FOUNDATIONS OF LEO SATELLITE EDGE COMPUTING: AN EMPIRICAL STUDY BASED ON THE HYPATIA SIMULATOR

by

TIEN DINH

A THESIS

Presented to the Department of Computer and Information Science
and the Division of Graduate Studies of the University of Oregon
in partial fulfillment of the requirements
for the degree of
Master of Science

June 2023

THESIS APPROVAL PAGE

Student: Tien Dinh

Title: Foundations of LEO Satellite Edge Computing: An Empirical Study Based on the Hypatia Simulator

This thesis has been accepted and approved in partial fulfillment of the requirements for the Master of Science degree in the Department of Computer and Information Science by:

Lei Jiao	Chair
Jun Li	Core Member
and	
Krista Chronister	Vice Provost for Graduate Studies

Original approval signatures are on file with the University of Oregon Division of Graduate Studies

Degree awarded June 2023

THESIS ABSTRACT

Low Earth Orbit (LEO) satellite networks have gained significant attention in recent years due to their potential to revolutionize global connectivity. SpaceX, Amazon, and Telesat are the most prominent players aiming to provide global internet coverage and high-speed broadband connectivity. Despite significant advancements and ongoing research in the field, there have only been a few studies that managed to construct a comprehensive, scaled simulator for LEO satellite networks. Project Hypatia developed by Amazon stands out as one of the few publications that has effectively accomplished such a task. This thesis builds upon the foundation of Hypatia to explore and provide an insight into the key aspects of LEO satellite networks. Our research consists of two main components: firstly, addressing fundamental issues such as network topology, routing delay, handover, and service provisioning in LEO satellite networks, and secondly, deploying code on top of Hypatia to analyze and explore these issues in depth. All in all, this thesis aims to deepen the understanding of LEO satellite networks through the examination of the Hypatia simulator and its implications.

ACKNOWLEDGEMENTS

I’d like to express my deepest gratitude to my advisor, Professor Lei Jiao, for his guidance, motivation, and insightful discussions throughout the project. This project would not have been possible without Professor Lei Jiao's assistance. I extend my sincere appreciation to the members of the committee, Professor Jun Li, and Professor Joe (Yingjiu) Li, and the students of CCSP group for creating an excellent research and study environment. Special thanks to my dear friend Alex Xiao Cai for answering numerous Python-related questions and for always teaming up with me on the projects. I’d also like to thank my uncle Duc Luong for encouraging and pushing me to achieve greater things in life. Lastly, I’d like to thank my parents and my sister for their unwavering support throughout the years when I was studying far from home. They have made my dream a reality.

TABLE OF CONTENTS

Chapter Page

LIST OF FIGURES

LIST OF TABLES

CHAPTER I

INTRODUCTION

Low Earth Orbit (LEO) Satellite Networks have recently emerged as an alternative solution to address the growing demand for seamless connectivity and global coverage. LEO satellite networks orbit the Earth at typically low altitudes, ranging from 200 to 2,000 km height above sea level. Being much closer to the Earth offers several advantages, including low latency, minimal path loss, and true global coverage. While it may be premature to assume that LEO satellite networks will completely overtake terrestrial networks, they have the potential to challenge the current terrestrial infrastructure in some specific markets like global service coverage, navigation, or location-based services. However, LEO satellite networks are complex systems containing thousands of satellites orbiting at a fast pace. This raises many questions that this paper aims to address.

Under the context of a LEO satellite network, several concepts play an important role in understanding its functioning. First is the network topology referring to the arrangement of the satellites and the interconnection with ground stations in the network. The topology determines how the network is structured and the pathways through which data communication flows. Other aspects to be considered are delay and distance. Routing in the satellite network means the most efficient path for data transmission from a satellite source to the satellite destination, considering the link availability between the two. Satellite distance can be divided into two main categories: satellite-satellite and satellite-ground station. It directly affects the propagation delay and overall latency in the network, influencing the quality of communication.

The topic of connection handover deserves some attention due to its dynamic movement of satellite within the LEO constellation. It means continuing to transfer ongoing communication sessions from multiple angles, either satellite spotbeam coverage, satellite to satellite or satellite network layer adaptation. The successful management of connection handover processes are crucial for maintaining uninterrupted connectivity and ensuring efficient communication within the LEO satellite network.

Exploring the previous topics allows us to address one of the challenges in LEO satellite Edge Computing: Service Provisioning. It is making the decision in which satellite to host a particular service and defining the time slots for service delivery. Additionally, service provisioning considers the dynamic nature of LEO satellites' movement, as service migration or service replica may be necessary when the service on satellite is moving away from the ground station or user. To arrive at a potential solution, we implement several methods to optimize the placement and analyze the outcomes.

Our work employes Hypatia satellite network simulation framework [2] to explore and address these fundamental issues. Developed by Amazon, Hypatia offers a realistic representation of the network constellations, allowing for evaluating the performance of LEO satellite networks, studying constellation behavior, and developing optimization strategies. The primary objective of the paper is to utilize Hypatia as a tool for experimentation and investigation of the aforementioned issues.

In this thesis, we provide the following contributions:

- We conduct a comprehensive study of LEO satellite fundamentals topic of topology, routing, handover, and service provisioning (Chapter 3).
- We use the Hypatia simulator to investigate the distance and relationship between LEO satellites in the network (Section 4.2.1)
- We follow the routing strategy in Hypatia and discuss the available routing protocol options for LEO satellite networks (Section 4.2.2)
- To evaluate service placement, we implement the two baseline algorithms: randomize and greedy service placement. We analyze the results of these algorithms and discuss alternatives to enhance service placement (Section 4.2.3).

Additionally, we review relevant existing work in the field in Chapter 2 and draw conclusions based on our findings and consider future directions in Chapter 5.

CHAPTER II

RELATED WORK

The study of satellite networks has been around since 1960s when the geostationary satellites (GEO) was introduced. However, Low Earth Orbit (LEO) satellites gained momentum in the 2000s and witnessed an increasing popularity after 2020. In this chapter, we present a comprehensive review of the existing research on fundamental studies of LEO satellite networks along with the capabilities of a few open-source LEO satellite network simulators. Our work is heavily inspired by "Exploring the 'internet from space' with Hypatia" [1] and several other publications that have employed LEO satellite simulations.

In 2020, the paper "Network Simulator for Large Low Earth Orbit Satellite Networks" [3], provides an open-source network simulation framework [4] for LEO satellites. The simulation is implemented in Python3 and contains three simple components: Constellation, Simulation, and Graphic User Interface (GUI) utilizing OpenGL for 3D visualization. The Constellation class represented nodes and links using Python network classes, while the movement of satellites in the simulation is calculated using the PyAstronomy library [11]. At the time of publication, this paper served as a valuable resource for researchers to explore LEO satellite network design and routing strategies. However, this work has faced challenges over time due to the limited availability of public data on satellite nodes and ground stations, which impacts the accuracy and realism of the simulation. The constellation defined in the paper is based on the authors' own specifications. In more recent developments, project Hypatia [2] has emerged, offering a simulator built on the Telesat, Kuiper, and Starlink constellations, which are the three major competitors in the LEO satellite industry.

The popularity of the aforementioned LEO satellite simulator [4] has served as a base for the research in the paper "Optimizing Content Delivery in Large LEO satellite Communication Networks" [5] by T. Pfandzelter and D. Bermbach. The authors developed their own simulation [6] based on the framework provided in [4] and further expanded from it. Although they faced the same challenges in lacking realistic data, they imported their own data of US and Switzerland cities to support the research. The focus of their paper [5] revolves around adapting content delivery network (CDN) concepts from traditional terrestrial networks to LEO satellite networks, utilizing the points-of-presence technique to serve groups of clients in close vicinity. This related work was mentioned to highlight the significance of a reliable LEO satellite network simulator.

During that period, the same group of researchers with the inclusion of J. Hasenburg, published a paper titled "Towards a Computing Platform for the LEO Edge" [7]. The paper explains the important features of LEO Edge computing, such as mobile server infrastructure which refers to the dynamic movement of servers deployed on LEO satellite. The authors emphasize the importance of distributed servers to enable efficient processing and data storage within the limit computation of a satellite. Their findings lead to the conclusion that serverless functions is the optimal choice for the LEO satellite environment due to its high availability, utilization of open-source technology, and flexible deployment options. This paper lays the groundwork for LEO Edge computing platform.

Lastly, we must highlight the paper “Exploring the 'internet from space' with Hypatia” [1], which presents a missing key component in the form of a large-scale LEO satellite network simulator, Hypatia [2]. Hypatia represents a significant improvement over the previous simulator discussed in [4]. The project includes a more popular constellation framework and enhances 3D visualization. The Hypatia simulator accurately model after three constellations: Telesat [15], Kuiper [14], and Starlink [13]. It effectively captures the characteristics of satellite networks, including routing, latency, and link connections. Overall, project Hypatia serves as a crucial tool for addressing the challenges encountered in LEO satellite networking, enabling researchers to make progress in this field.

CHAPTER III

FUNDAMENTALS

In the domain of Low Earth Orbit (LEO) Satellites, there are several fundamental issues that have a significant impact on the network's performance and functionality. We introduce four key fundamental issues: network topology, distance and routing, connection handover, and service provisioning. Network topology refers to the arrangement and connections of LEO satellites within the network. As satellites are constantly moving, we examine the distance and neighboring relationships between satellites to gain insights into the network's stability and calculate future states. The link distance in the topology directly affects factors such as latency, routing, and delay, which in turn influence the communication efficiency and service delivery in the network. As satellites move away from specific locations and from each other, the issue of connection handover arises. Lastly, we establish a connection between the aforementioned fundamentals to address the challenge of service provisioning, which involves the allocation and optimization of resources to efficiently deliver services across the LEO satellite network.

In section 3.1, we explore three network topologies: Walker Star, Walker Delta, plus Grid constellation. Section 3.2 discusses different types of satellite distances and their impact on network routing. Section 3.3 addresses various types of LEO satellite handover and potential solutions. Section 3.4 discusses the concept of service provisioning and examines service migration.

3.1. Network Topology

The LEO satellite network can be organized into different topologies, with the Walker Star and Walker Delta constellations being popular in the past, and plus Grid constellation is a popular choice in modern times.

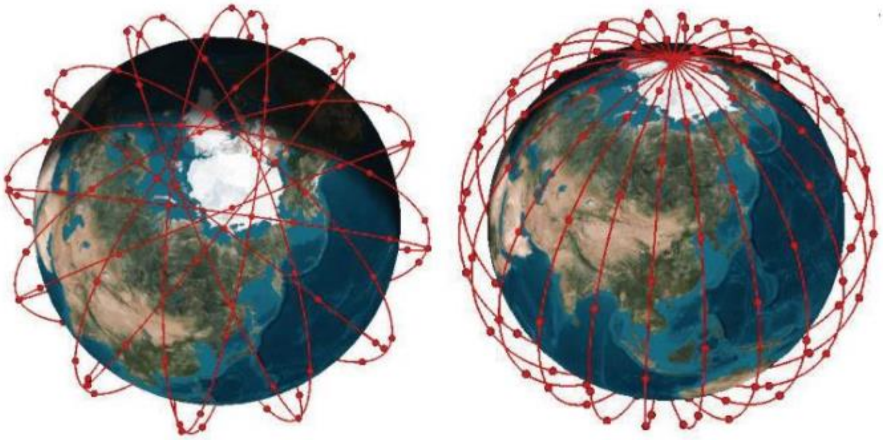

Figure 1. Walker Delta (left) and Walker Star (right). This figure was copied from the original paper [8]

In the early development year of LEO satellite, Walker Delta and Walker Star was the common constellation due to their simplicity but also provide good coverage area. In the Walker Star topology (Figure 1, right), satellites are evenly distributed in circular orbits at the same altitude from North Pole to South Pole, forming a star-like pattern. The Walker Delta topology (Figure 1, left) is a variation of the Walker Star configuration. It involves grouping three or more satellites together in a triangular or delta-shaped formation within the same circular orbit. Unlike the evenly spaced satellites in the Walker Star topology, the satellites in the Walker Delta topology are positioned closer to one another. This clustering allows for enhanced coverage and capacity, especially in densely populated regions.

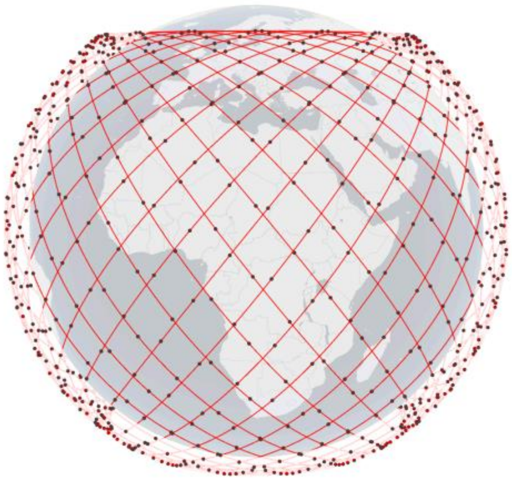

Figure 2. Amazon Kuiper + Grid Constellation. This figure was copied from the original paper [1]

In modern times, satellite providers are deploying an increasing number of satellites into orbit. The original Walker Star constellation failed to keep up with this growth, leading to the development of the plus grid or “+ Grid” constellation. In this design, satellites are strategically positioned in a grid-like formation, resembling a plus sign (+). In the + Grid constellation, each satellite is surrounded by four neighboring satellites. This includes two of the same neighbors, which are satellites positioned on the same orbit and forming a vertical or horizontal alignment. Additionally, there are two adjacent neighbors, which are satellites located in adjacent orbits and forming a diagonal alignment within the grid. The adjacent orbits in the + Grid constellation are characterized by having the same height but slightly different inclination angles and are in relatively close proximity. Importantly, the relationship among these four neighboring satellites remains constant over time. This allows two points of contact in the network to find each other hops to hops via the network link. This well-defined neighboring relationship facilitates efficient communication and data transfer across the network.

3.2. Distance and Routing

In the constellation, distance between satellites impacts the network connection. There are four types of distances to be considered:

1. Satellite-to-ground station is the straight-line physical distance.
2. Ground station-to-ground station considers the Earth's surface curvature, known as geodesic distance.
3. Satellite-to-satellite is the physical distance is the straight-line separation. However, the routing distance depends on established links between satellites.
4. In-range distance defines the maximum length that a satellite and a ground station establish effective communication.

Besides, satellite collision avoidance is important for orbit planning. It occurs when satellites come into close physical distance, leading to variations in their orbital heights or the implementation of collision detection and avoidance maneuverability on the satellite.

Routing in satellite networks involves establishing link connections between satellites and finding available hops to route from one satellite to another. The delay in routing is influenced by satellite distances. There are various types of routing delays that contribute to the latency and overall performance of the system. The transmission delay is usually insignificant if there is an unobstructed path for light transmission in the communication. The propagation delay, on the other hand, is determined by the number of hops multiplied with link distance divided by the speed of light. Queuing delay and processing delay in the LEO satellite network are comparable to those in terrestrial networks, referring to the time required for data waiting in the buffer and processing.

Hypatia [2] simulator as well as other simulator framework focuses primarily on the Propagation Delay when calculating the forwarding time. This is because important information about the computational power and specifications of LEO satellite is missing, which led to the exclusion of Queuing Delay and Processing Delay from the codebase. However, despite these limitations, Hypatia serves its purpose as an abstract software tool for studying the satellite network.

3.3. Connection Handover

In LEO satellite networks, handover is a process to ensure seamless communication and uninterrupted connectivity as satellites move across the sky. There are generally three types of handovers: spotbeam handover, inter-satellite link handover, and network layer handover. Each type of handover presents its own challenges and requires specific solutions to maintain reliable communication over the network.

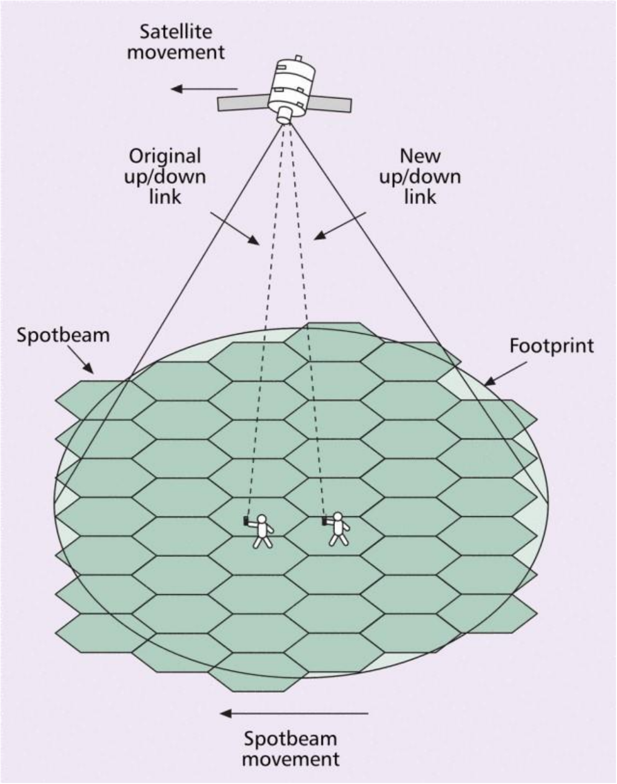

Figure 3. Spotbeam handover. This figure was copied from the original paper [9]

Spotbeam handover occurs when a user or ground station moves from the coverage area of one satellite's spotbeam to another. A spotbeam is a focused beam of radio frequency signals that provides coverage in specific areas on the Earth's surface. Spotbeam handover only concerns one satellite. As long as the user or ground station remains within the satellite's coverage area, the satellite will continue to establish an up/down link. This is the most frequent handover in a LEO satellite system, happening approximately every 1-2 minutes [9]. Maintaining connection during spotbeam handover relies on a single satellite mechanism. Pulak et al. [9] discussed two aspects: dynamically allocating radio channels and forming a policy to always guarantee successful handover.

Satellite providers could also explore the development of a beam tracking technique that allows satellites to continuously steer spotbeams toward the user or ground station, though it introduces more extensive computing.

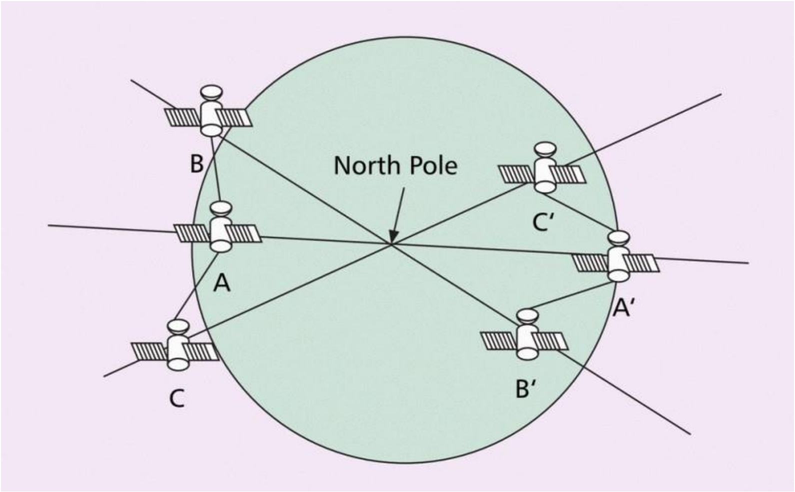

Figure 4. Inter-satellite link handover. This figure was copied from the original paper [9]

Inter-satellite link (ISL) handover takes place when the connection link is transferred from one satellite to another in the network. ISL handovers happen with a frequency that is the same as the visibility duration of a LEO satellite to a location on Earth, typically every 10-20 minutes. During an ISL handover, the connection is rerouted to the next satellite moving in the same location. Rerouting can happen as fast as a fraction of a millisecond for transferring between two satellites at the same orbit to a few hundred milliseconds if there are no direct link connection between the two satellites, which requires multiple hops routing. The key to solving this challenge lies in optimizing the routing algorithm, minimizing the rerouting frequency, and taking advantage of LEO satellites' predictable movement.

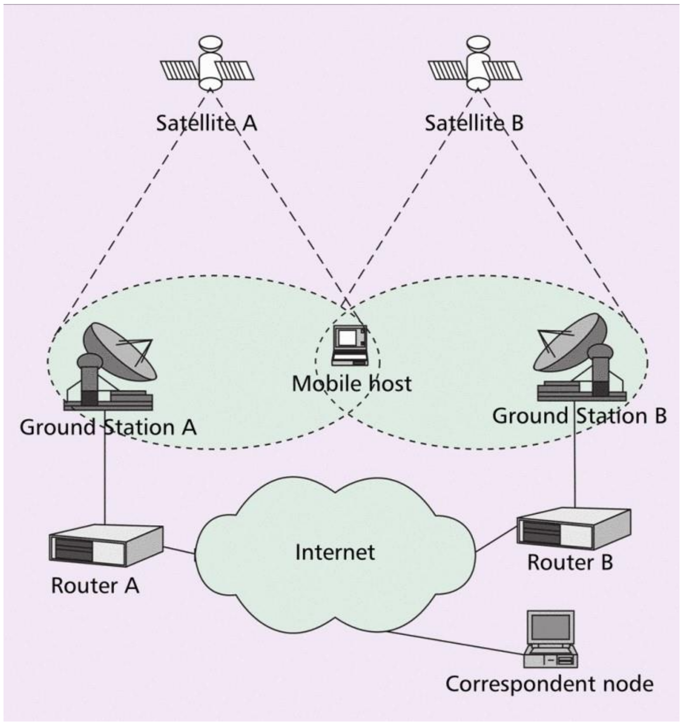

Figure 5. Network layer handover. This figure was copied from the original paper [9]

Network layer handover occurs when satellites need to transfer a communication session to a ground station or user while maintaining the network level connection. In their work, Pulak et al. [9] briefly mention two scenarios: satellite as a router and satellite as a host. Typically, LEO satellites are primarily designed for fast packet forwarding, thus often serving as routers. However, there are situations where a LEO satellite needs to provide services, becoming a mobile host while the ground stations assume the role of routers. In both scenarios, IP address changes will happen, either to the user/ground station or the satellite itself. Efficient network layer handover considers multiple factors such as signal strength, latency, and network load to select a suitable path. One potential approach is to implement a variant of Mobile IP in the satellite network.

A common factor among the three types of handovers is the utilization of the predictable pattern in satellite movements. Park et al. [10] reviewed several decision-making algorithms that incorporate reinforcement learning and game theory. Machine learning techniques promise to advance the progress of research on connection handover in LEO satellite networks.

3.4. Service Provisioning

Service provisioning is one of the key focuses in LEO Edge Computing, referring to the optimization of service access. The objective is to ensure seamless service delivery with minimal latency, thereby enhancing the overall user experience. In the network, a satellite possesses its own computing capability which allows them to host services in space. These services include various domains such as communication, observation, and navigation. This capability opens up opportunities for enhanced connectivity, real-time data analysis, remote sensing applications and many more.

Resources available on LEO satellites are, however, limited, raising several important questions. The first key question is to determine which set of satellites to host the service. There are many factors to be considered, such as distance to the user, optimizing rerouting mechanisms, or efficient resource allocation within a satellite. Another important aspect is defining the time slot for the service, involving decision-making at each time step, be it in seconds or minutes, given the fast and constant movement of satellites. Furthermore, the number of users demands or requests from the ground could be a factor for balancing resource utilization and meeting user expectations. The constant movement of LEO satellites introduces connection handover, commonly known as service migration. A typical service migration deals with ISL handover, where services need to be seamlessly transferred from one satellite to another as they move away from users or ground stations. This emphasizes the significance of rerouting mechanisms to maintain uninterrupted service.

In Section 4.2.3 of our paper, we investigate these service provisioning issues by utilizing the Hypatia simulator to conduct experiments and answer related questions, despite Hypatia's limitations in terms of information on satellite resource capabilities and lack of congestion control features mentioned by the authors [1]. We analyze two baseline strategies: randomized service placement and greedy service placement under various conditions.

CHAPTER IV

EXPERIMENTS WITH HYPATIA

4.1. Overview

This chapter utilizes the project Hypatia developed by Amazon to investigate a few issues in the LEO satellite networks. We explore the following topics: satellite neighboring distance, routing, and service provisioning. By conducting experiments using the Hypatia codebase, we gain insights into the behavior and performance of LEO satellite networks, enabling us to address one challenge after another.

4.1.1. Framework and Components

Project Hypatia [2] is a LEO satellite network simulator that is designed and built using Python's nx.Graph() library. The network is structured as a plus Grid constellation, where each node maintains four neighbors that remain unchanged over time. The simulator uses Floyd Warshall or Djisktra algorithm to calculate the shortest paths within the network. Additionally, ground stations are treated as user or terrestrial connection points entry to the network. Even though they are not part of the network, the simulator performs calculation from ground stations viewpoint.

Hypatia provides pre-defined configurations for the three well-known LEO constellations: Telesat, Kuiper, and Starlink. The constellation itself is built upon the Python PyEphem library [12], which enables precise astronomical computations. This library supports distance functions, satellite's locations, and pre-calculation of the constellation in future time step. Furthermore, the simulator offers post-processing features like graphs and 3D visualization, though the latter requires a third-party license. With these features, Project Hypatia provides a flexible platform for studying and analyzing LEO satellite networks.

4.1.2. Capability and Limitation

Hypatia shows their capability to simulate large-scale network topologies, with a specific focus on popular LEO satellite constellations. The simulator preprocesses data inputs from ground stations and satellites through multiple states to construct the network's behavior ahead in the future. It accurately represents the routing mechanisms employed in the networks, enabling the study of routing protocols and their impact on network performance. Hypatia can also provide a great environment for satellite constellation 3D visualization.

However, Hypatia does have certain limitations. Hardware-related issues, such as handover procedures and satellite specifications, are not fully captured in the simulator despite the complexity in the constellation. An example of this is the implementation of interfaces in satellites, where switching between interfaces occurs seamlessly and without any additional overhead. The authors of the simulation framework also admitted to the fact that the project could benefit from further improvement in multi-path routing and congestion control mechanisms [1] to improve realism in the satellite network. Overall, it is anticipated that researchers will use this project to some extent and try to incorporate their own implementation into their work.

4.2. Experimental Studies

In this section, we use Hypatia to experiment and analyze results from several aspects. In section 4.2.1, we examine potential distancing issues between satellites in the network and ground stations. Section 4.2.2 investigates the impact of link distance on routing delay. Lastly, section 4.2.3 addresses the service placement and migration challenges associated with the LEO satellite network.

4.2.1. Results on Neighborhood Distance

Our initial investigation determines the number of satellites in range of a ground station in different network constellations provided by various distributors. In Project Hypatia, we define the in-range distance of a satellite to a ground station as the straight-line distance being less than a pre-defined maximum length. We focus on three configurations: Telesat, Kuiper, and Starlink constellations, which are detailed in the table below.

Constellation	Satellites	Ground Stations	Max Length (km)
Telesat	700	100	5845
Kuiper	1156	100	1260
Starlink	1234	100	1090

Table 1. Satellite, ground station and max (in range) length setting for the three constellations: Telesat, Kuiper and Starlink

It is important to note that although these constellations have the different satellites setting, they utilize the same set of ground stations. Using this information, we calculate the number of in-range satellites at different time epochs with fixed ground stations.

	Number of satellites in range of the ground station			
Constellation	Chicago	Tokyo	Delhi	London
Telesat	70	62	53	80
Kuiper	13	10	7	14
Starlink	12	10	8	18

Table 2. Average number of satellites in range of 4 different ground stations in 1 hour

The results presented in Table 2 wide range distribution. Depending on the concentration of satellites across different regions, ground stations positioned in Chicago or London for example, will encounter a higher number of satellite contacts. On average, the number of satellites within range varies between 8 and 15 for Kuiper and Starlink, while Telesat exhibits a higher range of 50 to 80 due to its significantly longer maximum in-range length. These findings provide us with a general understanding of satellite – ground station relationship to aid further development in the following sections.

Next, we gather information on the distances between neighboring satellites in the three constellations using a plus Grid topology. The four connections or direct links from the same orbit and adjacent orbits remain unchanged over time. This raises the question of the distance relationship between satellites in the same orbit and those in adjacent orbits when they are linked. To address this question, we look into a specific scenario within the Kuiper constellation. We track satellite 1 (sat 1), which shares the same orbit with satellites 0 and 2 and is adjacent to satellites 35 and 1123 in a neighboring orbit. We measure the straight-line distances in kilometers between these satellites over a one-hour period. The result is displayed in Figure 6.

This figure shows that satellites in the same orbit maintain a consistent distance over time, while satellites in adjacent orbits experience changing in distance. The changes follow a predictable pattern, with the distance at time step t = 0 being the same as at time step t = 50 minutes. The distance curve resembles a sine or cosine arc. To better understand this behavior, let's consider two adjacent orbits within the same shell. For example, satellite 1 belongs to the first orbit rotating north-east, while satellites 35 and 1123 belong to the second orbit rotating north-west. In Kuiper setting, one satellite (35) in the adjacent orbit will be mostly closer to satellite 1 while the other (1135) is mostly further away. At a certain point after 45 minutes from the starting time, satellite 1135 comes closer to satellite 1 compared to satellite 35. This repeating pattern is also observed in the Telesat and Starlink constellations. The neighboring distance in the Starlink satellite constellation exhibits a more gradual curve, whereas the distance in the Telesat constellation shows greater fluctuations, as depicted in Figure 7 and Figure 8.

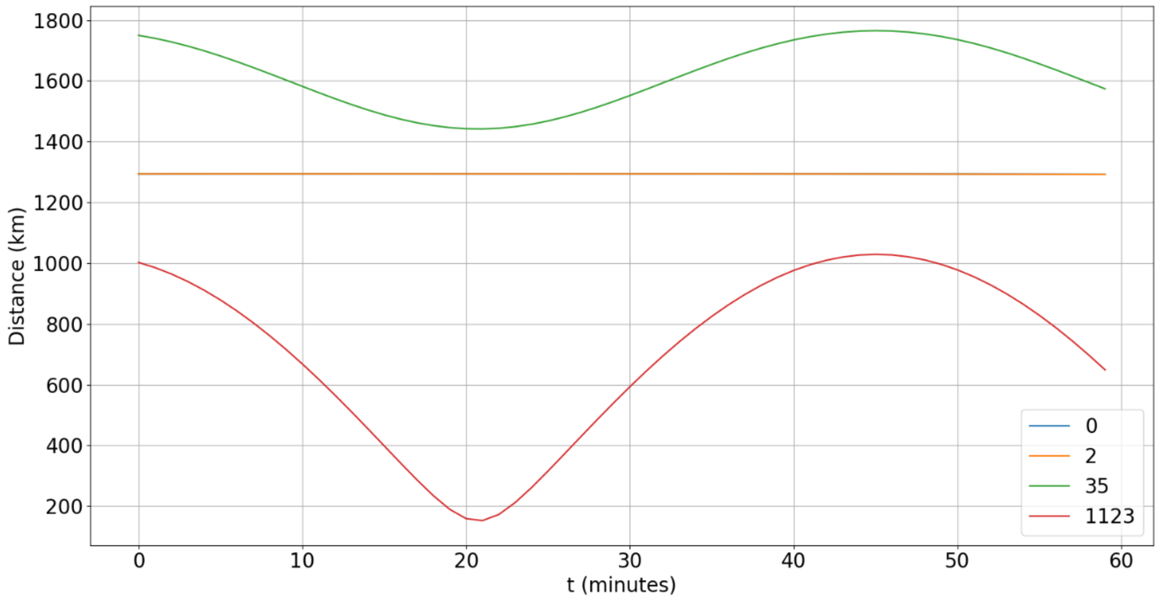

Figure 6. Kuiper constellation: Distance difference of Satellite id=1 with Same Orbit Satellite (0,2) and Adjacent Orbit Satellites (35, 1123) over 1-hour at 1-minute intervals

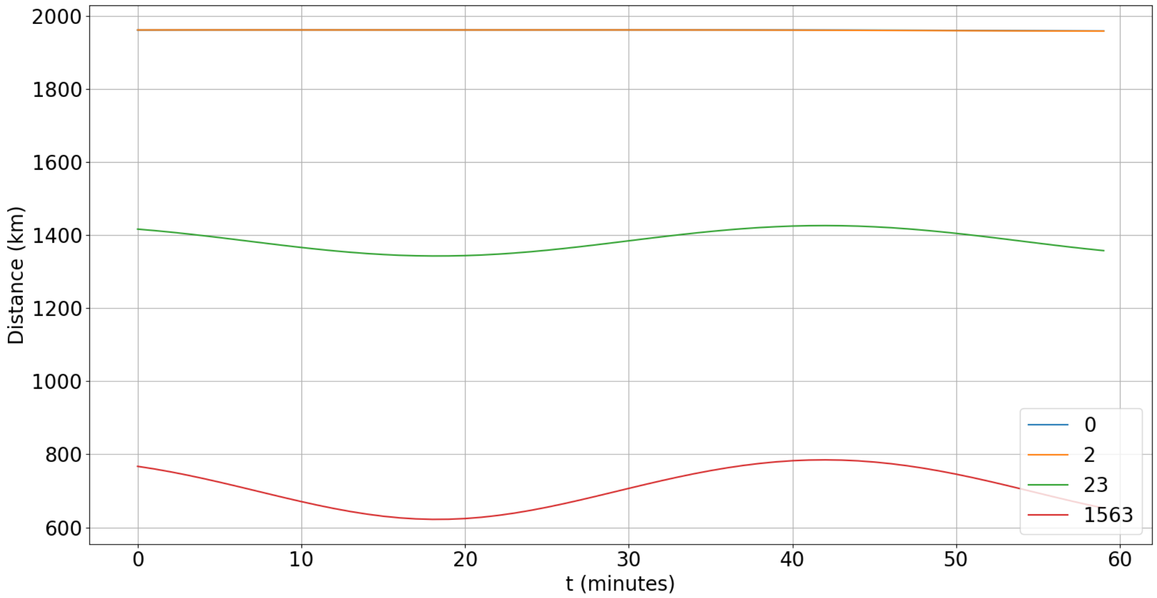

Figure 7. Starlink constellation: Distance difference of Satellite id=1 with Same Orbit Satellite (0,2) and Adjacent Orbit Satellites (23, 1563) over 1-hour at 1-minute intervals

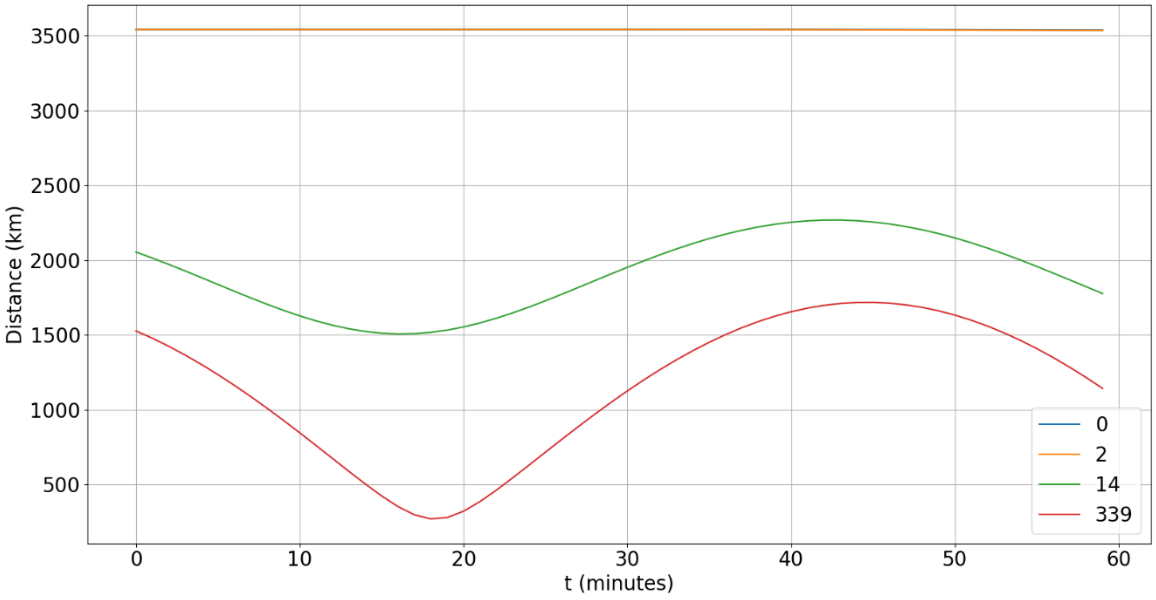

Figure 8. Telesat constellation: Distance difference of Satellite id=1 with Same Orbit Satellite (0,2) and Adjacent Orbit Satellites (14, 339) over 1-hour at 1-minute intervals

It is worth noting that both the Kuiper and Starlink figures have a lower range of distances between satellites compared to Telesat, which is the consequence of higher density of satellites in the constellations. Regardless, it can be concluded that the distances between satellites within the same orbit remain stable over time, while distances between satellites in different orbits fluctuate in a predictable manner.

The possibility of satellites flying near each other raises concerns about collision avoidance and risk reduction. We employ an algorithm that compares the distances between every satellite to each other within the network over a time period. The analysis focuses on the standard Kuiper constellation configuration, consists of 1156 satellites, but observes in a shorter duration. The results are presented in Figure 9.

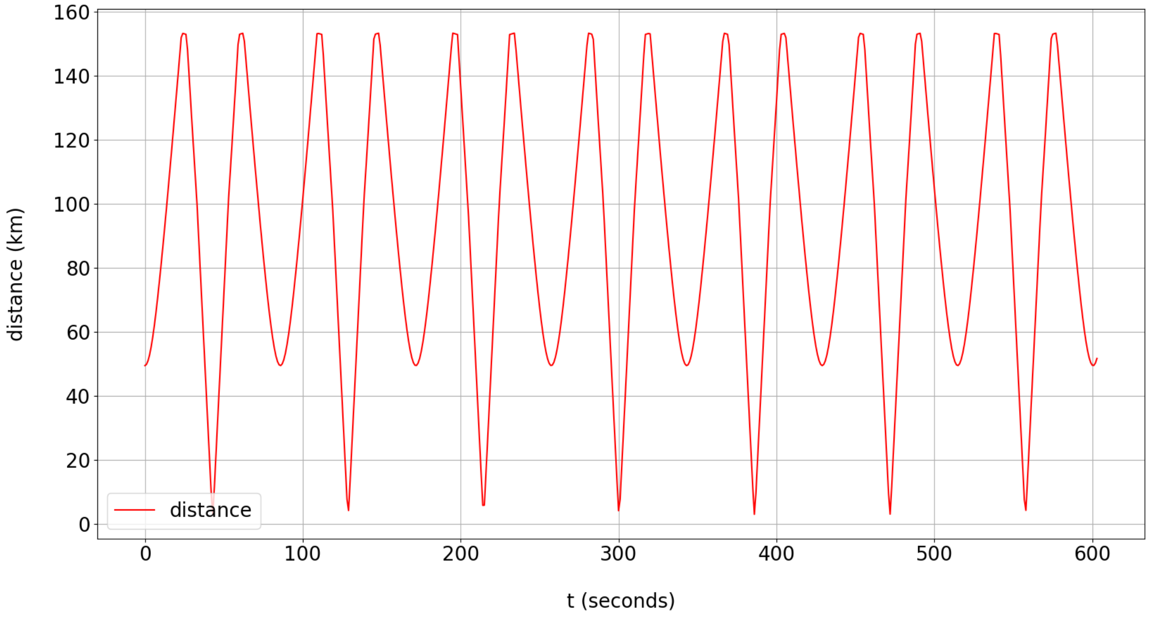

Figure 9. Minimum satellite pair-distance (km) over 10-minutes at 1-second interval

We see a repetitive change but in terms of the minimum distance among satellites. At the closest point, two satellites reach 2.9147 km, which is relatively far away. We discovered that while orbiting in the same shell, the satellite's height varies to minimize collision risks. In the experiment, Kuiper satellite orbits in a shell at height 639 ± 1 km. Although satellites within the same constellation are less likely to collide, outside factors such as debris or satellites from other manufacturers can introduce additional risks. Typically, satellites incorporate their own collision detection system and are designed to perform collision avoidance maneuvers, such as panel shrinking to reduce its own size. Additionally, in industry, satellite providers establish agreements to launch their constellations at different altitudes to reduce collision risks.

4.2.2. Results on Path and Routing

In satellite network routing, there are scenarios where two satellites in close distance exchange data to maintain continuous service at a specific location on Earth. To understand the routing distance, at time-step 386.0 seconds, we observe two satellites in figure 7 reaching their closest point. The gathered information for these satellites includes:

- Real distance: 2.9147 km
- Routing distance: 21,812.84 km
- Number of hops: 19
- Round-Trip Time (RTT): 145.475 ms

We use the Dijkstra algorithm to calculate the shortest path between the two satellites. Despite their close physical distance, the shortest available path between these satellites is extensive, resulting in a high RTT. This outcome highlights the possibility of routing taking a much larger distance path around the world, even when satellites are physically nearby.

Next, we experiment with the current state of Hypatia routing option. Hypatia by default always uses the available shortest path protocol. To test the routing capability, we implement algorithm to calculate shortest paths from every pair of ground station through the satellite network. The experiment runs in the Kuiper constellation, with 1256 nodes and 3190 edges including the end connection with the ground station.

Time step	t = 100s	t = 1000s	t = 2000s	t = 3000s	t = 3600s
Top frequency 1	466	470	435	478	519
Top frequency 2	377	449	411	472	519
Top frequency 3	363	433	402	422	519

Table 3. Highest frequency of satellite nodes in the network from multi way traffic across 100 ground stations over 3000 seconds duration in Kuiper constellation.

Table 3 measures the few highest satellite appearances in all paths at each time step when always utilizing the shortest path. This is under Hypatia simulator setting that assumes future knowledge of the network state. Path switching occurs every few seconds. In result, the appearances of a satellite averaging 500 times at peak indicates a heavy bottleneck issue, as also discussed in Kassing et al. [1]. This routing option results in significant queueing delay, rendering the RTT calculation in the Hypatia simulator inaccurate. Additionally, there will be notable hardware performance and energy consumption costs for every single satellite.

Assuming no traffic or congestion, we employ the default shortest-path routing protocol in Hypatia to estimate the round-trip time (RTT) from the Chicago ground station to the Tokyo ground station across three constellations. The data was obtained from the same ground station in Chicago at the initial time. Figure 10, 11, and 12 illustrate the RTT values from the Kuiper, Starlink, and Telesat constellations.

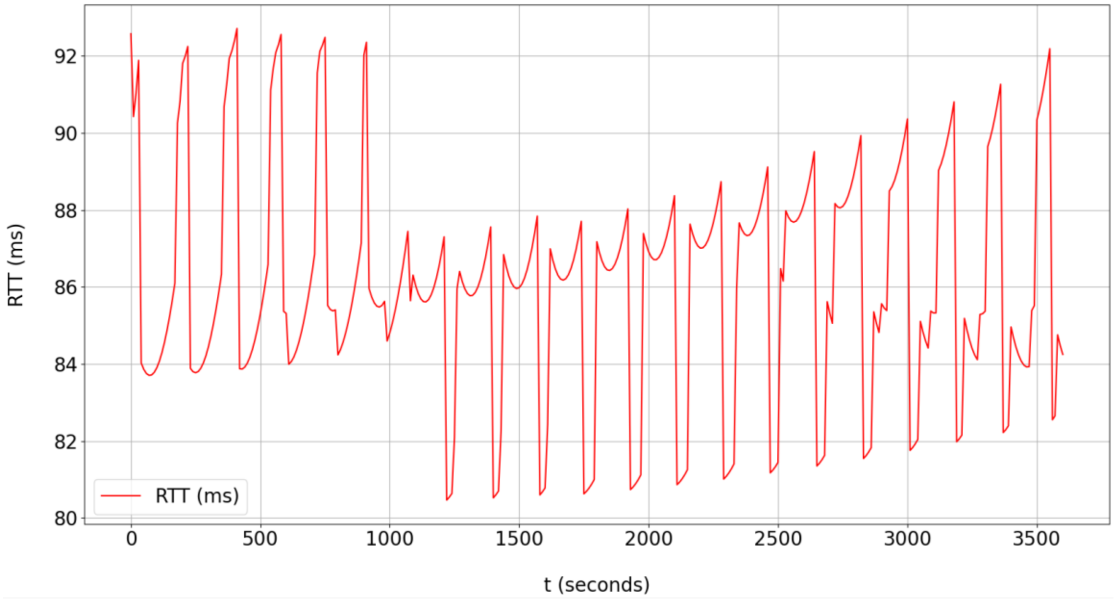

Figure 10. RTT (ms) between Chicago and Tokyo in the Kuiper Constellation: 1-hour duration with 10-seconds intervals

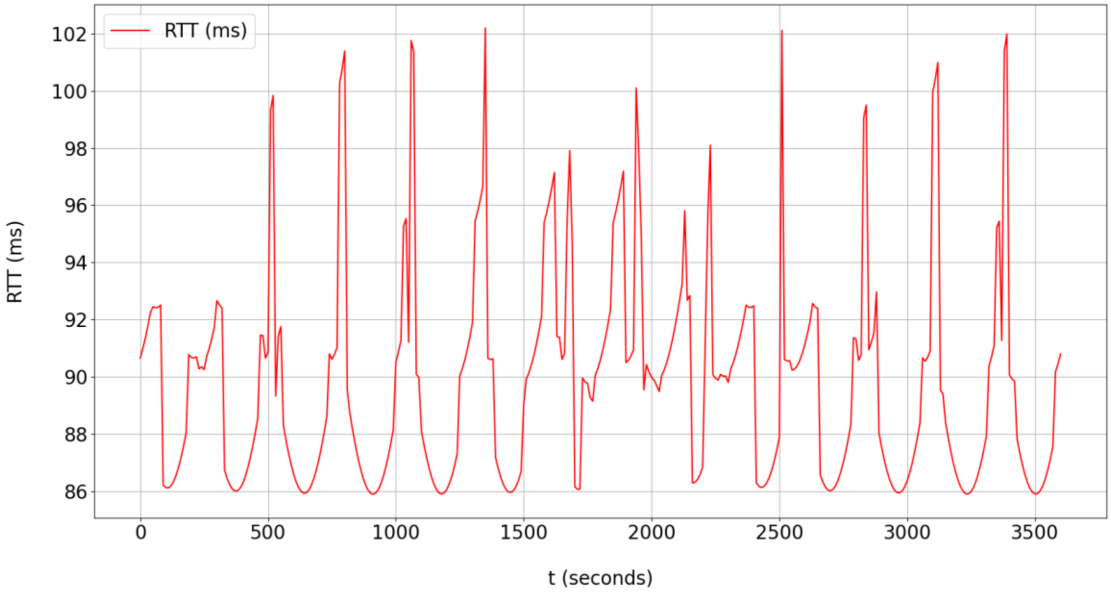

Figure 11. RTT (ms) between Chicago and Tokyo in the Starlink Constellation: 1-hour duration with 10-seconds intervals

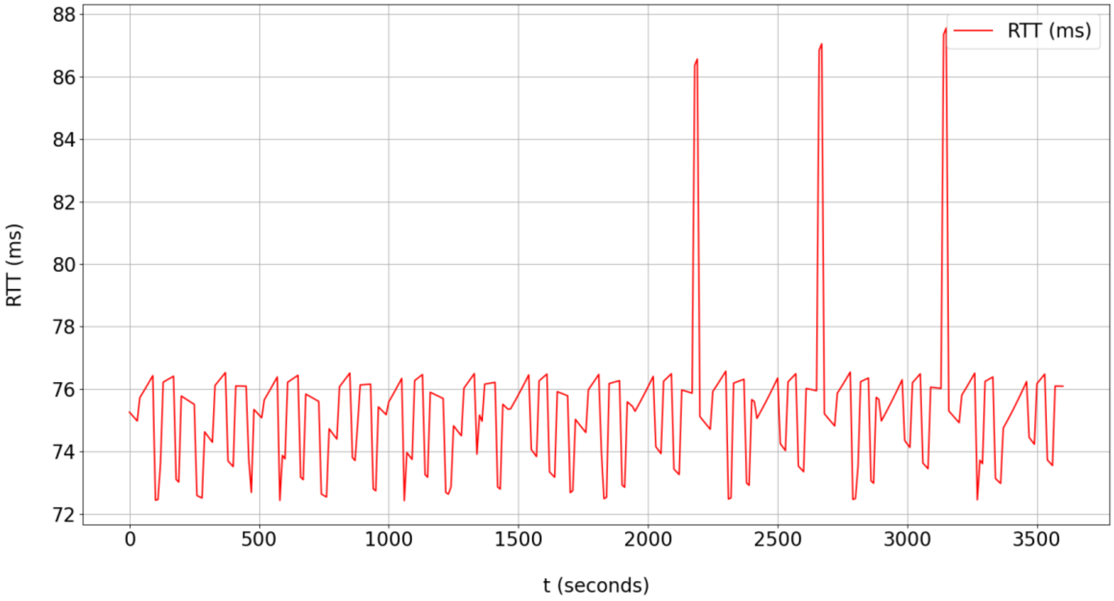

Figure 12. RTT (ms) between Chicago and Tokyo in the Telesat Constellation: 1-hour duration with 10-seconds intervals

The average RTT observed in Figure 10, 11, and 12 is 86.1ms, 89.8ms, and 75.4ms, respectively. The packet travel time from Chicago to Tokyo is roughly similar in both the Starlink and Kuiper constellations, while the Telesat constellation experiences a slightly faster RTT. There is considerable noise in the results due to the potential updates in the shortest path at each time interval. An interesting phenomenon is that Telesat, despite having a simpler constellation with fewer satellites, performs better path routing compared to the other two constellations. However, upon closer examination of Figure 8, it becomes noticeable that data transfer between two satellites can span a distance range of up to 3500km, which may not be realistic considering additional factors such as the quality of light beam transmission or the curvature of the constellation similar to the Earth's surface. Nevertheless, the consistently low RTT values across three constellations demonstrate an improvement compared to the RTT observed in the terrestrial network.

The dynamic nature of LEO satellites prevents the direct application of traditional terrestrial network routing protocols. A synchronized state of the network is short-lived, leading to a series of computationally expensive changes when the network changes. One potential solution is to make use of the knowledge of the entire constellation, as demonstrated in the simulator, to determine beforehand the complete system routing path at each time step in a distributed manner. Zhu et al. [16] also propose a method involving a few centralized satellites dedicated to handling rerouting computational tasks. These approaches aim to evenly distribute traffic flow within the network while minimizing the impact on communication efficiency.

4.2.3. Results on Service Placement and Migration

Service placement mainly concerns the accessibility from a ground station to the closest service, while service migration involves rerouting service from one satellite to another. We assume a satellite in the constellation is capable of hosting service and routing capability given or take extensive traffic. We focus on Kuiper constellation, consisting of 1156 satellites and 100 ground stations. Each node in the network contains 4 path connection to other nodes. At each time step, we add each ground station as an end node to the closest in-range satellite.

Our investigation has two baseline algorithms:

- Random approach: For every time step, a satellite is randomly chosen for service placement. Migration is not considered in this approach, as a ground station must always find the connection to the closest service among the randomized options.
- Greedy approach: First, initialize a number of ground-station groups in close proximity. For each time step, for every group, find a satellite closet to that group. If a satellite moves out of range of a group, we calculate the migration of services.

In this experiment, we deploy 20 services across satellites in the constellation. Our goal is to calculate the Round-Trip Time (RTT) from a ground station in Chicago to the nearest service and analyze the performance of the two algorithms.

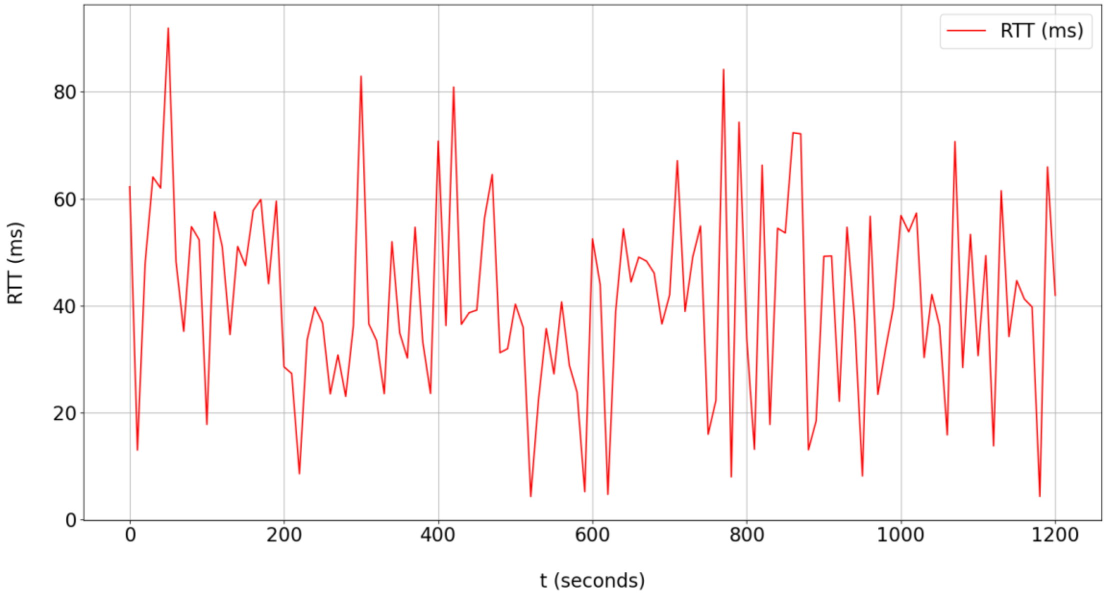

Figure 13. RTT from Chicago ground station to the closest of 20 randomized services over 20-minutes at 1-second interval

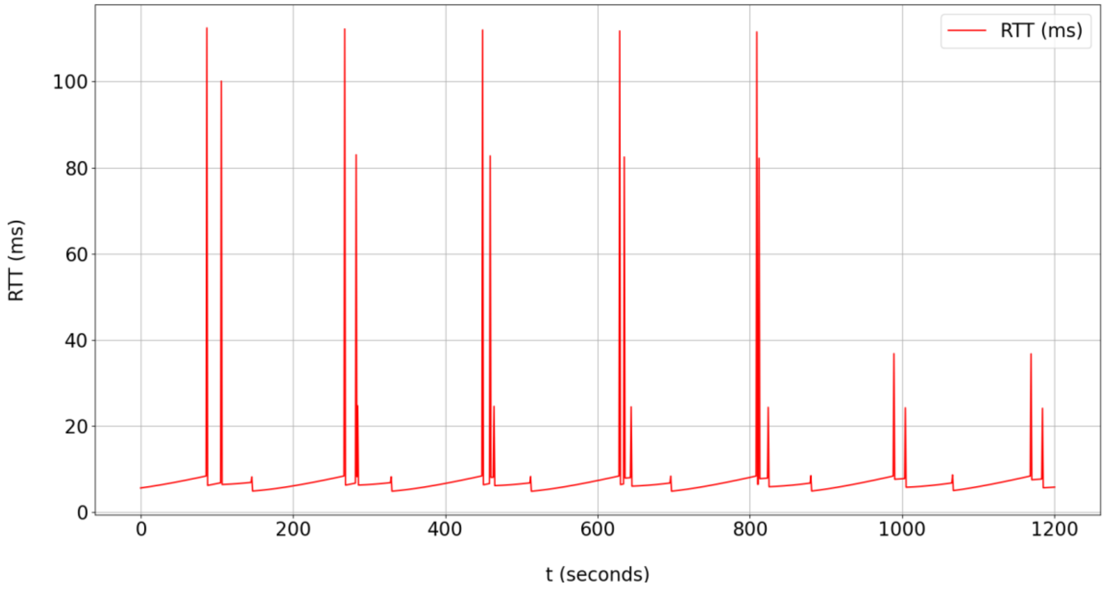

Figure 14. RTT from Chicago ground station to the closest of 20 greedy service placements (plus migration) over 20-minutes at 1-second interval

The average RTT to access the closest service from Chicago ground station is approximately 41.2ms (Figure 13) with the randomize algorithm, and 7.43ms (Figure 14) with the greedy algorithm. In the random algorithm, services are initially placed randomly, potentially resulting in services being further away from the Chicago ground station. This leads to more noise and fluctuation in routing speed. On the other hand, the greedy algorithm pre-calculates the closest service for each group, ensuring that each group of ground stations always has a nearby satellite for service access. Migration occurs every less than 200 seconds as shown in Figure 10. It occurs when a satellite with the service moves away from the group, but a closer satellite comes within range, triggering rerouting similar to the issue discussed in section 4.2.2.

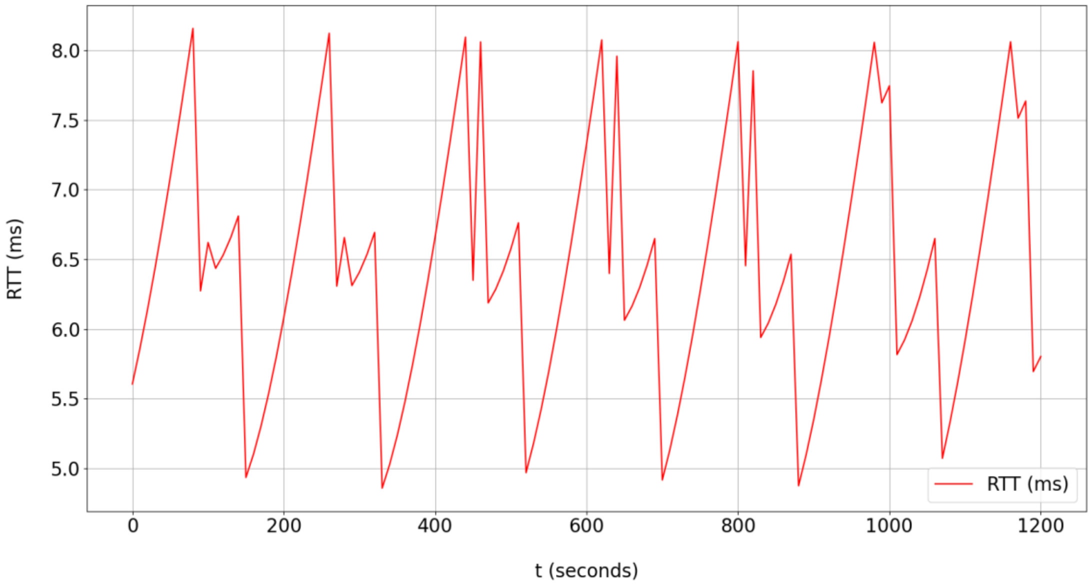

Figure 15. RTT from Chicago ground station to the closest of 20 greedy service placements (without migration) over 20-minutes at 1-second interval

It's important to note that if we assume all satellites have the capability to host the same service without replication, the RTT is mainly calculated based on the distance to the closest satellite. Figure 15 illustrates that under ideal conditions, where there is no traffic congestion, and no delays other than propagation delay, the average RTT is approximately 6.47ms. It is noteworthy that the Hypatia simulator has limitations in representing some level of multipath routing and congestion control. To sum up, assuming no significant computational overhead and traffic congestion, the greedy service placement will outperform the randomized approach. This experiment, nonetheless, provides us with an abstract understanding and performance of the service provisioning problem.

CHAPTER V

CONCLUSION

Our thesis explores the fundamental study of LEO satellite edge computing and utilizes the Hypatia simulator to analyze each problem. In Chapter 3, we discussed network topology, distance and routing, connection handover, and service provisioning, investigating their linkage. We address questions on each topic by developing Hypatia codebase. We found out that the four-neighbor topology remains consistent over time, while distance changes in a predictable arc. We discovered that in the simulator, routing is directly influenced by satellite distances. By gathering data from three constellations, we observe relatively low and consistent RTT values compared to terrestrial networks. Lastly, we experimented with two sets of algorithms to optimize service placement, revealing that the greedy algorithm outperforms the random algorithm. However, it is important to highlight that these experiments assume conditions with no computational overhead, significant traffic, or congestion, considering the limitations of the simulation software. Thus, our future work will incorporate these factors into the analysis.

REFERENCES

[1] Kassing, S., Bhattacherjee, D., Águas, A., Saethre, J., & A. Singla. (2020). Exploring the “internet from space” with Hypatia. ACM Internet Measurement Conf., New York, NY, USA: ACM, 2020. doi: 10.1145/3419394.3423635

[2] Kassing, S., Bhattacherjee, D., Águas, A., Saethre, J., & Singla, A. (2020). Exploring the “Internet from space” with Hypatia. In ACM IMC. [Online]. Available: https://github.com/snkas/hypatia

[3] Kempton, B., & A. Riedl. (2021). Network Simulator for Large Low Earth Orbit Satellite Networks. ICC 2021 - IEEE International Conference on Communications, Montreal, QC, Canada, 2021, pp. 1-6. doi: 10.1109/ICC42927.2021.9500439

[4] Kempton, B. (2020). Simulator for Large LEO-Satellite Communication Networks (SILLEO-SCNS). [Online]. Available: https://github.com/Ben-Kempton/SILLEO-SCNS

[5] Pfandzelter, T., & Bermbach, D. (2021). Edge (of the Earth) Replication: Optimizing Content Delivery in Large LEO Satellite Communication Networks. IEEE/ACM 21st International Symposium on Cluster, Cloud and Internet Computing (CCGrid), Melbourne, Australia, 2021, pp. 565-575, doi: 10.1109/CCGrid51090.2021.00066

[6] Pfandzelter, T., & Bermbach, D. (2021). Edge (of the Earth) Replication: Optimizing Content Delivery in Large LEO Satellite Communication Networks. [Online]. Available: https://github.com/pfandzelter/LLEOSCN-CDN-Sim

[7] Pfandzelter, T., Hasenburg, J., & Bermbach, D. (2021). Towards a Computing Platform for the LEO Edge. EdgeSys '21. Association for Computing Machinery, New York, NY, USA, 43–48. doi: 10.1145/3434770.3459736

[8] Su, Y., Liu, Y., Zhou, Y., Yuan, J., Cao, H., & Shi, J. (2019). Broadband LEO Satellite Communications: Architectures and Key Technologies. IEEE Wireless Communications, vol. 26, no. 2, pp. 55-61. doi: 10.1109/MWC.2019.1800299

[9] Chowdhury, P. K., Atiquzzaman, M., & Ivancic, W. (2006). Handover schemes in satellite networks: state-of-the-art and future research directions. IEEE Communications Surveys & Tutorials, vol. 8, no. 4, pp. 2-14. doi: 10.1109/COMST.2006.283818

[10] Park, S., & Kim, J. Trends in LEO Satellite Handover Algorithms. (2021). Twelfth International Conference on Ubiquitous and Future Networks (ICUFN), Jeju Island, Korea, Republic of, 2021, pp. 422-425. doi: 10.1109/ICUFN49451.2021.9528738

[11] Czesla, S., Schroter, S., Schneider, C. P., Huber, K. F., Pfeifer, F., Andreasen, D. T., & Zechmeister, M. (2019). PyA: Python astronomy-related packages. [Online]. Available: https://github.com/sczesla/PyAstronomy/

[12] Rhodes, B. C. PyEphem: Astronomical Ephemeris for Python. (2011). Astrophysics Source Code Library. ascl:1112.014. [Online]. Available: https://github.com/brandon-rhodes/pyephem/

[13] Starlink. (2023) Starlink. [Online]. Available: https://www.starlink.com/

[14] Amazon. (2023) Project Kuiper. [Online]. Available: https://www.amazon.jobs/en/teams/projectkuiper

[15] Telesat. (2023) Oneweb. [Online]. Available: https://www.telesat.com/

[16] Zhu, Y., Qian, L., Ding, L., Yang, F., Zhi, C., & Song, T. Software defined routing algorithm in LEO satellite networks. (2017). International Conference on Electrical Engineering and Informatics (ICELTICs), Banda Aceh, Indonesia, 2017, pp. 257-262. doi: 10.1109/ICELTICS.2017.8253282

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 - 1346 USA