routing path, which is returned as the final output of the algorithm.

This heuristic represents an initial step in adapting the DisCoRoute framework to the 3T scenario. However, it exhibits certain limitations. Notably, it still relies on a mathematical model originally designed for the 4T architecture to estimate the minimum number of hops. As will be demonstrated in Section 4.3, this mismatch results in a significant performance gap when compared to the Dijkstra 3T algorithm, which is employed as the reference baseline for the 3T case. This discrepancy motivates the development of a new mathematical model, specifically tailored to the 3T scenario, for accurately computing the minimum number of hops between any two nodes. The details of this enhanced model are presented in the following section.

Algorithm 4 DisCoRoute 3T (Cases A2A/D2D & A2D/D2A)

```
procedure Disco3T_V1($Const., src, dst, adjMatrix$)
    $H_h, H_v \leftarrow$ MinHopCount($Const., src, dst$)
    $route_s \leftarrow [src]$
    $route_t \leftarrow [dst]$
    $H_v \leftarrow H_v + 2 \cdot H_h$
    $h \leftarrow 0$
    while ($h < H_h$ & $H_v \geq 0$) do
        $next_s \leftarrow get\_h\_next(Const., src)$          ▷ Get next satellite in horizontal direction
        $next_t \leftarrow get\_h\_next(Const., dst)$
        if $adjMatrix[src, next_s]$ & $adjMatrix[dst, next_t]$ then
            $reward_s \leftarrow |\varphi_{src} + \varphi_{next_s}|$
            $reward_t \leftarrow |\varphi_{dst} + \varphi_{next_t}|$
            if $reward_s < reward_t$ then
                $route_t \leftarrow next_t :: route_t$
                $dst \leftarrow next_t$
            else
                $route_s \leftarrow route_s :: next_s$
                $src \leftarrow next_s$
            end if
            $h \leftarrow h + 1$
        else if $adjMatrix(src, next_s)$ then
            $route_s \leftarrow route_s :: next_s$
            $src \leftarrow next_s$
            $h \leftarrow h + 1$
        else if $adjMatrix(dst, next_t)$ then
            $route_t \leftarrow next_t :: route_t$
            $dst \leftarrow next_t$
            $h \leftarrow h + 1$
        else
            if $|\varphi_{src}| < |\varphi_{dst}|$ then
                $next_s \leftarrow get\_v\_next(Const., src)$
                $src \leftarrow next_s$
            else
                $next_t \leftarrow get\_v\_next(Const., dst)$      ▷ Get next satellite in vertical direction
                $dst \leftarrow next_t$
            end if
            $H_v \leftarrow H_v - 1$
        end if
    end while
    $src \leftarrow route_s[end]$                ▷ Assign the last element of $route_s$ to $src$
    $dst \leftarrow route_t[0]$                  ▷ Assign the first element of $route_t$ to $dst$
    assert($o_{src} = o_{dst}$)                  ▷ Check they are on the same plane
    $H_v \leftarrow$ MinHopCount($Const., src, dst$)
    if $H_v = 0$ then
        $route_t \leftarrow route_t\,[1, :]$          ▷ Remove the first element
    else
        $route_s \leftarrow route_s \mathbin{+\!\!+} [s_{o_{src},1}, ..., s_{o_{src},H_v-1}]$
    end if
    return $route_s \mathbin{+\!\!+} route_t$
end procedure
```

3.3 MinHopCount for 3T scenario

Since MinHopCount was conceptually designed for the 4T scenario, a direct implementation in the 3T scenario is not feasible. The model does not account for the intermittent unavailability of horizontal links and lacks mechanisms to compute the additional vertical hops required to overcome missing horizontal hops. For these reasons, we propose an adapted version of the MinHopCount, called Interleaved Scenario - MinHopCount (IS-MHC).

3.3.1 Interleaved Scenario - MinHopCount (IS-MHC)

Assuming no initial link failures, the interleaved topology naturally constrains minimum-hop paths to generally follow a staircase-like trajectory.

Given the constant availability of vertical hops in the 3T scenario, horizontal hops emerge as the primary limiting factor in route computation. Let H_h and H_v denote the minimum number of horizontal and vertical hops, respectively, required to connect a source and a destination satellite in the 4T architecture [3], following a given direction (i.e., one of $\{\nearrow, \nwarrow, \searrow, \swarrow\}$). Analogously, let H_h^* and H_v^* represent the corresponding minimum hop counts in the 3T scenario.

Trivially, in the interleaved configuration, the equality $H_h^* = H_h$ always holds, as the number of horizontal hops required does not change between the 4T and 3T scenarios. Furthermore, in cases where $H_v \geq H_h$, it follows that $H_v^* = H_v$, since it is always feasible to construct a valid staircase path that connects the source and destination planes across different latitudes. An example of such a path is illustrated by the green path in Figure 3.2.

However, in all remaining cases, specifically, when $H_v < H_h$, the vertical hop count increases in the 3T model. This is a direct consequence of the absence of certain horizontal cross-links, which necessitates additional vertical detours to maintain connectivity. More precisely, for each horizontal hop that is missing relative to the 4T scenario, the 3T model requires two extra vertical hops to compensate. This effect is exemplified by the blue path in Figure 3.2.

In order to compute the 3T hop counts with greater accuracy, we consider the rectangular grid, referred to as the *search space*, defined by the MinHopCount model. Within this space, the source and destination satellites are positioned at diagonally opposite corners, thereby delimiting the boundaries of the region over which all possible routing paths may be explored.

Within this grid, given $m \triangleq H_v + 1$ and $n \triangleq H_h$, we define two matrices:

- The *index map* matrix $\mathbf{IM} \in \{1, \ldots, T\}^{m \times (n+1)}$, which collects all satellite IDs in the search space. The primary applications of this matrix are found within the routing algorithm introduced in Section 3.4.

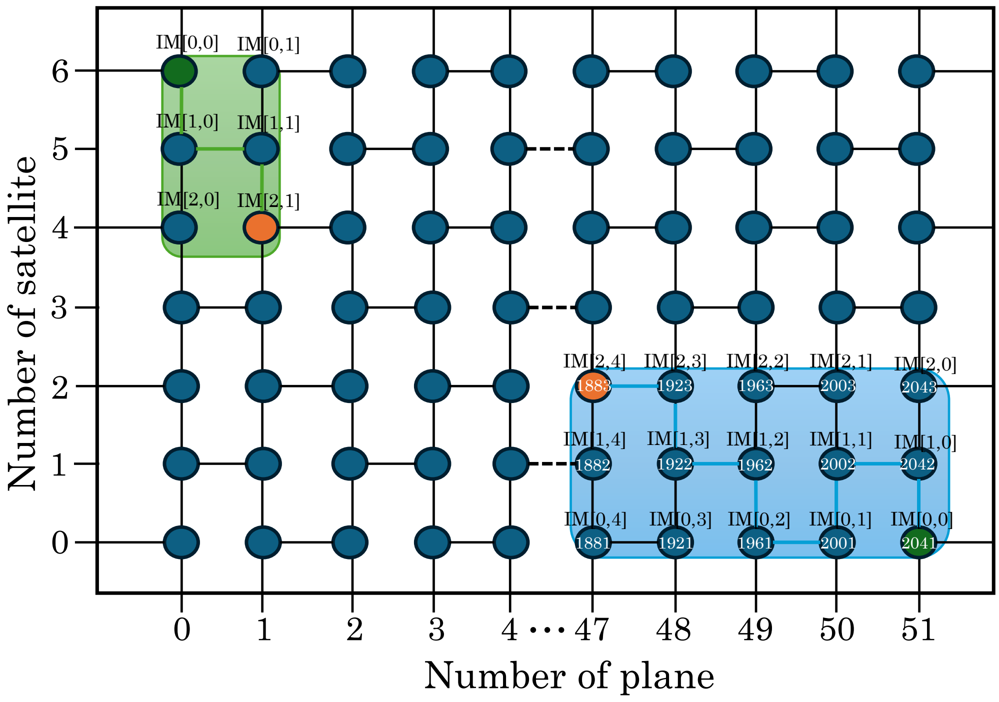


Figure 3.2: Extract of a 3T ISL topology for Walker-Delta 65°: 2080/52/10 constellation with interleaved inter-plane links. The green path is an example of the case $H_v \geq H_h$, where the hop counts for 4T and 3T coincide. In contrast, the blue path shows that two additional hops are needed in 3T. The two source nodes are marked as ● while the destination nodes are shown as ●. In addition, the annotations show the global node IDs and how nodes are assigned to the **IM** matrix as described in Section 3.3.1.

- The *inter-plane adjacency matrix* $\mathbf{H} \in \{0,1\}^{m \times n}$. The primary applications of this matrix are found within the application of the IS-MHC introduced in this section.

The matrix **IM** is defined so that $\mathbf{IM}[0,0]$ corresponds to the source, while a generic satellite is indexed by $\mathbf{IM}[i^*, o^*]$, where o^* is the plane offset and i^* is the intra-plane offset relative to the source, with $i^* \in \{0, \ldots, m-1\}$ and $o^* \in \{0, \ldots, n-1\}$.

Inter-plane links in the grid are represented by the adjacency matrix **H**, where $\mathbf{H}[i^*, o^*] = 1$ indicates a horizontal link exists between $\mathbf{IM}[i^*, o^*]$ and $\mathbf{IM}[i^*, o^* + 1]$, and $\mathbf{H}[i^*, o^*] = 0$ otherwise.

For instance, the matrices $\mathbf{IM} = \begin{pmatrix} 2041 & 2001 & 1961 & 1921 & 1881 \\ 2042 & 2002 & 1962 & 1922 & 1882 \\ 2043 & 2003 & 1963 & 1923 & 1883 \end{pmatrix}$ and $\mathbf{H} = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix}$ correspond to the blue path in Figure 3.2, where the source is at $(o, i) = (51, 0)$

with ID 2041 and the destination is at $(o, i) = (47, 2)$ with ID 1883. The entry $\mathbf{H}[0, 0] = 0$ reflects the absence of a link between satellites 2041 and 2001.

This representation enables a precise model for estimating the vertical hop count H_v^* in the 3T scenario.

Given the $m \times n$ adjacency matrix $\mathbf{H}$, we distinguish two cases:

- If $m > n$, then $H_v^* = H_v$
- If $m \leq n$:

$$H_v^* = \begin{cases} H_v + 2(k+1) & \text{if } n - m = 2k + 1 \\ H_v + 2k & \text{if } n - m = 2k \text{ and } tr(\mathbf{M}) = m \\ H_v + 2(k+1) & \text{if } n - m = 2k \text{ and } tr(\mathbf{M}) \neq m \end{cases} \tag{3.1}$$

  where $k \in \mathbb{N}$, $\mathbf{M}$ is an $m \times m$ submatrix of $\mathbf{H}$ originating from the source, and the trace operator $tr(\mathbf{M})$ sums the diagonal elements of $\mathbf{M}$.

The IS-MHC formulation remains fundamentally rooted in the original MinHop-Count model, as it continues to derive the number of required hops in both the horizontal and vertical directions from it. The expression proposed in Equation (3.1) can thus be interpreted as a procedure for quantifying the additional vertical hops necessitated by the 3T topology, effectively capturing the routing overhead introduced by the absence of certain horizontal inter-plane links. In particular, the multiplicative factors k or $k+1$ appearing in Equation (3.1) represent the number of missing horizontal hops along the path between two satellites. These missing hops are then multiplied by two, reflecting the fact - previously discussed in Section 3.2 - that each unavailable horizontal hop in the 3T topology requires two additional vertical hops to maintain end-to-end connectivity.

Equation (3.1) is then applied to the four directional solutions produced by MinHopCount in 4T, namely $(H_v^{\nwarrow}, H_v^{\swarrow}, H_v^{\nearrow}, H_v^{\searrow})$, to derive their corresponding intra-plane hop counts in 3T. Meanwhile, the horizontal hop counts remain unchanged: $H_h^{*\leftarrow} = H_h^{\leftarrow}$ and $H_h^{*\rightarrow} = H_h^{\rightarrow}$.

The total minimum-hop path is then computed by evaluating the minimum among all four directional combinations:

$$\min\left\{H_h^{*\leftarrow} + H_v^{*\nwarrow},\ H_h^{*\leftarrow} + H_v^{*\swarrow},\ H_h^{*\rightarrow} + H_v^{*\nearrow},\ H_h^{*\rightarrow} + H_v^{*\searrow}\right\}$$

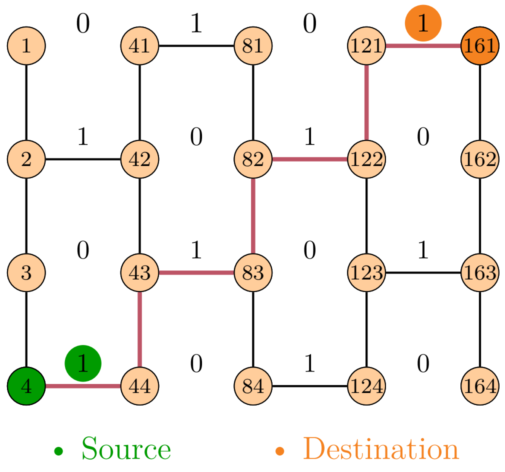
Finally, it is worth noting that IS-MHC also provides directional indicators for the path achieving the minimum hop count. In the next section, we leverage this framework to define the DisCo3T algorithm, while its validation will be discussed in Section 4.4.

The following examples illustrate the application of the algorithm, with the aim of familiarizing the reader with its operational behavior and decision-making process under various routing scenarios.

Example 1

Continuing with the Walker-Delta constellation configuration $65° : 2080/52/10$, this example illustrates a representative "staircase" routing solution. For simplicity, in this and the subsequent examples, we restrict the visual representation to only the satellites and the intra-plane and inter-plane links that fall within the defined search space. Additionally, the satellite identifiers are presented solely by their unique IDs, omitting the (o, i) tuple notation.

In the scenario under consideration, the source satellite has ID 4 and the destination satellite has ID 161. According to the MinHopCount model, the computed hop counts are $H_h = 4$ and $H_v = 3$, corresponding to a North-East directional path. In the following examples, we will not detail the entire routing procedure but will instead focus on the application of Equation (3.1), which represents the most critical, and potentially the most nuanced, component of the IS-MHC formulation. Here the search space representation:



Next, we extract the satellite identifiers and the adjacency matrix corresponding to the horizontal hops (shown on the left). These matrices are then rearranged so that the source node and the cross-link associated with the source satellite, are positioned in the top-left corner (highlighted in green), while the destination node and its associated cross-link are placed in the bottom-right corner (highlighted in orange). This reorganization facilitates the construction of the **IM** and **H**, depicted on the right:

$$\begin{bmatrix} 1 & 41 & 81 & 121 & 161 \\ 2 & 42 & 82 & 122 & 162 \\ 3 & 43 & 83 & 123 & 163 \\ 4 & 44 & 84 & 124 & 163 \end{bmatrix} \quad \rightarrow \quad \mathbf{IM} = \begin{bmatrix} 4 & 44 & 84 & 124 & 164 \\ 3 & 43 & 83 & 123 & 163 \\ 2 & 42 & 82 & 122 & 162 \\ 1 & 41 & 81 & 121 & 161 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix} \quad \rightarrow \quad \mathbf{H} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

$$\mathbf{M} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix} \equiv \mathbf{H}$$

From the **H** matrix, it is possible to obtain the values of m and n, and check the conditions illustrated in Equation (3.1):

$$\begin{gathered} m = 4, \quad n = 4 \\ n - m = 0 \Rightarrow \textit{even} \quad \wedge \quad \operatorname{tr}(\mathbf{M}) = 4 = m \\ k = \tfrac{n-m}{2} = 0 \end{gathered}$$

Once the conditions have been verified and the value of k has been computed, the corresponding expression from Equation (3.1) can be applied to determine the total number of vertical hops required under the IS-MHC model:

$$H_v^* = H_v + 2 \cdot k = 3 + 2 \cdot 0 = 3$$

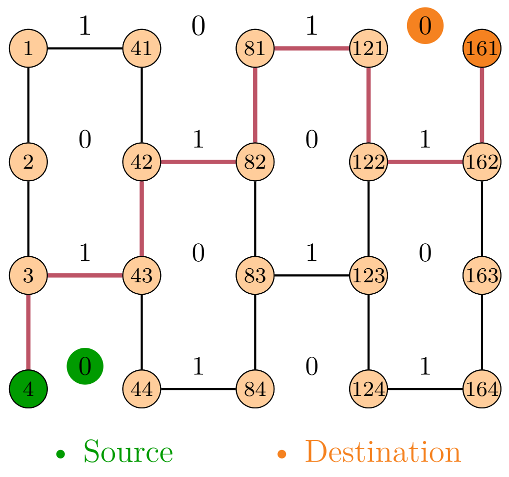
Table 3.1 presents the final results obtained from both the MinHopCount and IS-MHC models. In this particular case, no differences are observed in the hop count calculations of the two models, as a valid "staircase" path connecting the source and destination can be successfully constructed and the **M** matrix coincides with the **H** matrix. The results indicate that, in this particular scenario, the 3T and 4T configurations yield identical minimum hop counts, demonstrating that the change in topology does not always impose additional routing overhead.

Example 2

In this example, we revisit the same scenario analyzed in the previous case, maintaining identical source and destination satellites as well as constellation parameters. However, we changed its topology to observe the impact of different topologies in the same scenario. The current search space is as follows:

	MinHopCount	**IS-MHC**
H_h	4	4
H_v	3	3

Table 3.1: Example 1 results on horizontal and vertical hop counts for MinHop-Count and IS-MHC



It is immediately evident that the path with the minimum hop count, depicted in red, no longer exhibits the "staircase" structure observed in the previous example. For clarity, only one of the possible minimum-hop paths is illustrated; however, it is straightforward to verify that this observation holds true for all equivalent paths achieving the same hop count. As in the previous example, we proceed to construct the **IM** and **H** matrices:

$$\begin{bmatrix} 1 & 41 & 81 & 121 & 161 \\ 2 & 42 & 82 & 122 & 162 \\ 3 & 43 & 83 & 123 & 163 \\ 4 & 44 & 84 & 124 & 163 \end{bmatrix} \quad \rightarrow \quad \mathbf{IM} = \begin{bmatrix} 4 & 44 & 84 & 124 & 164 \\ 3 & 43 & 83 & 123 & 163 \\ 2 & 42 & 82 & 122 & 162 \\ 1 & 41 & 81 & 121 & 161 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix} \quad \rightarrow \quad \mathbf{H} = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}$$

$$\mathbf{M} = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix} \equiv \mathbf{H}$$

From the **H** matrix, it is possible to obtain the values of m and n, and check the conditions illustrated in Equation (3.1):

$$\begin{gathered} m = 4, \quad n = 4 \\ n - m = 0 \Rightarrow even \quad \wedge \quad \mathrm{tr}(\mathbf{M}) = 0 \neq m \\ k = \tfrac{n-m}{2} = 0 \end{gathered}$$

Once the conditions have been verified and the value of k has been computed, since the trace of **M** is different from m, the last formula of Equation (3.1) is applied to determine the total number of vertical hops required under the IS-MHC model:

$$H_v^* = H_v + 2 \cdot (k + 1) = 3 + 2 \cdot (0 + 1) = 5$$

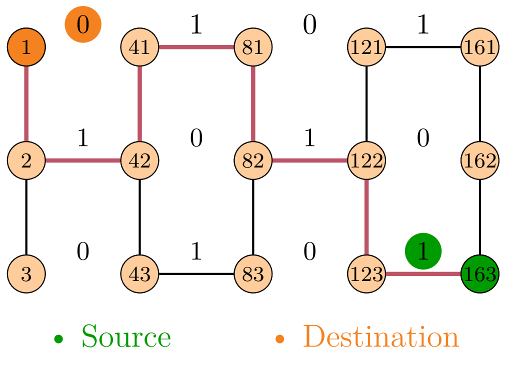
Table 3.2 presents the final results obtained from both the MinHopCount and IS-MHC models. In contrast to the previous example, the hop count results produced by the two models diverge. Specifically, the 3T scenario requires two additional vertical hops compared to the 4T case as a consequence of the change in the topology, which induced the absence of a valid "staircase" path connecting the source and destination, although the **M** matrix coincides with the **H** matrix.

	MinHopCount	**IS-MHC**
H_h	4	4
H_v	3	5

Table 3.2: Example 2 results on horizontal and vertical hop counts for MinHop-Count and IS-MHC

Example 3

In this example, we consider a scenario in which the **H** matrix is no longer square, unlike in the previous cases. Specifically, there exists an odd difference between the number of rows m and columns n. The corresponding search space is illustrated as follows:



Here, the source's ID is 163, while destination's ID is 1, with MinHopCount results equal to $H_h = 4$ and $H_v = 2$. We construct the **IM** and **H** matrices:

$$\begin{bmatrix} 1 & 41 & 81 & 121 & 161 \\ 2 & 42 & 82 & 122 & 162 \\ 3 & 43 & 83 & 123 & 163 \end{bmatrix} \quad \rightarrow \quad \mathbf{IM} = \begin{bmatrix} 163 & 123 & 83 & 43 & 3 \\ 162 & 122 & 82 & 42 & 2 \\ 161 & 121 & 81 & 41 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix} \quad \rightarrow \quad \mathbf{H} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}$$

$$\mathbf{M} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} \not\equiv \mathbf{H}$$

In this example, the **M** matrix differs from the **H** matrix due to the greater number of columns relative to rows. However, the **M** matrix does not affect the selection of the expression from Equation (3.1), as the difference between n and m is odd:

$$\begin{gathered} m = 3, \quad n = 4 \\ n - m = 1 \Rightarrow odd \\ k = \tfrac{n-m-1}{2} = 0 \end{gathered}$$

Given the verified conditions, we apply the first formula from Equation (3.1):

$$H_v^* = H_v + 2 \cdot (k + 1) = 2 + 2 \cdot 1 = 4$$

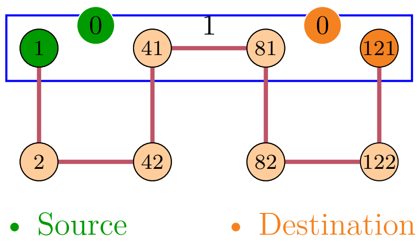
Table 3.3 presents the final results obtained from both the MinHopCount and IS-MHC models. The hop count results produced by the two models diverge, showing that the 3T scenario requires two additional vertical hops compared to the 4T case.

	MinHopCount	**IS-MHC**
H_h	4	4
H_v	2	4

Table 3.3: Example 3 results on horizontal and vertical hop counts for MinHop-Count and IS-MHC

Example 4 (horizontal-only path)

In this final example, we examine the application of the IS-MHC model to the horizontal-only path scenario, which, as discussed in Section 3.2.1 represents the most challenging case to handle due to the complete absence of vertical progression within the path. The source's ID is 1, while the destination's ID is 121. The MinHopCount model results are $H_h = 3$ and $H_v = 0$. The search space is delineated by the blue rectangle encompassing satellites $\{1, 41, 81, 121\}$ in the figure below. For enhanced clarity and to aid the reader's understanding of the routing behavior, we also depict the minimum hop count path in the 3T scenario using a red line, even though it extends beyond the defined search space:



From the search space, we construct the **IM** and **H** matrices:

$$\begin{bmatrix} 1 & 41 & 81 & 121 \end{bmatrix} \quad \rightarrow \quad \mathbf{IM} = \begin{bmatrix} 1 & 41 & 81 & 121 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 1 & 0 \end{bmatrix} \quad \rightarrow \quad \mathbf{H} = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}$$

$$\mathbf{M} = \begin{bmatrix} 0 \end{bmatrix} \neq \mathbf{H}$$

We check the conditions from Equation (3.1):

$$\begin{gathered} m = 1, \quad n = 3 \\ n - m = 2 \Rightarrow \textit{even} \quad \wedge \quad \operatorname{tr}(\mathbf{M}) = 0 \neq m \\ k = \frac{n-m}{2} = 1 \end{gathered}$$

Thus, we can compute the total number of vertical hops in the 3T scenario:

$$H_v^* = H_v + 2 \cdot (k + 1) = 0 + 2 \cdot 2 = 4$$

Table 3.4 presents the final results obtained from both the MinHopCount and IS-MHC models. The hop count results obtained from the two models differ significantly, with the 3T scenario requiring four additional vertical hops compared to the 4T case, which, being a horizontal-only path, did not involve any vertical movement. This example illustrates the effectiveness of the IS-MHC model in handling such cases, thereby addressing one of the primary limitations of the heuristic adaptation of DisCoRoute to the 3T setting, as discussed in Section 3.2.2. Specifically, it overcomes the challenge of accurately determining the number of vertical hops required, thus enabling proper management of horizontal-only paths.

	MinHopCount	**IS-MHC**
H_h	3	3
H_v	0	4

Table 3.4: Example 4 results on horizontal and vertical hop counts for MinHop-Count and IS-MHC

3.4 DisCo3T

Having developed a minimum hop count model tailored for the 3T scenario, capable of precisely determining the number of horizontal and vertical hops, we are now in a position to integrate this model into the routing algorithm. To this end, we introduce the final adaptation of DisCoRoute for the 3T topology, which we denote as *DisCo3T*. In the following, we present the corresponding algorithm, emphasizing its key features and illustrating how it leverages both the IS-MHC model and the **IM** matrix.

3.4.1 DisCo3T Routing Algorithm

The development of the DisCo3T algorithm builds upon the foundations established for DisCoRoute in the 4T context. However, due to the constraints imposed by the 3T topology on path construction between satellites, the version proposed here eliminates the need to differentiate between the cases of ascending-to-ascending (or descending-to-descending) and ascending-to-descending (or vice versa). These cases, which refer to the direction of satellite propagation along their orbital planes, previously required separate algorithmic treatments as discussed in [3] and Section 2.2.2.

Nonetheless, the key idea of DisCoRoute, favoring horizontal hops near the poles to minimize the distance between adjacent planes, is retained, as it continues to offer advantages in terms of end-to-end delay. The route is constructed in parallel from both the source ($route_s$) and the destination ($route_t$), provided the number of hops used remains below the threshold determined by IS-MHC. Horizontal hops are prioritized due to their limited availability, similarly to what has been done in Section 3.2.2.

The two paths grow toward each other in a “staircase” fashion until all horizontal hops are exhausted. If properly constructed, the remaining vertical hops suffice to connect the two segments. This strategy ensures route construction remains as close to the poles as possible, preserving the delay benefits.

As illustrated in Algorithm 6, the number of horizontal and vertical hops for both 3T and 4T topologies is obtained using IS-MHC, along with the travel direction that yields the shortest path. The dimensions of the search space, m and n, are then set, and the index map **IM** is initialized using the ESS (Expand Search Space) procedure (Algorithm 6, line 3), which is reported in Algorithm 5. This procedure expands the **IM** matrix, if required, and sets the indexing and initialization considering two distinct cases:

- $m \leq n$: The matrix **IM** is expanded following Algorithm 5, enabling the algorithm to explore potentially better paths beyond the search space defined by MinHopCount. Therefore, in this context, the term "expansion" refers to the process of including additional satellites in the search space beyond those strictly required by the MinHopCount model. Concretely, this corresponds to the addition of rows to the **IM** matrix. To avoid unnecessary computational and memory usage, the number of additional rows is limited based on the maximum allowable vertical hops without violating IS-MHC. This maximum depends on the parity and difference between m and n, as defined in Line 5 and Line 8.

Since the optimal direction for expansion cannot be known a priori, the matrix is symmetrically expanded above the source and below the destination by *expValue* rows, resulting in a total of $2 \cdot expValue + H_v + 1$ rows. To expand above the source, we move *expValue* satellites upward from the source, identifying a new top-left corner node, *srcExp*, from which the index map is constructed. Note that: (1) *srcExp* and *src* are distinct, and (2) the expansion is applied exclusively in the vertical direction, via $2 \cdot expValue$ additional rows. This design choice reflects the 3T architecture, where the additional hops, compared to the 4T case, occur only vertically. Horizontal expansion is avoided as it would violate the constraints of IS-MHC, which guarantees that the number of required horizontal hops remains constant between 3T and 4T, as detailed in Section 3.3.1.

Following expansion, index variables and partial routes are initialized for both the source $(j, l, route_s)$ and destination $(p, q, route_t)$, accounting for the offset introduced by *expValue*.

- $m > n$: In this scenario, matrix expansion is unnecessary, as the shortest path lies entirely within the initially defined space. The index variables and the partial routes are both initialized directly from the source and the destination.

Concluded the ESS procedure, Algorithm 6 proceeds by obtaining the horizontal successors of the last nodes visited by both routes: $\mathbf{IM}[j, l+1]$, from the source moving toward the destination, and $\mathbf{IM}[p, q-1]$, from the destination moving toward the source. Boolean conditions $Cond_s$ and $Cond_t$ are then computed based on the existence of horizontal links between the respective node pairs, i.e. they can take on the values true (1) or false (0) depending on the presence or absence of a horizontal hop between satellites $\mathbf{IM}[j, l]$ and $\mathbf{IM}[j, l+1]$, and satellites $\mathbf{IM}[p, q]$ and $\mathbf{IM}[p, q-1]$. These values are retrieved from the adjacency matrix *adjM*, a square matrix which encodes link availability between satellites.

If both horizontal links are present ($Cond_s = Cond_t = 1$), the procedure between Line 12 and Line 15 in Algorithm 6 is executed. The algorithm calculates rewards for both directions using the absolute value of the sum of the latitudes of the current node and its successor, similarly to Line 9 and Line 10 in Algorithm 2. If

Algorithm 5 Expand Search Space

```
procedure ESS($Const., src, dst, adjM, H_h, H_v, SearchDir$)
    Compute $m, n$ based on $H_v, H_h$
    if $m \leq n$ then
        if $(n - m)$ is odd then
            $expValue \leftarrow \frac{n-m-1}{2} + 2$
            Retrieve satellite $srcExp$ in position $(o, i \pm expValue)$
        else                                                        ▷ $n - m$ is even
            $expValue \leftarrow \frac{n-m}{2} + 1$
            Retrieve satellite $srcExp$ in position $(o, i \pm expValue)$
        end if
        $H_v \leftarrow H_v + 2 \cdot expValue$
        IM ← ConstructIM($Const., srcExp, adjM, H_h, H_v, SearchDir$)

        $m^* = size(\mathbf{IM}, rows)$                                   ▷ Retrieve the number of rows
        $n^* = size(\mathbf{IM}, cols)$                                ▷ Retrieve the number of columns
        Initialize $route_s = [src], route_t = [dst]$
        Set $j = 0 + expValue, \quad l = 0 + expValue$
        Set $p = m^* - 1 - expValue, \quad q = n^* - 1 - expValue$
    else                                                        ▷ No expansion needed
        IM ← ConstructIM($Const., src, adjM, H_h, H_v, SearchDir$)
        Initialize $route_s = [src], route_t = [dst]$
        Set $j = 0$, $l = 0$, $p = m - 1$, $n - 1$
    end if
    return IM, $j$, $l$, $p$, $q$
end procedure
```

$reward_t > reward_s$, this implies that nodes along the destination path are closer to the poles, making them more favorable. The successor of $\mathbf{IM}[p, q]$ is added to $route_t$. Conversely, if $reward_s \geq reward_t$, the successor of $\mathbf{IM}[j, l]$ is added to $route_s$.

If only one horizontal link is available, the corresponding successor, from source or destination, is appended. When neither is available, a vertical hop is taken, applying a similar reward-based decision process using the absolute value of the sum of the latitudes. Every time a new satellite is appended to $route_s$ or $route_t$, the number of horizontal hops H_h^* or the number of vertical hops H_v^*, depending on the case, is decreased by one.

This process continues until all horizontal hops have been used. The remaining vertical hops are then applied to connect $route_s$ to the last node in $route_t$. Finally, the complete route is formed by appending $route_t$ to $route_s$, and the full path is returned.

Algorithm 6 DisCo3T

```
procedure DisCo3T($Const., src, dst, adjM$)                    ▷ adjM = adjacency matrix
    $H_h^*, H_v^*, H_h, H_v, SearchDir \leftarrow$ IS-MHC($Const., src, dst, adjM$)
    $\mathbf{IM}, j, l, p, q \leftarrow$ ESS($Const., src, dst, adjM, H_h, H_v, SearchDir$)
    while $H_h^* > 0$ do
        Get $\mathbf{IM}[j, l]$'s horizontal successor $\mathbf{IM}[j, l + 1]$
        Get $\mathbf{IM}[p, q]$'s horizontal successor $\mathbf{IM}[p, q - 1]$

        $Cond_s = adjM[\mathbf{IM}[j, l], \mathbf{IM}[j, l + 1]]$       ▷ 1 iff. horiz. hop available
        $Cond_t = adjM[\mathbf{IM}[p, q], \mathbf{IM}[p, q-1]]$

        if $Cond_s \wedge Cond_t$ then                ▷ true if both horizontal hops available
            Compute $reward_s$, $reward_t$ (based on latitude)
            Favor direction with higher latitude (closer to poles)
            Update corresponding route and index
            $H_h^* \leftarrow H_h^* - 1$
        else if $Cond_s \vee Cond_t$ then             ▷ true if one horizontal hop available
            Take the available horizontal hop
            Update route and index
            $H_h^* \leftarrow H_h^* - 1$
        else                       ▷ Enter this case when no horizontal hops are available
            Get $\mathbf{IM}[j, l]$'s vertical successor $\mathbf{IM}[j + 1, l]$
            Get $\mathbf{IM}[p, q]$'s vertical successor $\mathbf{IM}[p - 1, q]$
            Compute vertical hop rewards for both directions
            Take hop with higher latitude reward
            Update route and index
            $H_v^* \leftarrow H_v^* - 1$
        end if
    end while
    if $H_v^* > 1$ then
        Append remaining vertical path to $route_s$
    end if
    return $route_s \mathbin{+\!\!+} route_t$
end procedure
```

3.5 MinHopCount for Walker-Star Constellations

Walker-Star and Walker-Delta constellations differ primarily in the angular distribution of their orbital planes. In a Walker-Star configuration, the Right Ascension of the Ascending Nodes (RAANs) are evenly distributed over 180 degrees, resulting in orbital planes that are symmetrically arranged around the poles. This geometry typically corresponds to near-polar orbits with inclinations close to 90 degrees, enabling frequent coverage of high-latitude regions.

In contrast, Walker-Delta constellations distribute RAANs uniformly across the full 360 degrees, producing a more azimuthally spaced set of inclined orbits with inclinations less than 90 degrees. This configuration offers more uniform global coverage, especially in equatorial and mid-latitude regions. The difference in the satellite distribution between the two constellation types, Walker-Delta and Walker-Star, has a significant impact on the inter-plane connectivity, particularly between the satellites located in the first and last orbital planes. In Walker-Delta constellations, where the orbital planes are evenly distributed over the full 360 degrees of longitude, it is possible to establish inter-plane links between the terminal planes. However, this is not the case for Walker-Star constellations, where the orbital planes span only 180 degrees, resulting in a lack of connectivity between the first and last planes.

Consequently, the original MinHopCount model, which was specifically designed for Walker-Delta constellations, is not directly applicable to the Walker-Star architecture. To address this limitation, it becomes necessary to formulate a revised version of the MinHopCount model, explicitly tailored to capture the structural and topological characteristics of Walker-Star constellations, which we named Walker-Star-MinHopCount (WS-MHC).

3.5.1 Walker-Star-MinHopCount (WS-MHC)

Building upon the MinHopCount model originally proposed by Stock et al. in [3], we introduce a revised formulation specifically adapted to the structural characteristics of Walker-Star constellations. This new model accounts for the distinct spatial distribution and connectivity constraints inherent to Walker-Star architectures, thereby enabling accurate estimation of routing parameters, such as the number and direction of hops, within this class of constellation.

In the following, we introduce the newly proposed formulation:

- Let $L_{0,1}$ and $L_{0,2}$ denote the initial longitudes of the ascending nodes of the source and destination satellites, respectively, both defined in the interval $[0, \pi)$.

- Let $L_{0,\mathrm{max}}$ be the greater of the two longitudes of the ascending node, also defined in $[0, \pi)$, such that:

$$L_{0,max} = \begin{cases} L_{0,1} & L_{0,1} > L_{0,2} \\ L_{0,2} & \text{otherwise} \end{cases}$$

- Let $L_{0,\mathrm{min}}$ be the lower of the two longitudes of the ascending node, also defined in $[0, \pi)$, such that:

$$L_{0,min} = \begin{cases} L_{0,1} & L_{0,1} \leq L_{0,2} \\ L_{0,2} & \text{otherwise} \end{cases}$$

- Let $\Delta L_0 = (L_{0,\mathrm{max}} - L_{0,\mathrm{min}})$ be the difference between the maximum and the minimum longitudes of the ascending nodes in $[0, \pi)$.
- Being $H = \left\lfloor \frac{\Delta L_0}{\Delta \Omega} \right\rceil$ the number of horizontal hops from the minimum to the maximum longitude, where $\Delta\Omega = \pi/P$ for Walker-Star constellations.

We can then give the following conditions for the number of horizontal hops in west and east directions:

$$H_{h,s}^{\leftarrow} = \begin{cases} H & \text{if } L_{0,1} > L_{0,2} \\ \infty & \text{otherwise} \end{cases} \tag{3.2}$$

$$H_{h,s}^{\rightarrow} = \begin{cases} H & \text{if } L_{0,1} \leq L_{0,2} \\ \infty & \text{otherwise} \end{cases} \tag{3.3}$$

In Equation (3.2) and Equation (3.3), the number of horizontal hops, denoted as H, is computed and assigned to the westward direction if the ascending node angle of the source satellite is greater than that of the destination. In this case, the number of horizontal hops in the eastward direction is explicitly set to ∞. This conditional assignment ensures that the routing algorithm does not select a path requiring a transition from the last orbital plane back to the first, a connection that is not physically realizable in Walker-Star constellations, as explained above. As a result, the algorithm is constrained to consider only feasible routing directions that respect the topology of the Walker-Star configuration. Similarly, if instead the ascending node angle of the source satellite is lower than or equal to that of the destination, H is assigned to the eastward direction, while the westward is set to ∞, for the same reasons described above.

Sequentially, we adapt the formulas of the fraction $\Delta\vec{u}$ of the phase angle difference that will be covered by intra-plane hops introduced in Section 2.2.1, distinguishing between the two directions:

$$\Delta\overrightarrow{u} = \begin{cases} (u_2 - u_1 - H_{h,s}^{\rightarrow} \cdot \Delta f) \bmod 2\pi & \text{if} \quad H_{h,s}^{\rightarrow} \neq \infty \\ \infty & \text{otherwise} \end{cases} \tag{3.4}$$

$$\Delta\overleftarrow{u} = \begin{cases} (u_2 - u_1 + H_{h,s}^{\leftarrow} \cdot \Delta f) \bmod 2\pi & \text{if} \quad H_{h,s}^{\leftarrow} \neq \infty \\ \infty & \text{otherwise} \end{cases} \tag{3.5}$$

Compared to the equations presented in Section 2.2.1, Equation (3.4) and Equation (3.5) exhibit only minor differences. The primary distinction lies in the additional conditions that assign the value ∞ to the phase angle difference fractions. These conditions are triggered when the corresponding number of horizontal hops is equal to infinity. Subsequently, the number of vertical hops in all four cardinal directions can be computed following the same methodology outlined in Section 2.2.1. The only modification introduced in this context concerns the formulas associated with the southward directions, for which an absolute value is applied. This adjustment ensures that the computed number of vertical hops belongs to $[0, \infty)$ and prevents the occurrence of invalid values, such as $-\infty$:

$$H_{v,s}^{\nwarrow} = \left\lfloor \frac{\Delta\overleftarrow{u}}{\Delta\Phi} \right\rfloor \qquad H_{v,s}^{\nearrow} = \left\lfloor \frac{\Delta\overrightarrow{u}}{\Delta\Phi} \right\rfloor$$

$$H_{v,s}^{\swarrow} = \left\lfloor \left| \frac{2\pi - \Delta\overleftarrow{u}}{\Delta\Phi} \right| \right\rfloor \qquad H_{v,s}^{\searrow} = \left\lfloor \left| \frac{2\pi - \Delta\overrightarrow{u}}{\Delta\Phi} \right| \right\rfloor$$

Finally, the minimum hop count is then determined by selecting the smallest value among all feasible combinations of horizontal and vertical hops across the valid routing directions:

$$\texttt{min}\left\{H_{h,s}^{\leftarrow} + H_{v,s}^{\nwarrow}, H_{h,s}^{\leftarrow} + H_{v,s}^{\swarrow}, H_{h,s}^{\rightarrow} + H_{v,s}^{\nearrow}, H_{h,s}^{\rightarrow} + H_{v,s}^{\searrow}\right\}$$

As can be observed, the overall flow of the procedure does not differ significantly from the original MinHopCount formulation. However, the introduction of the new conditions plays a crucial role in ensuring correctness, as it effectively eliminates routing directions that would rely on non-existent inter-plane links, an inherent constraint of the Walker-Star constellation architecture. Lastly, the newly proposed model can be seamlessly integrated into the previously presented routing algorithms without requiring any modifications to the algorithms themselves. This allows for straightforward adaptation to the Walker-Star constellation architecture while preserving the original routing logic and structure.

The outcomes resulting from the application of this newly proposed model are thoroughly discussed in Section 4.5.

Chapter 4

Results and Discussion

In this chapter, we present the results obtained from the implementations detailed in Chapter 3. We begin by providing an overview of the simulation setup and the hardware environment used during the experiments, as described in Section 4.1. Subsequently, the remainder of the chapter is dedicated to the presentation and discussion of the results, with the aim of evaluating the effectiveness and performance of the proposed methodologies.

4.1 Simulation Setup

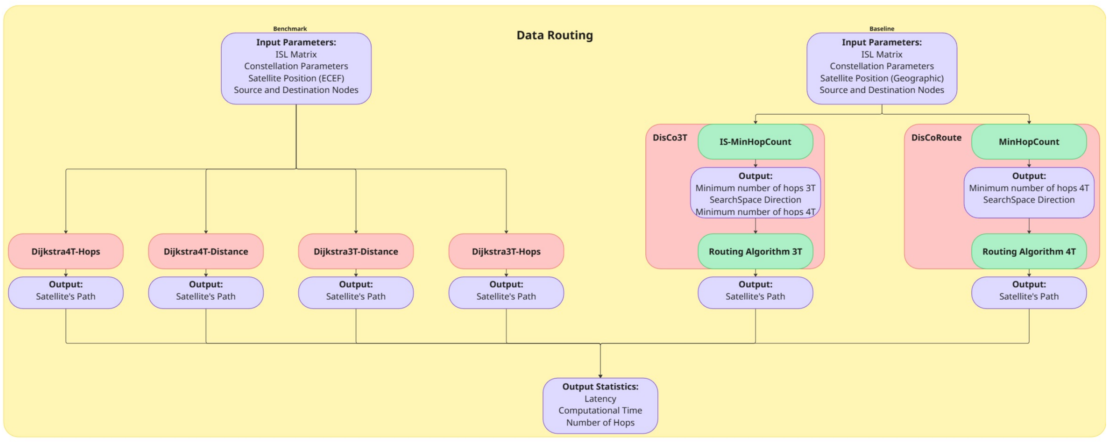


Figure 4.1: Simulator setup

In Figure 4.1, the architecture of the simulator is illustrated, clearly distinguishing between the Dijkstra-based algorithms (depicted on the left) and the

DisCo-based algorithms (on the right). Both categories of algorithms receive identical input parameters, with the sole exception being the satellite positional data. Owing to their distinct computational approaches, the two algorithm classes utilize different coordinate reference systems: Dijkstra-based algorithms operate using ECEF coordinates, as they necessitate the computation of Euclidean distances between satellites. In contrast, DisCoRoute and, by extension, DisCo3T, employ geographic coordinates.

The simulations are conducted using the following constellation configurations:

- The Walker-Delta 65° : 2080/52/10 [11] constellation for Section 4.2, Section 4.3 and Section 4.4.
- The Walker-Star 65° : 2080/52/10 constellation for Section 4.5.

The results presented from Section 4.3 to Section 4.5 are obtained by simulating a total of 10,000 source-destination satellite random pairs at a single, fixed time instant. Specifically, on March 28th, 2025, at `11:00:00.000` (UTC). In contrast, the implementation associated with Section 4.2 is based on a dynamic scenario, which has been thoroughly discussed in Section 3.1. The initialization procedures and tool configurations used across all sections follow the setup described therein.

All simulations are conducted under ideal conditions, assuming full availability of all ISLs, with no consideration given to link failures, satellite outages, or changes in the constellation configuration.

Subsequently, the algorithms are executed, generating the computed paths as output. From these paths, we extract the statistics of interest:

- **Latency**, which measures the time required to traverse the path.
- **Computational Time**, which is crucial for evaluating the efficiency of the DisCo algorithms in comparison to state-of-the-art methods such as Dijkstra's.
- **Number of Hops**, which, when analyzed alongside latency, provides insight into the performance of algorithms that optimize for hop count, such as the DisCo variants and Dijkstra-Hop, relative to those that optimize for different objectives, such as Dijkstra-Distance. This also helps assess the influence of network topology on algorithmic performance.

The experiments were conducted to evaluate the performance of the routing algorithms developed using `MATLAB R2023b`. The analysis was based on the creation of satellite scenarios utilizing the default orbit propagator available within the Aerospace Toolbox for Satellite Mission Analysis, in conjunction with the Satellite Communication Toolbox. A proximity-based strategy was employed to determine which ISLs (Inter-Satellite Links) should be established, based on the Walker-Delta constellation pattern. This method allowed for the dynamic configuration of communication links with neighboring satellites according to their spatial proximity. The entire experimental setup was executed on a Lenovo IdeaPad S540 laptop, equipped with an AMD Ryzen™ 5 processor and 20 GB of RAM, ensuring adequate computational resources for scenario simulation and algorithmic evaluation.

Table 4.1 provides a summary of the algorithms analyzed throughout this study, including their optimization objectives and the search methods.

Algorithm	**Optimization**	**Search method**
DisCo4T	Number of Hops	MinHopCount
Dijkstra	Distance length (and thus latency)	Exhaustive Search
DijkstraHop	Number of Hops	Exhaustive Search
DisCo3T	Number of Hops	IS-MHC
Dijkstra3T	Distance length (and thus latency)	Exhaustive Search
DijkstraHop3T	Number of Hops	Exhaustive Search

Table 4.1: Comparison of routing algorithms, their optimization goals, and search methods.

Notation on the visualizations

In the following analysis, two primary statistical visualizations are employed:

- **Violin plot**: A density-oriented visualization that combines kernel density estimation with a mirrored display, offering a comprehensive representation of the data distribution. In addition to the distribution, vertical bars are shown to indicate the minimum, mean, and maximum values of the support.
- **Box plot**: A compact graphical representation of the distribution of the data that depicts the median, the mean, interquartile range, the minimum and the maximum values, and potential outliers, facilitating rapid comparison of central tendency and dispersion.

4.2 Time varying DisCoRoute

In this section, we present a comprehensive analysis of the results derived from the implementation of the DisCoRoute algorithm under dynamic simulation conditions in the 4T scenario, as detailed in Section 3.1. Within this framework, we evaluate the algorithm's performance according to three principal metrics, as defined in the aforementioned section:

1. **Latency Plot**: Violin plots showing the distribution of E2E delay at each simulation step, based on all computed routes. These visualize the algorithm's latency performance over time.

2. **Standard Deviation Plot**: A line plot tracking the latency standard deviation for each source-destination pair throughout the simulation, reflecting the stability and variability of routing performance.

3. **Path Change Frequency**: A bar chart displaying the percentage of routes that changed or remained stable at each time step, highlighting the algorithm's adaptability to dynamic topology changes.

4.2.1 Latency over Time

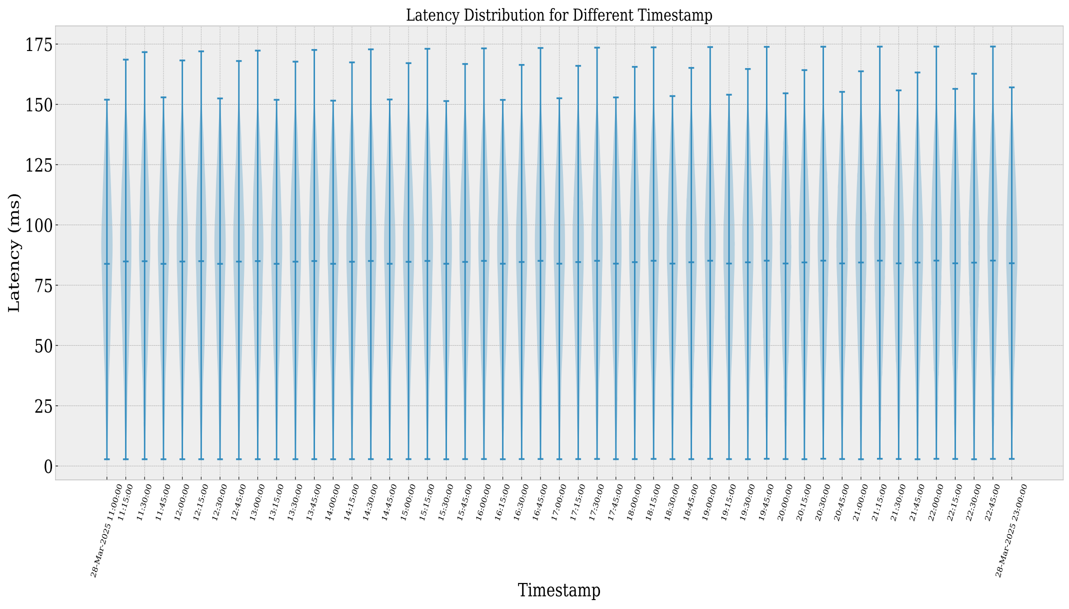


Figure 4.2: Latency distributions over time

Figure 4.2 illustrates the distributions of E2E latency for each simulation time instant. The dataset spans from March 28th, 2025 at `11:00:00` (UTC) to the same day at `23:00:00` (UTC), revealing a clear periodic variation in the latency distributions over time. Among the various statistical metrics, the most affected are the variance and, consequently, the maximum observed E2E delay.

Specifically, when compared to the initial simulation time, the latency variance increases by 12.58 % at the second time instant, and by 14.57 % at the third. At the fourth time instant, the variance returns to levels comparable to those observed at the beginning of the simulation. This cyclical pattern continues across subsequent time instants.

Nevertheless, a deviation from this pattern is observed starting at `17:00:00`. Although periodicity would suggest a return to the initial variance levels, the measured variance continues to increase. Similarly, the following two time instants, expected to replicate the variance levels of the second and third intervals, display decreasing and increasing trends, respectively.

This behavior can be explained by the orbital mechanics of the constellation under study. The orbital period of the satellites is approximately 91 minutes. Given a simulation sampling interval of 15 minutes, the satellite positions repeat approximately every hour. However, since the orbital period is not an exact multiple of the sampling interval, the positions gradually diverge with each cycle. Over time, these small discrepancies accumulate, eventually leading to observable differences in satellite positions and, consequently, in the latency distribution.

Finally, it is worth noting that while the variance of latency exhibits significant periodic fluctuations, the average latency values remain comparatively stable. These averages also follow a periodic trend of increase and decrease, though with less pronounced variation, typically within a 1.2 % range.

The standard deviation of the E2E delays over the entire simulation time interval, computed for each simulated satellite pair, is presented in Figure 4.3. This visualization clearly highlights that the variations in latency experienced by individual node pairs are not uniform across the constellation. Each source-destination pair exhibits distinct fluctuations in delay, primarily influenced by their relative geographical positions.

To provide a more in-depth analysis, we examine both the best-case and worst-case satellite pairs in terms of the standard deviation of the E2E latency observed over the entire simulation period:

- **Best Case**: The satellite pair exhibiting the lowest standard deviation in latency is [`src dst`] = [2049 2076]. Analysis of the computed routes throughout the simulation reveals that these two satellites, positioned only one orbital plane apart, consistently reside in opposing propagation directions. Specifically, the source in a descending direction and the destination in an

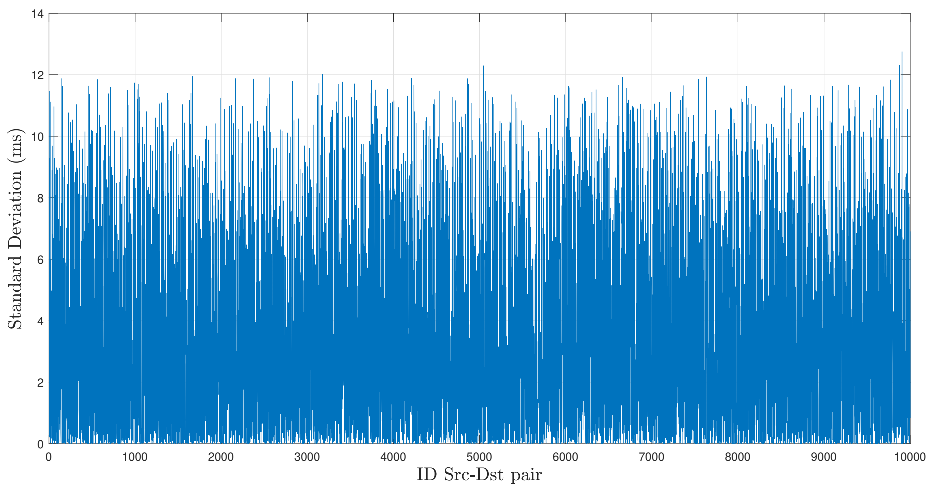


Figure 4.3: Standard deviation on the latency for each source-destination pair

ascending direction. As a result, DisCoRoute consistently applies the routing logic defined in Algorithm 3. In this scenario, the sole horizontal hop invariably occurs near the polar regions, which is the only feasible location for such hops in the A2D case. Since intra-plane hops, constituting the majority of the path, are of fixed duration, and the inter-plane hops near the poles involve minimal inter-satellite distances, the overall latency remains effectively constant across time. Consequently, this pair yields the lowest observed standard deviation, approximately 0 ms.

- **Worst Case**: The satellite pair with the highest observed standard deviation in latency is [`src` `dst`] = [445 1397], registering a standard deviation of 12.75 ms. In this case, both satellites propagate in the ascending direction and are separated by 24 orbital planes, nearly the maximum possible horizontal separation in the constellation, which corresponds to 26 horizontal hops. Due to the high number of horizontal hops required in their communication path, the pair's latency becomes highly sensitive to their relative geographical positions. This sensitivity is particularly pronounced when horizontal hops occur near the equator, where the distance between adjacent planes is maximal. As a result, the path latency for this pair experiences significant variability across different time instants, making it the worst-case scenario in terms of latency stability.

Based on the observations made regarding the worst-case scenario, we extended our investigation to encompass all satellite pairs exhibiting the highest standard deviations in E2E latency. The analysis confirmed that the pairs experiencing the greatest variability are those requiring a larger number of horizontal hops to establish connectivity.

These findings underscore a critical limitation of the DisCoRoute algorithm. Although the algorithm consistently aims to minimize the overall path length, its search is inherently constrained by the fixed number of hops and the propagation direction dictated by the MinHopCount model. As a result, it is compelled to follow a predetermined directional search for each source-destination pair, regardless of dynamic satellite positioning. Consequently, under varying satellite positions over time, the algorithm is more likely to produce suboptimal routes, particularly in cases involving extensive horizontal traversal. This leads to higher fluctuations in latency, as evidenced by the increased standard deviations observed in such scenarios.

4.2.2 Path Change Frequency

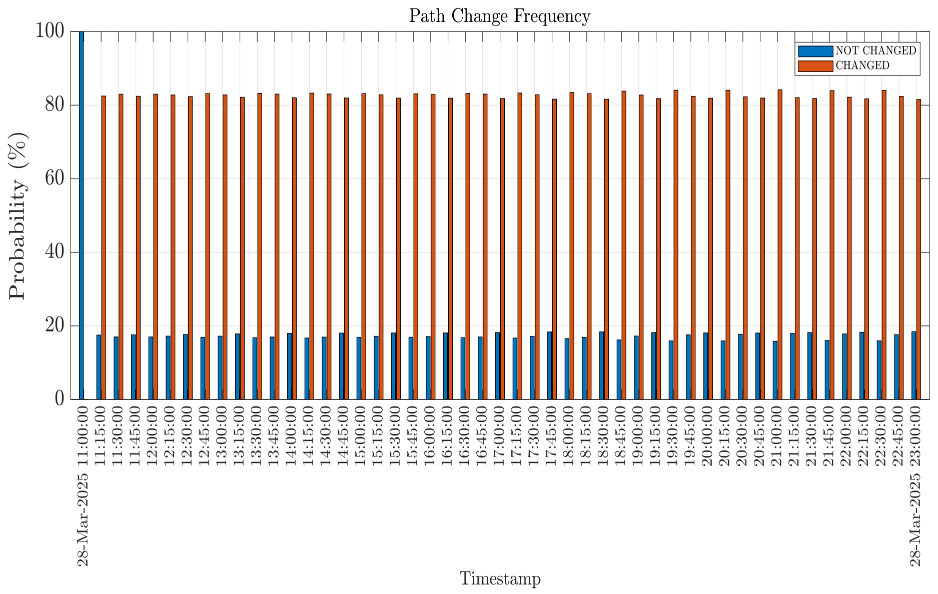


Figure 4.4: Path changes over time

Figure 4.4 illustrates the temporal evolution of routing decisions for each satellite pair at every simulation time step. The decisions are classified into two categories: `CHANGED`, indicating that a new routing path was selected compared to the previous interval, and `NOT CHANGED`, signifying that the same routing path was retained. The vertical axis of the figure reports the relative probability of each category across the time steps.

Excluding the initial time instant, during which no prior routing information is available, the remaining time points reveal a consistent behavioral pattern: over 80 % of the evaluated satellite pairs modify their routing paths between consecutive intervals. This high frequency of change is a direct consequence of DisCoRoute's strategy, which aims to minimize E2E latency by recalculating routes at each time step based solely on the current constellation geometry. The algorithm's responsiveness to topological changes demonstrates its ability to adapt effectively to dynamic network conditions, continuously searching for latency-optimal paths.

However, as previously discussed, this adaptability is subject to inherent limitations. Although frequent path updates reflect an algorithmic attempt to optimize latency, they do not necessarily yield a stable latency distribution over time. On the contrary, they may contribute to the variability in delay observed in Figure 4.2, particularly in the worst-case scenarios where latency increases significantly. These fluctuations could adversely impact the quality of service for end users.

Finally, it is important to emphasize that the results of this path change analysis are sensitive to the choice of the simulation sample time. A shorter sample interval would result in fewer routing changes, as satellite positions would vary less between consecutive time steps, reducing the likelihood of different routing decisions by DisCoRoute.

4.3 Adaptation of DisCoRoute to 3T scenario

In this section, we present the results obtained from the implementation of the initial adaptation of the DisCoRoute algorithm to the 3T scenario, as discussed in Section 3.2.2. The performance evaluation is conducted with respect to three principal metrics: end-to-end latency, number of hops, and computational execution time. The analysis is structured around two comparative assessments. First, we compare the performance of the newly proposed 3T heuristic against Dijkstra-based benchmark algorithms operating under the same 3T topology. Second, we contrast these outcomes with those derived from the 4T configuration. This dual comparison enables us to assess both the effectiveness of the DisCoRoute algorithm and the underlying MinHopCount model in constrained topologies, as well as the broader impact of the 3T configuration on the performance of the various routing approaches. Throughout this section, the heuristic developed for the initial adaptation of DisCoRoute to the 3T topology will be referred to as DisCo3T.

4.3.1 Latency

From the perspective of end-to-end latency, the results presented in Figure 4.5 clearly highlight the sub-optimal nature of the newly proposed heuristic, in the figure referred to as DisCo3T. In particular, when comparing this heuristic to the 3T variants of the Dijkstra-based algorithms, we observe a significant performance gap. Specifically, the latency of the heuristic exceeds that of the worst-case Dijkstra3T by approximately 66 %, and that of the worst-case DijkstraHop3T by approximately 50.91 %. Even when considering average performance, the discrepancy remains notable, with the heuristic exhibiting an average latency increase of 24.55 % relative to Dijkstra3T, and 19.13 % when compared to DijkstraHop3T. These performance deviations can be primarily attributed to the lack of a geometric model capable of precisely estimating the number of hops required to complete a path between two satellites within the 3T configuration. This modeling limitation significantly impacts the latency performance of the heuristic. Given these findings, the adoption of this heuristic in latency-sensitive scenarios would be strongly discouraged, as it not only results in increased end-to-end delay but also introduces greater variance in latency measurements. Nevertheless, it is important to emphasize that the main objective of this task was to assess the feasibility of adapting the DisCoRoute framework to operate within a 3T satellite topology. Despite the considerable performance gap observed, the results conclusively demonstrate that such an adaptation is indeed achievable.

As a preliminary step in evaluating the impact of the 3T satellite topology on routing performance, we conduct a comparative analysis of Dijkstra's algorithm in both the 3T and 4T configurations. The results reveal a substantial increase in average latency when employing the 3T topology. In particular, we observe an average latency rise of 34.14 % relative to the conventional 4T architecture. This increase provides a concrete and quantitative assessment of the performance cost associated with reducing the number of inter-satellite link terminals from four to three. The degradation in latency performance is largely attributable to the diminished availability of ISLs, which in turn restricts routing flexibility and significantly reduces the diversity of viable routing paths across the constellation.

A more detailed discussion of the implications of this architectural shift, along with an in-depth analysis of its effects on heuristic performance, is provided in Section 4.4.

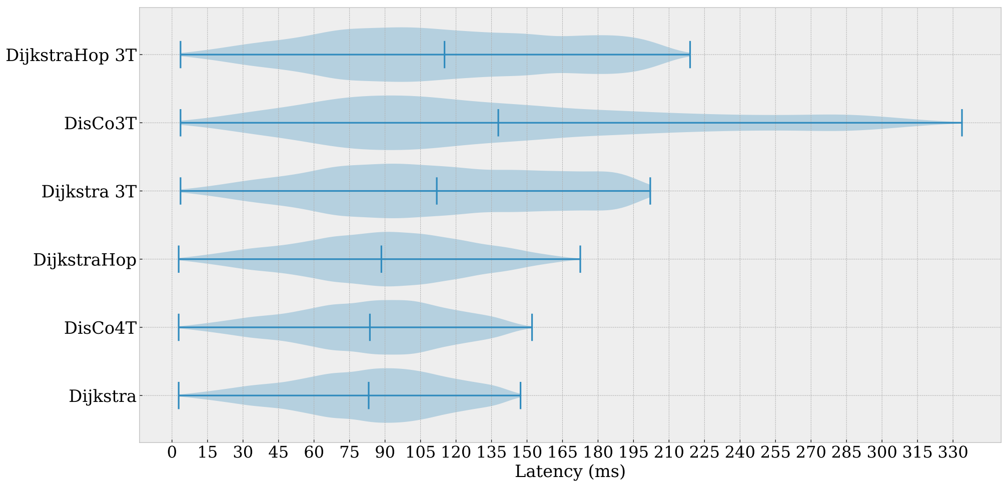


Figure 4.5: Latency distributions comparison among the tested algorithms

4.3.2 Hops Count

As previously outlined in the preceding section, the initial version of the new heuristic, DisCo3T, demonstrates several limitations when applied to the 3T topology, as evidenced in Figure 4.6. Chief among these is the absence of a MinHopCount model specifically designed for the 3T configuration, which critically undermines the algorithm's ability to estimate the number of hops required to establish a connection between two satellites.

This modeling deficiency results in a significantly higher variance in the number of hops, as compared to the 3T adaptations of the Dijkstra-based algorithms, namely Dijkstra3T and DijkstraHop3T, which consistently converge to identical and more stable distributions. In contrast, DisCo3T exhibits considerable performance gaps, quantified as follows:

- An average increase of 21.43 % in hop count compared to both Dijkstra3T and DijkstraHop3T.
- A worst-case increase of 55.77 % in hop count relative to the same Dijkstra-based benchmarks.

These findings underscore the pressing need for a MinHopCount model that is explicitly tailored to the constraints and structural characteristics of the 3T topology. In response to this need, we introduced the IS-MHC model, described in detail in Section 3.3.1, which is integrated into the final version of the DisCo3T algorithm presented in Section 3.4. The results of this enhanced implementation are thoroughly analyzed and discussed in Section 4.4.

Finally, it is worth highlighting that, in the case of the 4T configuration, the Dijkstra-based algorithms, Dijkstra and DijkstraHop, exhibit distinct path distributions. These differences stem from the distinct optimization criteria each algorithm employs: one minimizes latency, while the other focuses on hop count. However, in the 3T configuration, a notable convergence is observed between Dijkstra3T and DijkstraHop3T, which both yield identical distributions, as previously noted.

This convergence suggests that, within the constraints imposed by the 3T satellite topology, the path with the minimum number of hops coincides with the path of minimum latency. This finding is supported by [8], where it is similarly argued that in topologies characterized by reduced inter-satellite connectivity, the shortest feasible path is also the minimum-hop path. This insight is of particular significance in the validation of the proposed IS-MHC model, as further elaborated in Section 4.4.2.

4.3.3 Computational Time

In this subsection, we present a comparative analysis of the computational efficiency of the proposed heuristic, DisCo3T, against the Dijkstra3T algorithm. Additionally, we extend the comparison to include the original 4T configuration, evaluating the performance of DisCoRoute (DisCo4T) against the classical Dijkstra implementation. The results of this analysis are illustrated in Figure 4.7. As evidenced by the data, the DisCo3T heuristic demonstrates a significant computational advantage, requiring, on average, only 1.24 % of the execution time consumed by Dijkstra3T.

Furthermore, the distribution of computational time for DisCo3T closely resembles that of its predecessor, DisCoRoute, which was originally designed for the 4T topology. In particular, DisCo3T exhibits a tighter variance and more compact interquartile range than the 4T implementation, despite a marginally

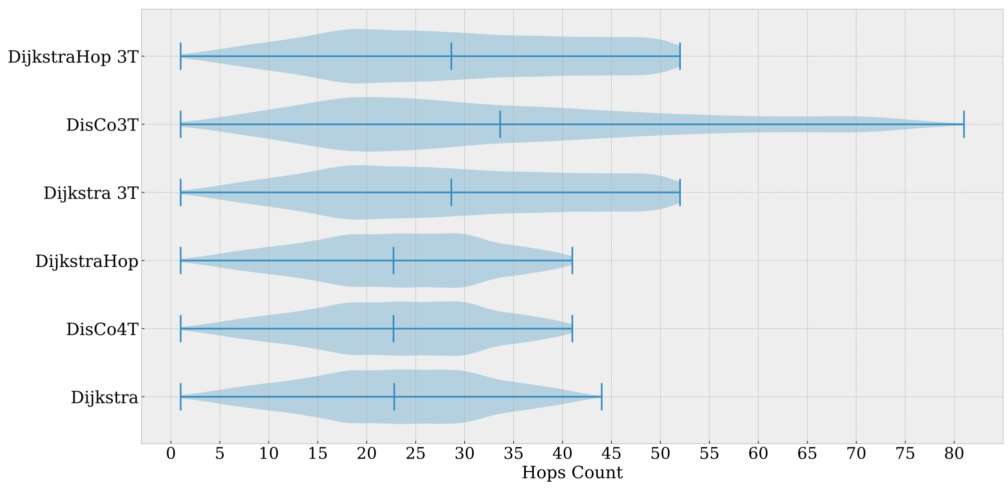


Figure 4.6: Hops count distributions comparison between the tested algorithms and DisCoRoute in 3T

higher mean execution time. This behavior is primarily attributable to the use of the MinHopCount model, which provides an efficient geometric estimation of path cost, as well as the structure of the routing algorithm employed in DisCo3T. Notably, when only a single horizontal hop is available, an edge case addressed in Section 3.2.2, the algorithm follows a simplified path expansion logic, further reducing computational overhead. Nevertheless, the slightly increased average execution time in the 3T scenario compared to the 4T version can be explained by the inherent characteristics of the 3T topology. Specifically, the reduced number of inter-satellite links necessitates longer paths (i.e., a higher number of hops) to traverse from a source to a destination node, thereby introducing additional processing steps in the route computation. Overall, these findings reinforce the computational viability of DisCo3T in constrained environments, while underscoring the trade-offs introduced by different topological configurations.

It is important to clarify that, although not explicitly reported in this section, the execution times of Dijkstra3T and DijkstraHop3T are essentially identical. This equivalence stems from the fact that the two algorithms share nearly identical implementation structures and computational logic, differing primarily in the metric used for path evaluation rather than in the search mechanism itself.

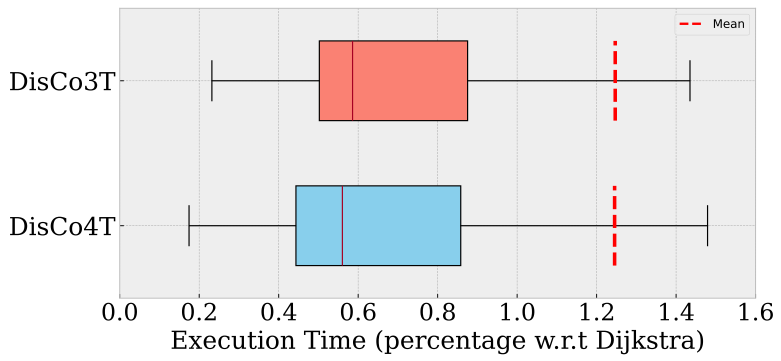


Figure 4.7: Execution time percentage of DisCoRoute and DisCo3T with respect to Dijkstra and Dijkstra3T

4.4 DisCo3T

In this section, we present the results obtained from the implementation of the proposed Interleaved Scenario-MinHopCount (IS-MHC) model and the newly developed routing algorithm DisCo3T, as detailed in Section 3.3.1 and Section 3.2, respectively. Adopting the same evaluation methodology employed in the previous section, we assess the performance of the proposed solution using three principal metrics: end-to-end latency, total number of hops, and computational execution time. Subsequently, we compare these results with those obtained under the 4T configuration, in order to evaluate the impact of the 3T topology on overall routing performance.

4.4.1 Latency

From a latency perspective, the results illustrated in Figure 4.8 reveal a significant increase when utilizing the 3T topology. Specifically, there is a 34.14 % rise in average latency when compared to the traditional 4T configuration. This observation provides a quantitative understanding of the performance cost incurred when transitioning from a four-terminal inter-satellite link (ISL) architecture to a three-terminal design. The increase is primarily attributed to reduced link availability, which limits routing flexibility and path diversity.

Focusing on the performance of the proposed DisCo3T heuristic under the 3T configuration, the results demonstrate that it achieves near-optimal performance relative to the Dijkstra3T baseline. On average, DisCo3T exhibits a latency gap of only 2.5 %, with the maximum observed deviation reaching 9.79 %. These outcomes are particularly encouraging, considering the reduced connectivity inherent in the 3T setup.

For comparison, the DijkstraHop3T algorithm exhibits a slightly higher average performance gap of 2.88 % and a notably larger worst-case deviation of 14.37 % relative to Dijkstra3T.

These findings yield two important implications for the design and operation of VLEO satellite constellations:

1. **Architectural Trade-offs:** The migration from a 4T to a 3T terminal configuration introduces a non-negligible latency overhead, averaging over 34 %. This underscores the inherent trade-off between architectural simplicity (fewer terminals, reduced cost, and complexity) and overall network performance. Therefore, such design decisions must be carefully balanced during the constellation planning phase, especially for applications sensitive to end-to-end delay.

2. **Heuristic Robustness under Constraints:** Despite the structural limitations imposed by the 3T topology, the DisCo3T heuristic exhibits highly competitive performance. Its ability to consistently approximate the optimal latency paths computed by Dijkstra3T, with minimal performance degradation, validates the effectiveness of the IS-MHC-based directional guidance used in its design. Furthermore, the superior results attained by DisCo3T compared to DijkstraHop3T highlight the advantage of integrating geometric insight into the routing logic, rather than relying solely on hop count minimization. This positions DisCo3T as a promising and efficient alternative for real-time, low-complexity routing in VLEO networks subject to hardware constraints.

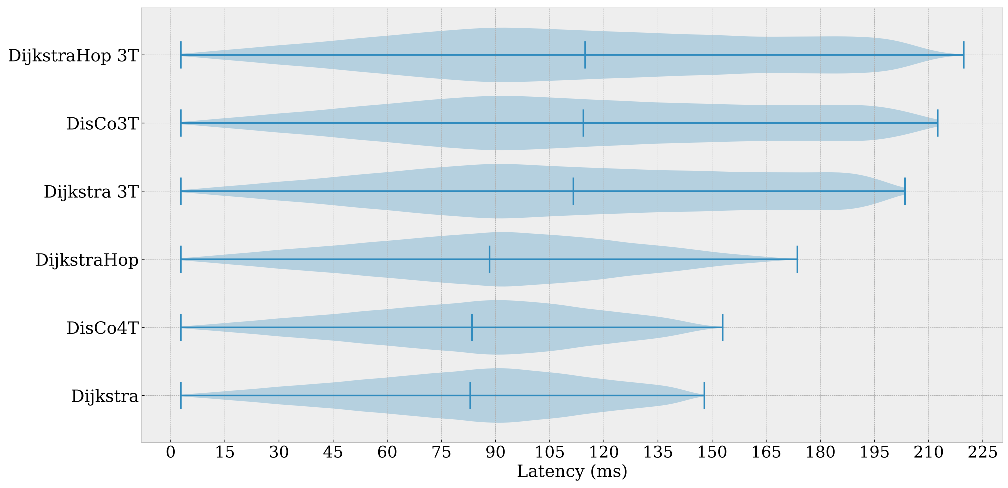


Figure 4.8: Latency distributions comparison between the tested algorithms and DisCo3T

4.4.2 Hops Count

In addition to latency, the effect of the 3T topology is also clearly observed in terms of the total number of hops required to complete end-to-end communication. Specifically, the results in Figure 4.9 indicate an average increase of 21.7 % in the number of hops when compared to the DisCoRoute heuristic operating under the 4T configuration. This increase underscores the direct consequence of reduced link availability on the efficiency of routing paths.

Despite this increase, a noteworthy observation emerges within the 3T scenario: all three algorithms under evaluation, DisCo3T, Dijkstra3T, and DijkstraHop3T, exhibit identical hop count distributions. This uniformity is attributed to the structural nature of the interleaved mesh topology, which enforces a deterministic correspondence between the shortest geometrical path and the minimum-hop route. These outcomes serve as strong empirical evidence supporting the validity and completeness of the proposed IS-MHC model.

The above findings yield several key implications for satellite network design and the evaluation of routing strategies:

1. **Structural Impact of the 3T Topology:** The increase in hop count by approximately 21.7 % highlights the inherent limitations introduced by the three-terminal configuration. With fewer available inter-satellite links, routing paths are often forced to traverse more satellites using stair-like trajectories, thereby increasing path lengths and, consequently, the number of intermediate hops.

2. **Model Validation:** The consistent and overlapping hop count distributions across all evaluated algorithms within the 3T configuration validate the theoretical accuracy of the IS-MHC model. Despite reduced connectivity, the heuristic successfully identifies the minimum-hop path, achieving parity with exhaustive approaches such as Dijkstra3T. This demonstrates both the model's robustness and its suitability for constrained topologies.

3. **Comparison with the 4T Case:** In contrast, routing solutions in the 4T configuration exhibit significantly tighter hop count distributions, centered around lower values. This reflects the greater degree of freedom provided by the additional cross-plane ISL, which enables more direct and efficient routing paths.

4.4.3 Computational Time

In this subsection, we provide a comparative evaluation of the computational performance of the proposed DisCo3T heuristic against the Dijkstra3T algorithm. As depicted in Figure 4.10, the DisCo3T approach demonstrates a significant advantage in terms of execution time, requiring on average only around 3.56 % of the computational time consumed by Dijkstra3T. This substantial reduction in runtime underscores the potential of DisCo3T to deliver near-optimal routing performance with markedly reduced computational overhead, thereby offering a highly efficient alternative for real-time routing in resource-constrained satellite network environments.

DisCo3T, while designed to approximate Dijkstra3T with lower complexity, experiences a notable increase in computational cost when compared to its predecessor, DisCoRoute, developed for the 4T scenario. Specifically, the execution time

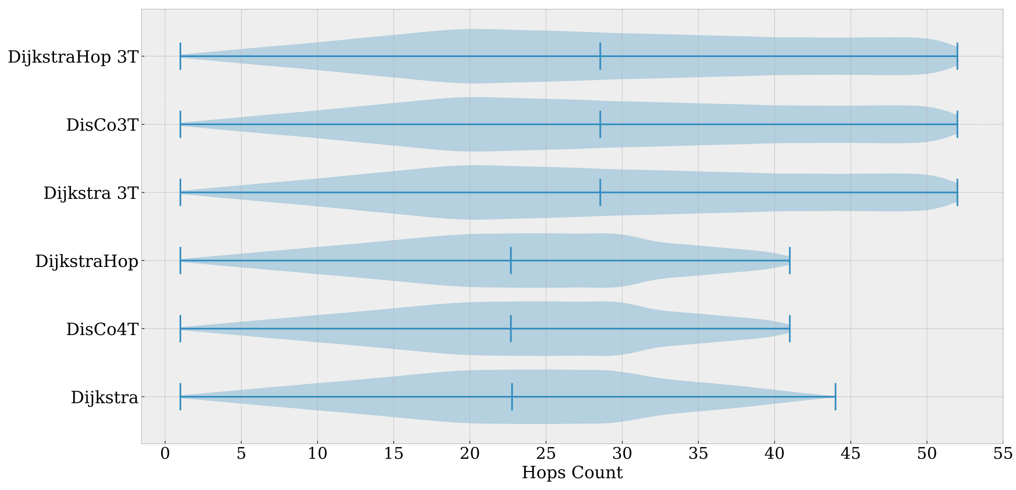


Figure 4.9: Hops count distributions comparison between the tested algorithms

of DisCo3T has more than doubled relative to the 4T version. This increase is primarily attributed to the added structural and algorithmic complexity introduced by the IS-MHC model, which replaces the simpler MinHopCount strategy previously employed.

As highlighted in Section 4.3.3, although not explicitly reported in this section, the execution times of Dijkstra3T and DijkstraHop3T are essentially identical. Consequently, in the comparative analysis that follows, we limit our attention to evaluating the performance of the DisCo-based heuristics relative to their corresponding Dijkstra-Distance variants, the latter being the algorithm yielding optimal solutions with respect to end-to-end latency and, in the context of the 3T topology, also deliver the minimal number of hops path.

4.4.4 Search Space Analysis

Based on the results previously discussed, it was observed that the DisCo3T algorithm and the Dijkstra3T algorithm exhibit identical distributions in terms of hop count. However, a noticeable discrepancy arises in their corresponding end-to-end latencies. The only plausible cause for this divergence lies in the manner in which each algorithm constructs its routing paths, highlighting the intrinsic differences in their respective routing strategies.

In light of this, we undertake a detailed comparison of the path discovery mechanisms employed by the two algorithms, with the objective of refining our heuristic in order to more closely approximate the performance and routing behavior

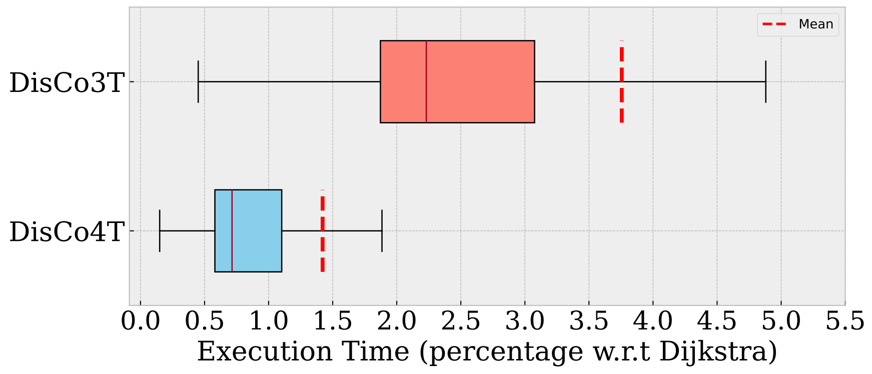


Figure 4.10: Execution time percentage of DisCoRoute and DisCo3T with respect to Dijkstra and Dijkstra3T

of the Dijkstra-based approach.

The comparative analysis is illustrated through the results presented in Figure 4.11, Figure 4.12, and Figure 4.13.

Upon examination, we find that in approximately 32% of the 10,000 iterations conducted, Dijkstra3T opted for a different route exploration strategy compared to DisCo3T. A closer inspection of Figures 4.12 and 4.13 reveals the following key observations:

1. The DisCo3T algorithm demonstrates a greater tendency to identify routing solutions that remain within the originally anticipated search region, compared to Dijkstra3T.

2. In contrast, Dijkstra3T more frequently explores outside the original search region compared to DisCo3T, and neither of the two algorithms exhibits a strong preference for any particular direction in this out-of-bound search.

3. The DisCo3T algorithm is more likely to perform bidirectional exploration, that is, it more frequently examines areas both above the source node and below the destination node, relative to Dijkstra3T. This behavior is especially pronounced in scenarios where the number of columns significantly exceeds the number of rows, i.e., when $m \ll n$. In particular, DisCo3T constructs two path segments originating from the source and the destination nodes, each following the directions suggested by the IS-MHC model. However, if the final node reached by the destination-initiated path lies above the source node and is within the remaining horizontal hop budget, the direction of expansion may be reversed. When $m \ll n$, this bidirectional growth often causes the two path segments to diverge, exploring entirely different areas of the network topology, regardless of the resulting path's efficiency. This phenomenon is

especially evident in routing paths composed exclusively of horizontal hops and can lead to notably suboptimal outcomes.

4. In a small but non-negligible subset of cases (approximately 2.4% of all iterations), Dijkstra3T explores a completely different region of the network compared to DisCo3T. These cases are denoted as *"NOT COMPARABLE"* in Figures 4.12 and 4.13.

Taken together, these insights strongly suggest that there remains substantial room for improvement in the current heuristic. By addressing the identified suboptimal behaviors, it should be possible to significantly enhance the performance of the DisCo3T algorithm, especially in terms of reducing latency while preserving its computational efficiency.

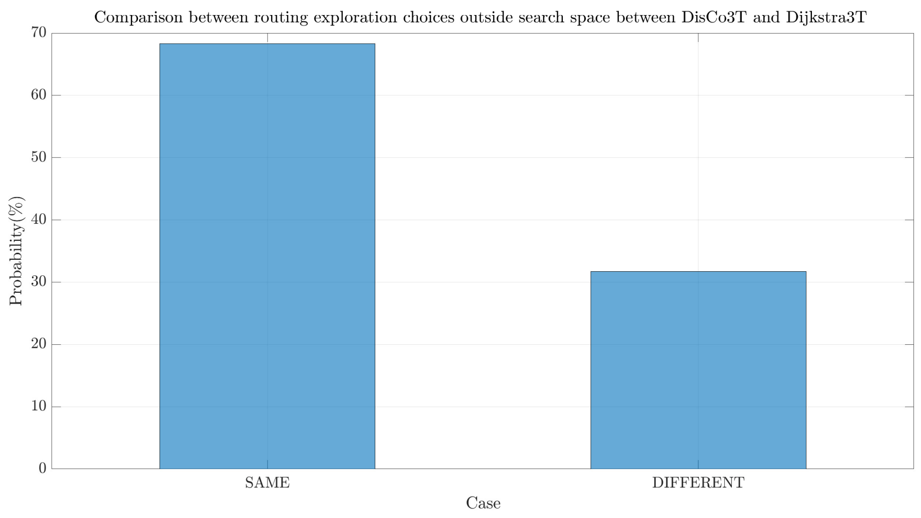


Figure 4.11: Exploration direction comparison between DisCo3T and Dijkstra3T

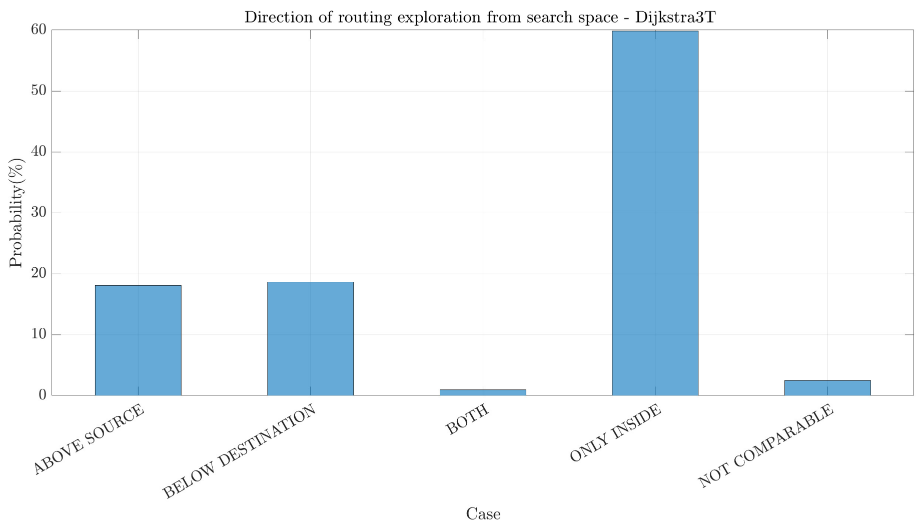


Figure 4.12: Exploration directions of Dijkstra3T

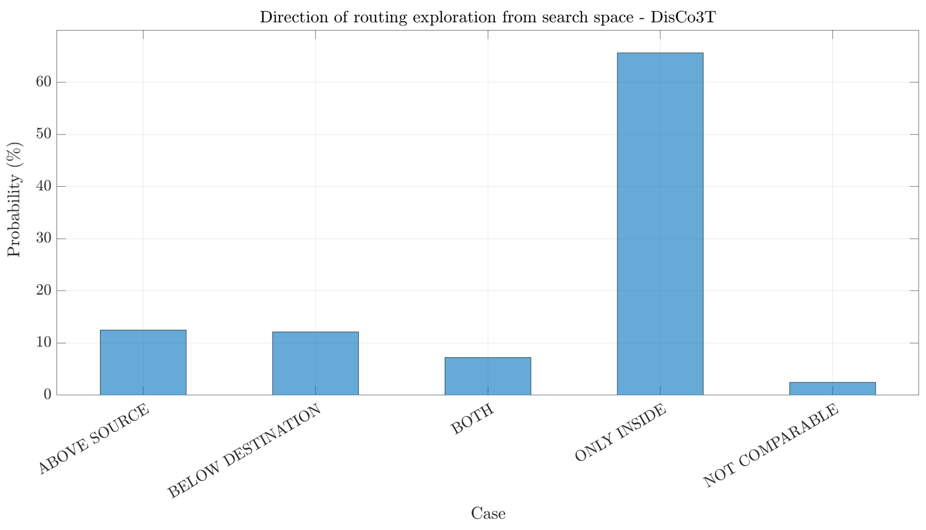


Figure 4.13: Exploration directions of DisCo3T

4.5 MinHopCount for Walker-Star Constellations

In this section, we present the results of integrating the proposed WS-MHC (Walker-Star-MinHopCount) model into the DisCoRoute routing algorithm, as detailed in Section 3.5.1. First, the performance of the previously developed models and algorithms is evaluated in the context of the Walker-Star constellation, in order to assess their feasibility for deployment within this type of configuration. Then, following the methodology established in previous sections, we evaluate the performance of the new proposed model across three key metrics: end-to-end latency, hop count, and execution time. The analysis is structured around two comparative assessments. First, comparing DisCoRoute with Dijkstra-based benchmarks under the Walker-Star constellation; second, contrasting these results with those from the Walker-Delta configuration. This two-tiered evaluation allows us to assess both the effectiveness of DisCoRoute under constrained topologies and the advantages introduced by the WS-MHC model. It is important to emphasize that the following analysis is conducted entirely within the context of a 4T topology. Furthermore, in the figures presented, the labels `DisCo-Delta` and `DisCo-Star` refer specifically to the DisCoRoute algorithm as described in detail in Section 2.2.2.

4.5.1 Impact of Walker-Star constellation

In the early stages of the thesis, we performed an additional experimental evaluation aimed at assessing the impact of a Walker-Star constellation on the routing algorithms employed in our previous simulations. The characteristics of this type of constellation are discussed in Section 3.5. In particular, the analysis highlights the implications of the inability to establish inter-satellite links between the first and last orbital planes, which stems from the distinct distribution pattern of the satellites' orbital planes. Unlike Walker-Delta constellations, which distribute their planes uniformly over a 360° range, the Walker-Star configuration confines this distribution to only 180°.

The findings of this analysis are summarized as follows:

- In the original Walker-Delta constellation topology, no routing failures were observed across any of the algorithms considered, neither the Dijkstra-based algorithms nor the DisCo-based heuristics, regardless of whether the network employed a 3T or 4T configuration.

- In contrast, under the Walker-Star constellation, the DisCo-based heuristics (DisCoRoute and DisCo3T) consistently failed to find valid routes in scenarios where the shortest path would have required traversing the now-absent cross-links between the first and last planes. Meanwhile, the Dijkstra-based algorithms (Dijkstra, DijkstraHop, Dijkstra3T, DijkstraHop3T) maintained full routing capability and succeeded in identifying feasible paths in all such cases.

This discrepancy can be directly attributed to the fundamental differences in the underlying mechanisms of the two algorithmic families. Dijkstra-based algorithms employ an exhaustive graph search strategy, systematically exploring all possible paths through the constellation. This approach inherently adapts to any topological variation, including missing links, ensuring optimal path discovery. Conversely, the DisCo heuristics rely on the MinHopCount model, which estimates the number of hops necessary to traverse the network based on geometrical assumptions tailored for the Walker-Delta configuration. Since the model presumes the existence of complete circular connectivity between orbital planes, it remains unaware of the structural absence of inter-plane links in the Walker-Star setting. As a result, the model may propose propagation directions that are invalid under the actual topology, inevitably leading to routing failures for certain source-destination satellite pairs.

These observations clearly highlight the need for an adaptation of the MinHop-Count model to account for the topology of Walker-Star constellations. To address this, we developed the WS-MHC model, which is discussed in detail in Section 3.5.1, and whose results are provided in next subsections.

Furthermore, a quantitative comparison of the routing success rates across the evaluated algorithms, under both Walker-Delta and Walker-Star constellation topologies, is presented in Table 4.2. The failure rates are omitted being `failure rate% = 100% - success rate%`. For clarity and consistency, the algorithm names have been annotated with the suffixes "`-Delta`" and "`-Star`" to denote their application within the Walker-Delta and Walker-Star contexts, respectively.

Finally, it is worth noting that, from an implementation perspective, this analysis required only a modification to the type of constellation generation function used in MATLAB. Specifically, the `walkerStar` function [15] was employed in place of the previously adopted `walkerDelta` function [12].

In the following subsections, the results obtained for the proposed WS-MHC model are presented and discussed.

4.5.2 Latency

Since the integration of the WS-MHC model into the DisCoRoute algorithm resulted in no routing failures throughout the simulation, it is now appropriate to proceed with a detailed evaluation of the obtained performance outcomes. The results illustrated in Figure 4.14 provide a clear demonstration of the performance implications associated with adopting a Walker-Star constellation architecture as opposed to the traditional Walker-Delta configuration. In particular, the comparative evaluation of DisCoRoute against Dijkstra-based routing algorithms reveals

Routing algorithm	**Probability of success (%)**
Dijkstra-Delta	100
DijkstraHops-Delta	100
DisCoRoute-Delta	100
Dijkstra3T-Delta	100
DijkstraHops3T-Delta	100
DisCo3T-Delta	100
Dijkstra-Star	100
DijkstraHops-Star	100
DisCoRoute-Star	75
Dijkstra3T-Star	100
DijkstraHops3T-Star	100
DisCo3T-Star	75

Table 4.2: Comparison of the routing success and failure rates across the evaluated algorithms, under both Walker-Delta and Walker-Star constellation topologies

several important observations:

1. The proposed heuristic DisCoRoute, when adapted to utilize the WS-MHC model, continues to deliver near-optimal performance. Specifically, it exhibits an average latency deviation of only 1.20 % relative to the baseline Dijkstra-Distance algorithm, with a worst-case gap limited to 4.95 %. Both algorithms exhibit closely aligned latency distributions, although DisCo3T is characterized by a slightly greater variance in the observed measurements.

2. Furthermore, DisCoRoute consistently outperforms the DijkstraHop algorithm. The latter shows an average latency gap of 5.95 % and a worst-case deviation of 4.72 % when compared to DisCoRoute.

3. A comparative analysis of the latency distributions under both the Walker-Delta and Walker-Star configurations reveals that, although the average latency values exhibit only marginal increases, approximately 1.22 % for both DisCoRoute and Dijkstra-Distance, and negligible variation for DijkstraHop, the more pronounced impact lies in the increased variability of the results. Specifically, the worst-case latency values increase by 38.36 % for Dijkstra-Distance, 39.47 % for DisCoRoute, and 30 % for DijkstraHop. This escalation in variance is attributable to the absence of cross-links between the first and last orbital planes in the Walker-Star topology, which limits routing flexibility and can lead to longer detour paths under certain conditions.

Based on these findings, it can be concluded that the integration of the DisCoRoute heuristic with the WS-MHC model in a Walker-Star constellation setting is not only feasible, but also highly effective. The heuristic continues to approximate the optimal performance of Dijkstra-Distance while consistently outperforming DijkstraHop, even under the more constrained routing environment imposed by the Walker-Star architecture.

Nonetheless, it is important to acknowledge the trade-off introduced by this architectural choice: while mean latency remains well-controlled, the increased variance may negatively affect worst-case performance. Consequently, these implications should be carefully considered during the constellation design phase, particularly in mission scenarios that impose stringent bounds on end-to-end delay reliability.

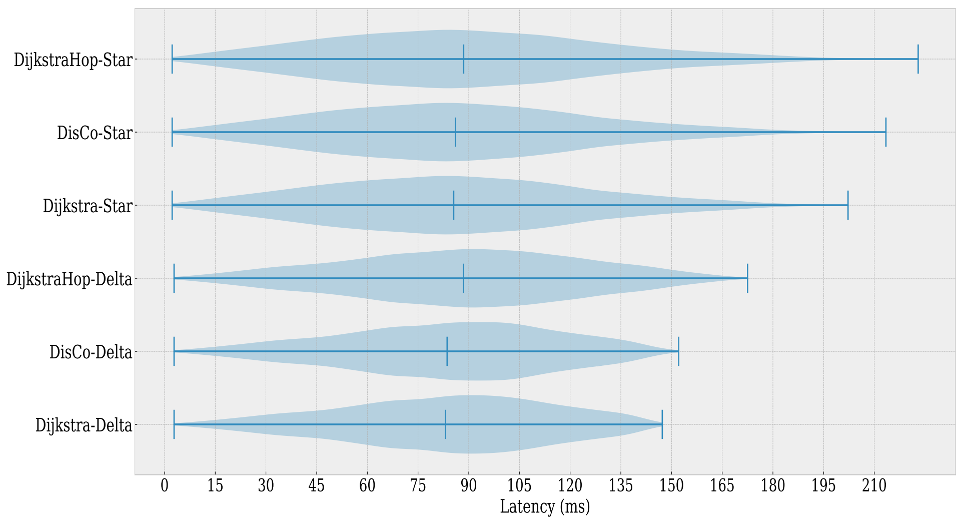


Figure 4.14: Latency distributions comparison between the tested algorithms in Walker-Delta and Walker-Star constellations

4.5.3 Hops Count

As shown in Figure 4.15, the transition to the Walker-Star constellation architecture yields a convergence of hops count distributions across all evaluated routing algorithms, a behavior analogous to that observed in the 3T configuration, proving the correctness of the WS-MHC model. This convergence can be primarily attributed to the elimination of inter-satellite links between the first and last orbital planes, which significantly constrains the number of routing directions available to the algorithms during path discovery. More specifically, the removal of cross-links between the first and the last planes results in only two feasible directions of horizontal propagation. Consider a scenario in which the longitude of the ascending node of the source satellite is lower than that of the destination. In this case, only the Eastward direction remains viable, as attempting to route Westward would necessitate crossing from the first to the last plane, which is no longer permitted under the Walker-Star configuration. Consequently, the set of valid propagation directions is reduced to North-East and South-East, effectively halving the directional search space. The same principle applies when the longitude of the source exceeds that of the destination, only with the opposite directions. Furthermore, in such constrained conditions, the two remaining propagation directions typically exhibit significant disparities in terms of hop efficiency. As a result, even Dijkstra-based algorithms, despite their exhaustive search capabilities, tend to select the same routing direction as the DisCoRoute heuristic, thereby leading to identical hop count distributions across all algorithms.

When comparing the performance under the Walker-Star configuration with that of the original Walker-Delta constellation, two key differences emerge. First, the average hop count experiences a noticeable increase of approximately 17.39 % across all evaluated algorithms. Second, the variance of the hop count distribution also increases substantially, with the worst-case scenarios under the Walker-Star configuration exceeding those of the Walker-Delta counterpart by at least 59.10 %.

These findings underscore the structural implications of the Walker-Star architecture on routing dynamics. The reduced connectivity not only limits path diversity but also imposes greater uniformity on routing behavior.

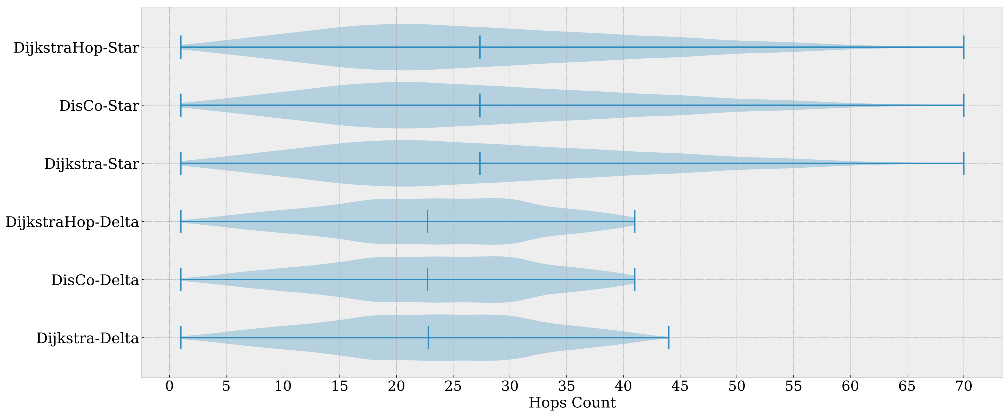


Figure 4.15: Hops count distributions of the tested algorithms in Walker-Delta and Walker-Star constellations

4.5.4 Computational Time

In Figure 4.16, we present a comparative analysis of the execution time percentages associated with the two versions of the DisCoRoute heuristic, namely, the original implementation designed for Walker-Delta constellations and the newly proposed variant adapted for Walker-Star configurations, both evaluated against the corresponding Dijkstra algorithms.

The results clearly illustrate the substantial computational efficiency of the proposed DisCoRoute implementation for Walker-Star constellations. Specifically, the new version requires, on average, only 0.75 % of the execution time consumed by the Dijkstra algorithm. This represents a remarkable improvement not only over its Dijkstra counterpart but also over its own Walker-Delta-based predecessor. Indeed, the Walker-Star version of DisCoRoute achieves a 46.81 % reduction in computational time relative to the original Walker-Delta variant.

These findings offer compelling evidence in support of both the effectiveness of the newly introduced WS-MHC model and the continued competitiveness of the DisCoRoute heuristic. Despite the structural constraints imposed by the Walker-Star topology, the new version maintains its hallmark of extremely low computational complexity, outperforming even the most established optimal-path algorithms such as those based on Dijkstra's method.

This result validates the viability of employing DisCoRoute in more constrained constellation architectures without sacrificing computational efficiency, thereby reinforcing its practicality for real-time or resource-limited onboard routing scenarios.

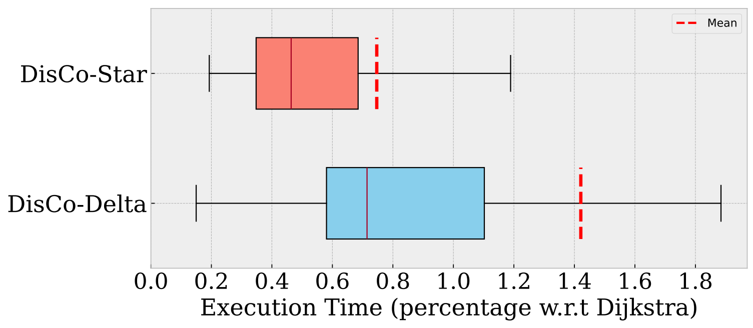


Figure 4.16: Execution time percentage of DisCoRoute with respect to Dijkstra in Walker-Delta and Walker-Star constellations

Chapter 5

Conclusions and Future Works

This master's thesis has conducted an in-depth investigation into the adaptability of the DisCoRoute algorithm and the MinHopCount model when applied to alternative inter-satellite link topologies for VLEO satellite constellations, with particular attention to the 3T topology and the Walker-Star configuration. These explorations introduced a variety of new challenges, primarily due to the additional structural constraints imposed by these configurations, constraints for which the original implementations were not initially designed.

Despite these challenges, the study has demonstrated that such adaptations are not only feasible but also highly effective. When benchmarked against state-of-the-art algorithms, particularly those based on Dijkstra's approach, the proposed methods achieved near-optimal performance in terms of E2E latency, an essential criterion for ensuring quality of service in satellite communication networks. Moreover, DisCoRoute consistently outperformed Dijkstra-based methods in terms of computational efficiency, achieving several orders of magnitude of improvement, the competitiveness and practical applicability of DisCoRoute in the context of VLEO/LEO satellite constellations.

The analysis began by extending the DisCoRoute algorithm to operate within a time-varying scenario. This represents a novel contribution, as previous works such as [3] and [4] evaluated the algorithm exclusively under static conditions, limited to a single time instant. In this study, a dynamic simulation environment was developed in `MATLAB`, allowing the assessment of DisCoRoute over a 12-hour window. The simulation considered 10,000 randomly selected source-destination pairs from a VLEO Walker-Delta constellation (65°:2080/52/10) [11], with satellite positions updated every 15 minutes.

The results demonstrate that DisCoRoute successfully adapts to the dynamic changes in satellite positions, recalculating paths at each time step to minimize cross-plane distances. This adaptation yielded average latency variations on the order of 1.20 %, indicating stable performance over time. However, a more significant impact was observed in the variance of latency, which increased by up to 14.57 %, particularly for routes involving a high number of horizontal hops. These findings highlight the limitations of a minimum-hop-based algorithm where both the number of hops and the direction of search are statically defined, regardless of the current network state.

Subsequently, the adaptability of DisCoRoute to the 3T topology was investigated. This began with the development of a variation of the original DisCoRoute, still grounded in the original MinHopCount model, but adapted to operate within the 3T scenario. The evaluation of this version revealed notable performance degradations, primarily due to the lack of a model capable of accurately predicting both the required number of hops and the appropriate direction of search under the 3T constraints. Specifically, when compared to Dijkstra-based algorithms, the DisCo3T implementation exhibited a performance gap on average of 24.55 % in terms of E2E delay and 21.43 % in hop count.

To address this, a modified hop-count model, IS-MHC, was introduced to ensure reachability in topologies with limited cross-plane connectivity. This enhancement was integrated into a revised version of the routing algorithm, denoted as DisCo3T, which retained the core logic of the first version of DisCoRoute for 3T.

Experimental results confirmed that the increased latency and hop count observed in 3T configurations are primarily due to structural constraints, rather than only the intrinsic limitations of the initial adaptation of DisCoRoute. Nevertheless, DisCo3T achieved routing performance comparable to that of the original 4T implementation, while maintaining the low computational complexity characteristic of the DisCoRoute family. Specifically, it demonstrated near-optimal results in terms of latency, exhibiting only a 2.50 % performance gap relative to Dijkstra3T, while requiring merely 3.56 % of its computational time.

Unlike state-dependent schemes, DisCo3T relies solely on static orbital parameters, enabling fully decentralized routing with minimal control-plane overhead.

Finally, a modified hop-count model tailored for Walker-Star constellation configurations was introduced. Once integrated into the DisCoRoute algorithm, simulations again demonstrated near-optimal performance when compared to the Dijkstra benchmark. With an average latency gap of only 1.20 %, the algorithm not only maintained its computational efficiency, requiring just 0.75 % of the computational time needed by Dijkstra-based methods, but also outperformed the original DisCoRoute implementation.

5.1 Conclusion

In conclusion, this thesis has contributed novel routing heuristics and minimum hop-count models adapted to various ISL topologies and satellite configurations. In particular, new models capable of accurately estimating the minimum number of hops in the 3T topology and in Walker-Star constellations were proposed. Their integration into DisCo-based algorithms yielded highly encouraging results, consistently achieving near-optimal performance in all simulated scenarios. The analytical structure of these algorithms renders them lightweight, scalable, and highly suitable for both onboard deployment and large-scale simulations.

Some of the results obtained during this thesis have been collected in [11].

5.2 Future Works

Building upon the promising results obtained in this thesis, several research directions can be pursued to advance the proposed methods further:

- As previously discussed, DisCo3T still presents opportunities for optimization. A focused study on improving its path selection strategy could further enhance both latency and stability.
- The time-variant analysis could be extended to both the 3T and Walker-Star versions of DisCoRoute, which were only evaluated under static conditions. Incorporating temporal dynamics would provide a more realistic assessment of routing behavior in operational satellite networks. A comparative analysis including Dijkstra-based algorithms under the same dynamic scenarios would further strengthen the evaluation.
- To improve resilience and practical applicability, future work should also explore the extension of DisCo3T to handle real-world conditions such as dynamic congestion, link failures, and satellite node outages. These enhancements would enable the algorithm to adapt more robustly in non-ideal environments.
- Finally, a combined analysis of DisCo3T applied to a Walker-Star constellation, thus merging the IS-MHC and WS-MHC models, remains an open and promising research direction. Evaluating the algorithm's performance under this hybrid condition could further validate its generality and effectiveness.

Bibliography

[1] N. H. Crisp, P. C. E. Roberts, S. Livadiotti, et al. «The Benefits of Very Low Earth Orbit for Earth Observation Missions». In: *Progress in Aerospace Sciences* 117 (2020), pp. 100–619. DOI: `10.1016/j.paerosci.2020.100619` (cit. on p. 2).

[2] V. Ray, T. E. Berger, Z. C. Waldron, et al. «The Impact of Space Weather on Very Low Earth Orbit (VLEO) Satellites». In: *AMOS* (2022) (cit. on p. 2).

[3] G. Stock, J. A. Fraire, and H. Hermanns. «Distributed on-demand routing for LEO mega-constellations: A starlink case study». In: *ASMS/SPSC 2022*. 2022, pp. 1–8. DOI: `10.1109//ASMS/SPSC55670.2022.9914716` (cit. on pp. 2, 4, 6, 7, 11, 15, 26, 36, 41, 70).

[4] Camilla Ottaviani, Alessandro Compagnoni, Juan A. Fraire, Giacomo Verardo, Gabriel Maiolini Capez, Daniel Gaetano Riviello, Gregory Stock, Carla Fabiana Chiasserini, and Roberto Garello. «Advanced Routing Strategies for LEO and VLEO Constellations: Ensuring Polar Coverage». In: *2025 12th ASMS/SPSC*. 2025, pp. 1–8. DOI: `10.1109/ASMS/SPSC64465.2025.10946062` (cit. on pp. 2, 70).

[5] Wei Wang, Yongli Zhao, Yuanjian Zhang, Xinyi He, Yue Liu, and Jie Zhang. «Intersatellite Laser Link Planning for Reliable Topology Design in Optical Satellite Networks: A Networking Perspective». In: *IEEE Transactions on Network and Service Management* 19.3 (2022), pp. 2612–2624. DOI: `10.1109/TNSM.2022.3168148` (cit. on p. 2).

[6] *Starlink: Satellite Technology*. 2025. URL: `https://www.starlink.com/technology` (visited on 06/12/2025) (cit. on p. 2).

[7] Yueyi Li, Junfeng Wu, Guohua Kang, Luyu Chen, Yuhuan Qiu, and Wenwen Zhou. «A Flexible Topology Control Strategy for Mega-Constellations via Inter-Satellite Links Based on Dynamic Link Optimization». In: *Aerospace* 11.7 (2024). ISSN: 2226-4310. DOI: `10.3390/aerospace11070510` (cit. on p. 2).

[8] Quan Chen, Lei Yang, Yong Zhao, Yi Wang, Haibo Zhou, and Xiaoqian Chen. «3-ISL Topology: Routing Properties and Performance in LEO Megaconstellation Networks». In: *IEEE Transactions on Aerospace and Electronic Systems* 61.2 (2025), pp. 4961–4972. DOI: `10.1109/TAES.2024.3512535` (cit. on pp. 3, 54).

[9] European Space Agency. *Handover, Data Routing and Radio Resource Management for Very Low Earth Orbit (VLEO) Broadband Constellations (HANDING-OVER)*. 2025. URL: `https://connectivity.esa.int/projects/handingover` (visited on 07/11/2025) (cit. on p. 3).

[10] Q. Chen, G. Giambene, L. Yang, and C. Fan. «Analysis of inter-satellite link paths for LEO megaconstellation networks». In: *IEEE Transactions on Vehicular Technology* 70.3 (2021), pp. 2743–2755. DOI: `10.1109/TVT.2021.3058126` (cit. on p. 11).

[11] Nicolò Benso, Alessandro Compagnoni, Gregory Stock, Juan A. Fraire, et al. «Distributed On-Demand Routing for VLEO Constellations with 3-Terminal Inter-Satellite Links». In: *SoftCOM2025* (2025), pp. 1–7 (cit. on pp. 15, 45, 70, 72).

[12] Inc The MathWorks. *SatelliteScenario.WalkerDelta*. 2022. URL: `https://it.mathworks.com/help/aerotbx/ug/satellitescenario.walkerdelta.html` (visited on 07/11/2025) (cit. on pp. 16, 65).

[13] Inc The MathWorks. *SatelliteScenario*. 2021. URL: `https://it.mathworks.com/help/aerotbx/ug/satellitescenario.html` (visited on 07/11/2025) (cit. on p. 16).

[14] Defense Mapping Agency. *Department of Defense World Geodetic System 1984*. DMA, TR 8350.2. 1991. URL: `https://apps.dtic.mil/sti/pdfs/ADA280358.pdf` (visited on 07/11/2025) (cit. on p. 16).

[15] Inc The MathWorks. *SatelliteScenario.WalkerStar*. 2023. URL: `https://it.mathworks.com/help/aerotbx/ug/satellitescenario.walkerstar.html` (visited on 07/11/2025) (cit. on p. 65).