A Comparative Evaluation of TCP Congestion Control Schemes over Low-Earth-Orbit (LEO) Satellite Networks

GEORGE BARBOSA, The University of Arizona, Tucson, AZ, United States

SIRAPOP THEERANANTACHAI, University of California, Los Angeles, Los Angeles, CA, United States

BEICHUAN ZHANG, The University of Arizona, Tucson, AZ, United States

LIXIA ZHANG, University of California, Los Angeles, Los Angeles, CA, United States

AINTEC '23: Proceedings of the 18th Asian Internet Engineering Conference (December 2023)
https://doi.org/10.1145/3630590.3630603
ISBN: 9798400709395

A Comparative Evaluation of TCP Congestion Control Schemes over Low-Earth-Orbit (LEO) Satellite Networks

George C.G. Barbosa
gcgbarbosa@arizona.edu
University of Arizona
Tucson, United States

Sirapop Theeranantachai
stheera@g.ucla.edu
University of California, Los Angeles
Los Angeles, United States

Beichuan Zhang
bzhang@cs.arizona.edu
University of Arizona
Tucson, United States

Lixia Zhang
lixia@cs.ucla.edu
University of California, Los Angeles
Los Angeles, United States

ABSTRACT

Low-Earth-Orbit (LEO) satellite constellations can provide global Internet connectivity with latency comparable to terrestrial networks, but their fast movement and frequent handovers result in highly dynamic connectivity changes, with an unknown impact on network congestion control design. In this paper we evaluate and compare the performance of four congestion control schemes over LEO satellite networks, especially their behavior in the face of path changes and delay variations caused by satellite movement. Our results show that, while dynamic connectivity leads to performance degradation in all the congestion control schemes we examined, the impact differs for different schemes: the loss-based schemes (NewReno and Cubic) suffer from high queuing delays, the delay-based scheme (Vegas) often achieves low throughput, and BBR can handle connectivity dynamics with moderate performance degradation. Furthermore, if a user terminal can only connect to one satellite at a time, the performance degradation is worse across all the examined schemes.

CCS CONCEPTS

• **Networks** → *Network performance evaluation*; **Network protocol design**; **Transport protocols**.

KEYWORDS

Named Data Networking, LEO satellite constellation

ACM Reference Format:
George C.G. Barbosa, Sirapop Theeranantachai, Beichuan Zhang, and Lixia Zhang. 2023. A Comparative Evaluation of TCP Congestion Control Schemes over Low-Earth-Orbit (LEO) Satellite Networks. In *Asian Internet Engineering Conference (AINTEC '23), December 12–14, 2023, Hanoi, Vietnam.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3630590.3630603

AINTEC '23, December 12–14, 2023, Hanoi, Vietnam
© 2023 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0939-5/23/12.
https://doi.org/10.1145/3630590.3630603

1 INTRODUCTION

Satellites have been used in communication for decades. Depending on their distance from the Earth's surface, satellites can be classified as Low-Earth-Orbit (LEO, <2,000 km), Medium-Earth-Orbit (MEO, <35,786 km), and Geosynchronous-Equatorial-Orbit (GEO, =35,786 km) [15]. The lower the orbit, the shorter the latency between the ground and the satellite, but the smaller the area each satellite can cover. Traditional network services based on GEO satellites have long latency and are unsuitable for interactive applications. LEO satellites can achieve latency comparable to terrestrial networks but require many more satellites for global coverage. In the last decade, technological advances in rocket launch, satellite miniaturization, and phased array antenna have made it possible to deploy thousands of LEO satellites for general Internet service. Companies such as SpaceX, Amazon, and OneWeb have been developing LEO satellite constellations for this purpose [9]. In particular, as of today, SpaceX's Starlink constellation has deployed more than 4,000 satellites and has been providing Internet service to end users for the last few years.

Existing satellite Internet access services, including Starlink, adopt the so-called "bent-pipe" model: a satellite receives signals from user terminals and sends them to an ISP ground station and vice versa. This model only uses ground-satellite links (GSLs); it requires both the sending and receiving ground devices be within the coverage area of the same satellite. The advent of LEO satellite constellations opens up the possibility of using laser-based inter-satellite links (ISLs) to relay traffic between satellites. This will expand Internet services to users far away from any ground station, and potentially provide shorter latency to general users due to the fact that light travels faster in space than in fiber [13]. Most recently, Starlink rported that it has starting using ISLs in its network [14]. Much research has been done on the topology construction, routing, and performance (mainly latency) of LEO constellation networks using both GSLs and ISLs [7].

GEO satellite links were notorious for TCP performance due to their long latency. LEO satellites alleviate this issue by their reduced latency by flying much closer to the earth surface. However, the fast movement of LEO satellites also introduces frequent handover and varying link delays. TCP congestion control mechanisms use packet loss and/or round-trip time (RTT) to detect network congestion and adapt the sending rate or window size accordingly [12]. This works well when the underlying end-to-end path is relatively stable like in terrestrial networks. As the path over LEO satellites varies, TCP cannot tell whether changes in RTT are due to path changes or network congestion, thus it might not be able to make proper adjustment. The actual impacts on TCP performance depend on the frequency and extent of the satellite path changes, and how different congestion control schemes detect and respond to those changes.

In this paper, we conduct packet-level simulations to systematically evaluate and compare the performance in terms of latency and throughput of four representative TCP congestion control schemes under the Starlink constellation with multi-hop ISLs. Our results confirm that LEO satellite movement decreases TCP's overall performance across the board, although the impact differs for different schemes. TCP NewReno and Cubic detect congestion by packet losses; they generally fill up the network buffer, resulting in the highest throughput and the highest queuing delay. TCP Vegas detects congestion by delay increase; it minimizes network queuing and achieves the best latency. However, its sensitivity to network delay sometimes leads to very low throughput. TCP BBR (Bottleneck Bandwidth and RTT) [5] shows the most resilient behavior due to its frequent probing of network bandwidth and minimal RTT, achieving a good balance between throughput and latency. Furthermore, we discover that the choice of GSLs, i.e., which satellite a ground device connects to, can significantly impact end-to-end latency and, therefore, TCP performance. More specifically, if a user terminal connects to its nearest satellite instead of the satellite on the shortest end-to-end path to the destination, sometimes it can experience much higher end-to-end latency. These results help us understand how TCP performs in this emerging network environment and highlight the need for new protocol designs.

The rest of the paper is organized as follows. Section 2 offers background information about LEO satellite constellations and TCP congestion control, Section 3 describes the basic simulation model, Section 4 evaluates and compares the performance of different congestion control schemes with different settings, Section 5 summarizes related work, and Section 6 concludes the paper.

2 BACKGROUND

A LEO satellite constellation can have thousands of satellites. These satellites may be deployed on different *shells*, where satellites on the same shell have the same altitude. For example, Starlink plans to deploy multiple shells of satellites with altitudes at 550 km, 560 km, 570 km, etc. Within each shell, satellites are placed on multiple orbital planes with the same *inclination*, which is the angle between the orbital plane and the Earth's equator. For example, a 90° inclination means the orbit goes directly above the north and south poles, while a 0° inclination means the orbit goes directly above the equator. Again, Starlink plans to deploy satellites on different inclinations, such as 53°, 70°, 97.6°, etc. Usually, satellites are distributed evenly on multiple orbits, and within each orbit, the satellites are also evenly placed [17]. In this paper, our simulations are based on Starlink's first shell of satellites, which has an altitude of 550 km, inclination 53°, and 1584 satellites distributed over 72 orbits. This shell of satellites provides good coverage over North America and Europe but not much over the polar areas [10].

Between satellites, laser-based optical links are used for communication. One challenge in establishing an ISL is to align the laser beam accurately tracing the fast-moving satellites. Satellites on the same orbit or neighboring orbits travel along similar directions. Therefore ISLs between them can be established and maintained with good quality. ISLs between satellites moving in different directions are much more challenging to maintain [6]. For example, consider an orbit of 53° inclination. Satellites on this orbit will travel northeast on one side of the Earth, then southeast on the other side of the Earth. Placing many such orbits evenly around the Earth, we will get a constellation that, viewed from any point in space, half of the satellites are moving northeast, and the other half is moving southeast. Establishing and maintaining ISLs between these two satellite groups is difficult [16]. While there are different ways of constructing inter-satellite network topology, due to the aforementioned concern, the most common one in literature is the so-called "+Grid", in which each satellite has four ISLs: two connect to neighboring satellites on the same orbit, and two connect to satellites on the adjacent orbits [10][16]. This is also the topology we use in our simulation.

With an altitude of 550 km, satellites travel at 27,320 km/h and complete a full circle around the Earth in 1.6 hours. This means that the ground-satellite handover will happen every few minutes, depending on the location of the ground device [4]. Since a constellation has thousands of satellites, in many cases, a ground device can have multiple satellites in its view at any given moment [4]. With the use of ISLs, the choice of satellite to connect to may have a significant impact on the end-to-end performance as we well show later. Existing research assumes that the ground device will choose the satellite that leads to the shortest end-to-end path to the final destination, which we refer to as the *shortest-path* strategy. Although this strategy brings better performance, it is unrealistic since users' traffic usually goes to many different destinations, and we cannot expect the user's antenna can switch satellites instantly with no penalty [1]. A more realistic strategy is for the user's antenna to connect to one satellite (e.g., the nearest), use it for a while, then hand over to another satellite, balancing end-to-end performance and GSL stability. Given there is no study on this topic for congestion control, in this paper, we compare the nearest-satellite strategy and the shortest-path strategy to understand the impact of GSL selection.

TCP congestion control is critical to the operation of the Internet. Over the years, many schemes have been developed. While some schemes are designed for specific network environments, such as wireless networks or data center networks, we are interested in the schemes that are designed for the general Internet and are representative of different design approaches. TCP NewReno is a typical "loss-based" scheme, which uses packet loss to detect congestion, after the network buffer has been filled up. TCP Cubic is the current default congestion control scheem used in Linux systems. It is also a "loss-based" design, but uses a different function to regulate its window size. TCP Vegas is a "delay-based" scheme, which detects congestion by increasing RTT and attempts to make full use of available bandwidth without filling up the network buffer, which minimize packet queuing delay in the network. TCP BBR is a more recent development and has been deployed in Google servers and clients. It frequently measures the bottleneck bandwidth and minimal RTT to obtain the bandwidth-delay product (BDP) of the path, and regulates TCP's sending rate such that the amount of outstanding data will match the path's BDP [5]. Our work evaluates and compares the above four schemes to observe their performance in the LEO satellite network, where path and latency changes are inherent to the underlying network rather than a result of congestion.

3 SIMULATION MODEL

We use Hypatia [10], a LEO simulator that integrates satellite movement with ns-3, to run simulations on the phase-1 deployment of the Starlink constellation. There are 1584 satellites evenly distributed over 72 orbits with an altitude 550 km and inclination 53°. Each satellite has four ISLs: two connecting the two adjacent satellites on the same orbit, and two connecting to satellites on two neighboring orbits, illustrated in Figure 1. This is the most common LEO satellite network topology used in the literature. The most significant benefit is that it avoids ISL handovers. Because ISLs are only used between neighboring satellites, and in large constellations such as Starlink, these neighboring satellites are always in communication range, there is no ISL handover. However, the delay of each ISL still changes because two neighbor satellites will move closer or farther from each other as they orbit the Earth. These varying link delays are computed based on the distance between satellites in the simulator when packets traverse these links. To focus on the impacts of path changes, we assume perfect laser alignment of the ISL and no link errors.

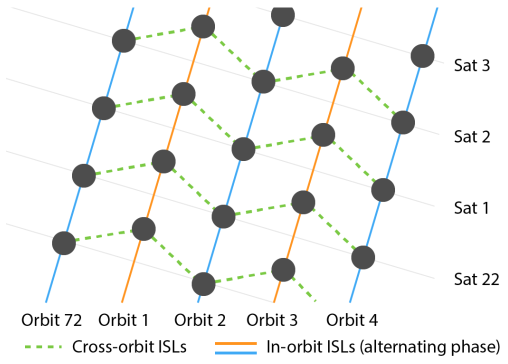

Figure 1: The Grid Topology of Satellite Networks

An end-to-end path between two ground devices comprises a radio up-link from the source device to the ingress satellite, followed by zero or more laser ISLs, and the egress radio down-link to the destination device [10]. In order to finish the simulations in reasonable amount of time, we set ISL bandwidth to 1 Gbps and GSL 10 Mbps. It is reported that Starlink's ISLs may have up to 100Gbps capacity and it caps user downlink at 200Mbps [14].

Given a set of satellites in range, we compare two satellite selection strategies for ground devices:

Nearest-satellite strategy: The ground device connects to the nearest satellite. The ISL part of the path is the shortest path between ingress and egress satellites, but the end-to-end path between the source and destination may not be the shortest.

Shortest-path strategy: The ground device connects to the satellite that leads to the shortest end-to-end path to the destination device.

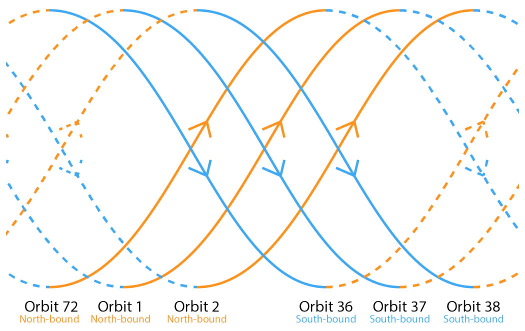

Figure 2: Northbound and Southbound Orbits

The shortest-path strategy gives shorter latency by selecting ingress and egress satellites on the same orbit or close-by orbits to avoid extra cross-orbit ISL hops. For the traffic of different destinations, it may need to use different ingress/egress satellites, which is not realistic if the ground device can only connect to one satellite at a time. With the nearest-satellite strategy, a ground device uses the nearest satellite for traffic to all destinations; it switches satellites only when another satellite becomes the nearest. The ingress and the egress satellites may be on orbits of different orientations (northbound vs. southbound in Figure 2), which can significantly increase the number of cross-orbit ISLs and end-to-end latency.

4 COMPARISON OF CONGESTION CONTROL SCHEMES

4.1 Overview of the simulation

As shown in figure 3, we choose four pairs of ground sources and destinations to represent different path orientations relative to the satellite orbit's inclination angle:

- Sao Paulo (red), northbound along the satellite orbit

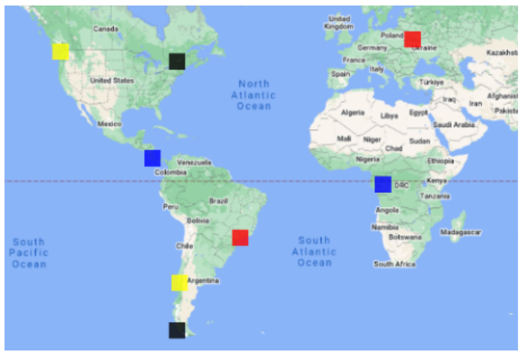

Figure 3: The locations of ground sources and destinations: Sao Paulo-P1 (red), San Jose-P2 (blue), Montreal-P3 (black) and Victoria-P4 (yellow)

- Victoria (yellow), southbound along the satellite orbit
- San Jose (blue) along the equator
- Montreal (black) along the north/south poles.

The intuition is that sending data along the satellite orbit inclination may enjoy a more direct ISL path, but sending data across different orbit orientations may take a longer ISL path – we want to capture this potential impact in our evaluation.

In each pair, the source and destination are about 11,000 km apart. To compare it with a shorter distance, we also use four other pairs with the same orientation but half of the distance, i.e., 5,500 km. In total, we used eight source/destination pairs in our simulations.

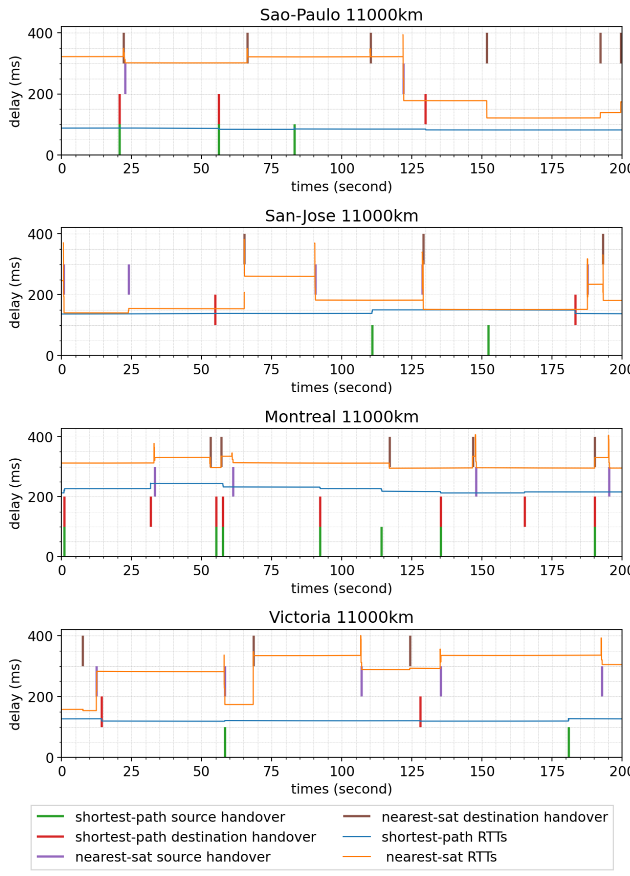

Figure 4: RTTs of four paths of 11,000 km

Before running congestion control schemes, we examine how satellite movement may affect the latency of the end-to-end path by pinging between the eight pairs of interests. We send a UDP packet every millisecond without any other traffic and collect the delay results in figure 4. There are two observations. First, the path delay varies over time as satellites move. In particular, when there is a GSL handover (noted by short bars in the figure), the path delay will change abruptly. Second, comparing the nearest-satellite and the shortest-path strategies, the former has a longer delay, sometimes almost doubling what the latter has. While the nearest satellite is expected to have a longer delay, it is surprising to see the difference can be so significant. Further examination shows that the dramatic increase in path delay often happens when the ingress and egress satellites move in different directions, one northbound and one southbound. When this happens, the path has to cross many orbits (e.g., go around the globe) to reach the destination because there is no direct link connecting satellites moving in different directions. This is supported by the observation that the delay difference between the two strategies is much more pronounced in the Sao-Paulo and Victoria paths than in the other two. The Sao-Paulo path is along the northeast orbit direction, and the Victoria path is along the southeast orbit direction. The shortest-path strategy will always use satellites in the direction that yields the better delay of the two, but if nearest-satellite chooses satellites in a worse direction, the delay difference will be the most. In reality, nearest-satellite is more practical for end-user terminals; thus, it may experience longer delay and more delay variance. This result highlights the importance of satellite selection by ground devices.

The rest of this section will focus on TCP congestion control results. We simulate loss-based (TCP NewReno and Cubic), delay-based (TCP Vegas) and model-based (TCP BBR) congestion control schemes by transferring continuously transferring data for 200 seconds and recording the RTT experienced by TCP packets and the amount of data successfully received. In analyzing the results, we examine various TCP parameters, including congestion window size (CWND), queue size, and throughput.

4.2 Performance with Shortest Paths

4.2.1 11,000 km pairs. Table-1 shows the throughput, i.e., the amount of data successfully transferred within 200s, made by different congestion control schemes over LEO network. For comparison, for each pair of locations, we also simulated the same file transfer in wired networks with the same number of router hops and the same average RTT. The percentages in parentheses next to the LEO values represent the percentage change compared to the wired network results. For example, in the Sao Paulo path, TCP NewReno transferred 1992.99 megabits in the wired network and 1920.01 megabits in the LEO network, which is a 3.8% reduction in throughput.

The main takeaway from this table is that in almost all cases TCP will experience performance degradation in LEO satellite network compared to wired network. Among different congestion control schemes, TCP NewReno and Cubic achieve the highest throughput, Vegas has the lowest throughput, and BBR is slightly worse than NewReno and Cubic. Vegas also showed the biggest variation across different location pairs.

Figure 5 compares the RTT in milliseconds across these cases. For example, the Ping RTT value (without congestion control) for the Sao Paulo location pair is 84 ms, with a range of max and min values from 81-89ms. NewReno and Cubic exhibit large delays: 156ms and 173ms respectively. Vegas has the lowest RTT at 91ms average. BBR's 97ms is larger than Vegas' but very close.

Loss-based schemes such as NewReno and Cubic detect congestion by packet loss, therefore they need to fill up the link bandwidth and router buffer before being able to detect congestion and back off. This results in persistent packet queues in the network with two implications: high throughput due to continuous supply of packets, and long end-to-end delays due to queuing. The delay-based

Table 1: Data (in Mb) transferred over 200s, distance 11,000 km

	NewReno		Cubic		Vegas		BBR	
Pair	wired	LEO	wired	LEO	wired	LEO	wired	LEO
Sao Paulo	1992.99	1920.01(-3.8%)	1997.65	1921.06(-4.0%)	1727.59	1827.25(+5.5%)	1881.86	1816.95(-3.5%)
San Jose	1977.29	1856.36(-6.5%)	1995.68	1918.09(-4.0%)	1789.71	1116.27(-37.7%)	1862.17	1779.95(-4.6%)
Montreal	1917.44	1842.74(-4.0%)	1990.88	1912.05(-4.1%)	1570.98	644.29(-59%)	1823.09	1835.06(+0.7%)
Victoria	1988.19	1914.12(-3.8%)	1996.31	1918.74(-4.0%)	1942.33	1783.68(-8.2%)	1806.93	1808.39(+0.1%)

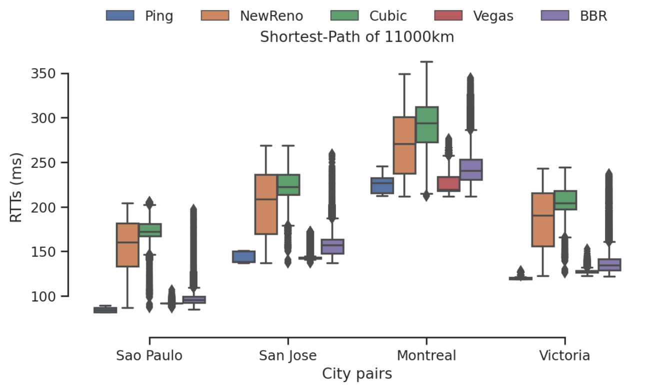

Figure 5: RTT distribution, distance 11,000 km, shortest-path strategy

scheme such as Vegas detects congestion by increasing delay. It aims to fill up the link bandwidth without building up a queue. It is able to achieve shortest delay, but the algorithm is very sensitive to the delay change. With many path delay changes in LEO satellite network, Vegas often gets confused and sends data slower than it should, resulting in low throughput. BBR also aims to fill up the link bandwidth without causing persistent queue but does it differently. It frequently probes the network for its minimal RTT and bottleneck bandwidth, and adjusts TCP's sending rate to match bandwidth-delay product. Although it is affected by LEO satellite's movement as well, its frequent probing makes it adjust quickly to the new path. Considering both throughput and delay, BBR gives the best tradeoff among the four schemes in our evaluation.

4.2.2 5,500km pairs. Table 2 compares the throughput of the congestion control schemes on shorter paths at 5500 km (half the distance of the experiments on Section 4.2.1).

Table 2: Data (in Mb) transferred over 200s, distance 5,500 km, shortest-path strategy

Pair	NewReno	Cubic	Vegas	BBR
Sao Paulo	1922.62	1923.09	1435.72	1879.28
SanJose	1921.46	1922.08	1125.63	1870.85
Montreal	1914.88	1920.61	1297.19	1820.78
Victoria	1922.28	1922.72	1160.50	1876.50

For example, under the SanJose pair, NewReno and Cubic algorithms over-performed Vegas and BBR, with BBR performing slightly behind while Vegas was about 25% worse.

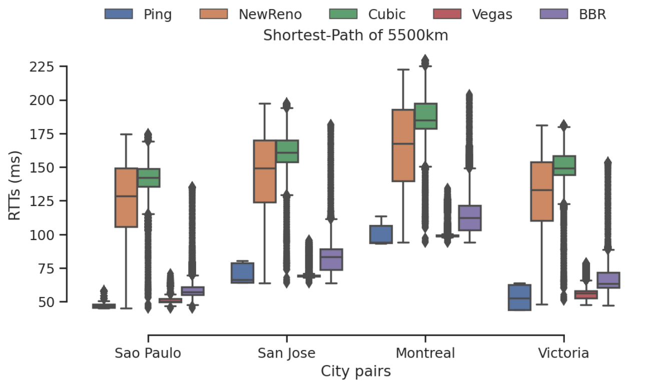

Figure 6: RTT distribution, distance 5,500 km, shortest-path strategy

Figure 6 shows the four congestion control schemes' RTTs over four pairs of locations that are 5500km apart. One can see that NewReno and Cubic's mean latency values are about three times the ones observed for Vegas and BBR, and that Vegas has a much smaller latency variation than BBR. These results are consistent with the results obtained for 11,000km pairs.

4.3 Performance with the Nearest Satellite

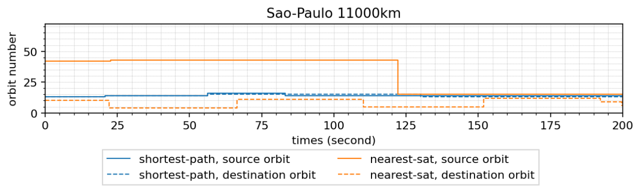

Figure 7: Orbit numbers between two ground devices 11,000 km apart

Instead of connecting to a satellite that minimizes the end-to-end delay between two ground devices, with the nearest-satellite strategy, a ground device picks the nearest satellite to connect to. This strategy change leads to a drastic increase in RTTs and the number of hops, hence significantly impacts the performance of the congestion control schemes. Take Sao Paulo at 11000km (Figure 7) as an example. During the first 122 seconds of the simulation, the RTTs of this strategy are consistently three times higher than the RTTs of the shortest-path strategy because of sub-optimal satellite selection.

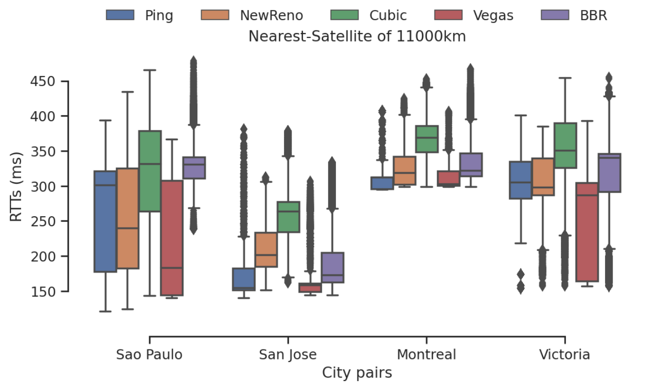

Figure 8: RTT distribution, distance 11,000 km, nearest-satellite strategy

Figure 8 depicts the RTT distribution under various congestion control schemes with the nearest-satellite strategy. We added the ping results as a baseline for comparison.

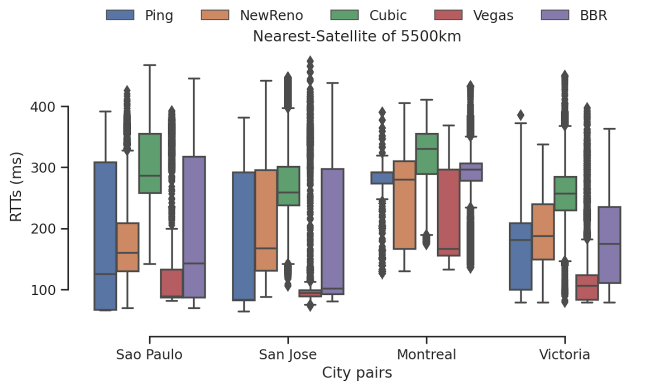

Figure 9: RTT distribution, distance 5,500 km, nearest-satellite strategy

Similarly, Figure 9 shows that Vegas has the lowest RTT values in all cases. BBR is the second best. NewReno and Cubic have significantly higher RTT.

Table 3 compares the throughput among the four pairs at 11,000 km. Table 4 compares the throughput for the same four pairs, but at a distance of 5,500 km. We can see that the general observations still hold: (1) loss-based schemes such as NewReno and Cubic achieve high throughput but suffers from large latency, (2) delay-based schemes such as Vegas achieves shortest delay but suffers from low throughput or variation in throughput, and (3) BRR makes a good tradeoff with good throughput and good delay.

Table 3: Data (in Mb) transferred over 200s, distance 11,000 km, nearest-satellite strategy

Pair	NewReno	Cubic	Vegas	BBR
Sao Paulo	1368.54	1872.43	1124.36	896.76
San Jose	1628.87	1902.00	522.62	1807.49
Montreal	1220.80	1899.28	859.25	1805.64
Victoria	1301.54	1893.24	152.47	1746.70

Table 4: Data (in Mb) transferred over 200s, distance 5,500 km, nearest-satellite strategy

Pair	NewReno	Cubic	Vegas	BBR
Sao Paulo	1244.25	1862.03	1071.29	1653.81
San-Jose	1724.94	1838.47	300.30	1746.87
Montreal	1114.34	1860.15	622.12	1801.20
Victoria	1393.29	1886.12	238.96	1746.85

4.4 Case Study of TCP Vegas

To fully understand how LEO satellite dynamics impact TCP congestion control algorithms, we analyzed the changes of window size and RTT in response to the variation of path delay. Here we will present one case study of TCP Vegas since it is the most affected among the four algorithms in our study.

This case study uses the Victoria location pair with the distance of 5,500 km. During the simulation period of 200s, there were eight path changes between the two ground devices, at 12.6s, 58.3s, 66.6s, 107s 130.0s, 135.4s, 190.7s, and 192.9s.

TCP Vegas uses the RTT reported by TCP to predict congestion. It first sets *BaseRTT* to the shortest RTT measured, and updates it whenever a measured RTT is smaller than the current *BaseRTT*. TCP Vegas uses *BaseRTT* to calculates the *ExpectedRate*, and compares it with the *ActualRate*.

$$\text{actualRate} = \text{cwnd}/\text{RTT}$$
$$\text{expectedRate} = \text{cwnd}/\text{BaseRTT}$$
$$\text{diff} = \text{expected} - \text{actual}$$

TCP Vegas uses three configurable parameters, α, β, and γ, to manage congestion control. If diff $< \alpha$, Vegas will increase CWND. If diff $> \beta$, CWND will decrease [12]. In our experiments, we adjusted α from 2 to 4 to increase the bandwidth utilization of Vegas.

Figures 10 and 11 show CWND and RTT over time. Between 0ms and 12.6ms, the network RTT remained stable around 80ms. The first acknowledgement reported RTT=79ms, which became the *BaseRTT*. Throughout the simulation, Vegas maintained a steady *BaseRTT* of 79ms for the entire 200 seconds. CWND started at 10 and increased to 69 between 0ms and 4.3ms (Figure 10). At 4.3ms, RTT=83ms, and the difference between expected rate and actual rate was exactly 4, thus cwnd remained unchanged. At 12.6s, a path change happened, and the new path had RTT of 208ms, which was 2.6 times of the previous one. TCP's moving average

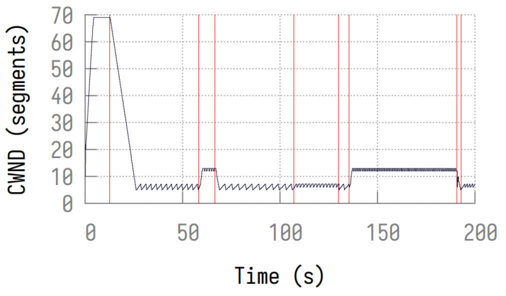

Figure 10: TCP Vegas CWND over time. Vertical red bars are path changes.

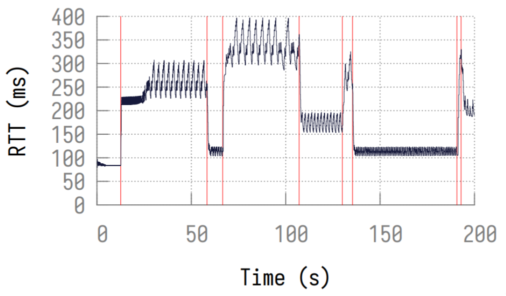

Figure 11: RTT of Vegas over time. Vertical red bars are path changes.

smoothed out this change. Still, once the longer path latency got reflected on TCP's RTT measurement, Vegas started decreasing the congestion window in response to the increased RTT. This is the primary example that conventional TCP congestion control algorithms cannot differentiate congestion from path change when the RTT increases. As a result, from 12.6s until 26.07s, TCP Vegas gradually decreased CWND down to 6, causing low throughput.

The new path at 58.3s was shorter than the previous one; the new path RTT was 99.32 ms. With this value, Vegas had room to increase CWND up to 13. The next path change at 66.6s made RTT to 273.78ms. Vegas decreased CWND again down to 6 to account for the extra delay.

The new path at 107.1s had RTT of 145.89 ms; since *BaseRTT* stayed constant at 79ms throughout the simulation, the difference was too significant, and CWND remained steady between 6 and 7. The change at 130.0s made the path longer at 225ms, but CWND did not change since it was already very low. The path change at 135.4 decreased RTT to 99ms. Again, Vegas adjusted CWND up to 13.

The two last path changes at 190.7s and 192.9s increased RTT to 292ms and 181ms, respectively. Since *baseRTT* was still at 79ms, CWND decreased to 6 again and remaind steady until the end of the simulation.

We can understand from the algorithm that CWND and RTT are crucial in TCP Vegas. In LEO, path changes and satellite movement affect the path latency between sender and receiver, which can affect the RTT and cause TCP Vegas to misinterpret the network conditions. Even if CWND remains constant, *Diff* can change due to satellite movement. This can lead to inaccurate congestion control decisions by TCP Vegas, causing under-utilization or over-utilization of network links.

5 RELATED WORK

LEO satellite networks have recently attracted much interest from the networking community due to its unique characteristics and the potential of using multi-hop ISLs. One focus of LEO research is quantifying the delay and comparing it to that of terrestrial networks. The authors in [3] provide an in-depth analysis of the delay of the Starlink network. They study the connectivity of Starlink's spatial and temporal characteristics and geographic variability. We have also seen interest in studying the design of the constellation topology and its implications [2][8]. Another research interest is building tools for other researchers to study LEO networks. In [7], a simulator was created using FCC filings to explore the use of laser links for a network and assess routing. Initial evaluation indicates the potential of low-latency communication and multipath opportunities. Another simulation tool is Hypatia [10], used in this paper. It aims to enable research in this space by integrating satellite movement modeling with popular packet-level simulator ns-3. The work also investigates latency and link utilization fluctuations over time in Starlink constellation. There's also work in future internet architecture protocols, such as Named Data Networking (NDN). In [11], the authors investigate the potential of using NDN in large LEO satellite constellations. They discuss the challenges of networking in this scenario and how NDN's architectural benefits, such as adaptive forwarding, in-network caching, off-the-grid communication, data mule service, in-network/edge computing, mobility support, and data-centric security can provide an effective and efficient networking system specifically for LEO constellations. To the best of our knowledge, none of the literature studies have systematically compared TCP congestion control schemes under LEO constellations.

6 CONCLUSIONS

LEO satellite networks are a brand new network environment in that the network infrastructure itself, comprised of satellites, ISLs, and GSLs, is constantly changing. Existing Internet protocols were designed with terrestrial networks in mind, and whether they will work or how well they will work over LEO satellite networks is unclear. This paper is the first step towards a complete understanding of the impacts of LEO satellite movement on TCP congestion control. Through systematic simulations and detailed analysis, we showed how underlying path changes may sometimes confuse TCP and cause performance degradation. The loss-based schemes are able to achieve the highest throughput at the expense of high delay, the delay-based scheme can achieve the shortest delay but suffers from low and unstable throughput. BBR seems to be able to adapt to the LEO environment and achieves both good throughput and delay. However, we must remember that our simulations have several simplifying assumptions, such as perfect GSL channels and perfect ISL alignment. It is our future work to improve the simulation model and evaluate the impacts of various factors from the operational environment. In addition, our work showed the importance of network design of LEO satellite constellations, i.e., the selection of GSLs and ISLs, and calls for further research into this topic.

REFERENCES

[1] Alexia Auddino, Anna Barraqué, Oana Hotescu, Jérôme Lacan, José Radzik, and Emmanuel Lochin. 2022. The Nearest Is Not The Fastest: On The Importance Of Selecting In/Out Routing Hops Over A Satellite LEO Constellation. In *2022 IEEE 96th Vehicular Technology Conference (VTC2022-Fall)*. 1–5. https://doi.org/10.1109/VTC2022-Fall57202.2022.10012796

[2] Debopam Bhattacherjee and Ankit Singla. 2019. Network Topology Design at 27,000 Km/Hour. In *Proceedings of the 15th International Conference on Emerging Networking Experiments And Technologies* (Orlando, Florida) *(CoNEXT '19)*. Association for Computing Machinery, New York, NY, USA, 341–354. https://doi.org/10.1145/3359989.3365407

[3] Vaibhav Bhosale, Ahmed Saeed, Ketan Bhardwaj, and Ada Gavrilovska. 2023. A Characterization of Route Variability in LEO Satellite Networks. In *Passive and Active Measurement*, Anna Brunstrom, Marcel Flores, and Marco Fiore (Eds.). Springer Nature Switzerland, Cham, 313–342.

[4] Shkelzen Cakaj. 2021. The Parameters Comparison of the "Starlink" LEO Satellites Constellation for Different Orbital Shells. *Frontiers in Communications and Networks* 2 (2021). https://doi.org/10.3389/frcmn.2021.643095

[5] Neal Cardwell, Yuchung Cheng, C. Stephen Gunn, Soheil Hassas Yeganeh, and Van Jacobson. 2017. BBR: Congestion-Based Congestion Control. *Commun. ACM* 60, 2 (jan 2017), 58–66. https://doi.org/10.1145/3009824

[6] Aizaz U. Chaudhry and Halim Yanikomeroglu. 2021. Free Space Optics for Next-Generation Satellite Networks. *IEEE Consumer Electronics Magazine* 10, 6 (2021), 21–31. https://doi.org/10.1109/MCE.2020.3029772

[7] Mark Handley. 2018. a *(HotNets '18)*. Association for Computing Machinery, New York, NY, USA, 85–91. https://doi.org/10.1145/3286062.3286075

[8] Mark Handley. 2019. Using Ground Relays for Low-Latency Wide-Area Routing in Megaconstellations. In *Proceedings of the 18th ACM Workshop on Hot Topics in Networks* (Princeton, NJ, USA) *(HotNets '19)*. Association for Computing Machinery, New York, NY, USA, 125–132. https://doi.org/10.1145/3365609.3365859

[9] Xingchi He, Urs Hugentobler, Anja Schlicht, Yufeng Nie, and Bingbing Duan. 2022. Influence of Ground Station Network Distribution on Orbit Accuracy of Low Earth Orbit (LEO) Satellites. (2022). https://doi.org/10.5194/egusphere-egu22-1607

[10] Simon Kassing, Debopam Bhattacherjee, André Baptista Águas, Jens Eirik Saethre, and Ankit Singla. 2020. Exploring the "Internet from Space" with Hypatia. In *Proceedings of the ACM Internet Measurement Conference* (Virtual Event, USA) *(IMC '20)*. Association for Computing Machinery, New York, NY, USA, 214–229. https://doi.org/10.1145/3419394.3423635

[11] Teng Liang, Zhongda Xia, Guoming Tang, Yu Zhang, and Beichuan Zhang. 2021. NDN in Large LEO Satellite Constellations: A Case of Consumer Mobility Support. In *Proceedings of the 8th ACM Conference on Information-Centric Networking* (Paris, France) *(ICN '21)*. Association for Computing Machinery, New York, NY, USA, 1–12. https://doi.org/10.1145/3460417.3482970

[12] Larry L. Peterson and Bruce S. Davie. 2011. *Computer Networks, Fifth Edition: A Systems Approach* (5th ed.). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

[13] Radhika Radhakrishnan, William W. Edmonson, Fatemeh Afghah, Ramón Martínez Rodríguez-Osorio, Frank Pinto, and Scott C. Burleigh. 2016. Survey of Inter-Satellite Communication for Small Satellite Systems: Physical Layer to Network Layer View. *IEEE Communications Surveys & Tutorials* 18 (2016), 2442–2473.

[14] Starlink. [n. d.]. Starlink's twitter post on its space lasers. Accessed: 2023-11-19.

[15] Krasimir Terziev and Dimitar Karastoyanov. 2020. The Impact of Innovation in the Satellite Industry on the Telecommunications Services Market. *Problems of Engineering Cybernetics and Robotics* (2020). https://doi.org/10.7546/pecr.73.20.03

[16] Rui Xue, Tong Wang, and Huaiyu Tang. 2020. A Novel Chip Pulse Employed by Ranging Code Based on Simultaneous Transmitting CPM Modulation and PN Ranging in Inter-Satellite Links of GNSS. *IEEE Access* 8 (2020), 132860–132870. https://doi.org/10.1109/ACCESS.2020.3010835

[17] Kaan Çelikbilek, Zainab Saleem, Ruben Morales Ferre, Jaan Praks, and Elena Simona Lohan. 2022. Survey on Optimization Methods for LEO-Satellite-Based Networks with Applications in Future Autonomous Transportation. *Sensors* 22, 4 (2022). https://doi.org/10.3390/s22041421