GLOBECOM 2024 - 2024 IEEE Global Communications Conference | 979-8-3503-5125-5/24/$31.00 ©2024 IEEE | DOI: 10.1109/GLOBECOM52923.2024.10901096

FlexSATE: Flexible and Distributed Traffic Engineering with Supervised Learning in Ultra-Dense Low-Earth-Orbit Satellite Networks

Xiaoyu Liu*, Zitian Zhang†, Xiaohan Qin*, Haibo Zhou* and Lian Zhao‡
*School of Electronic Science and Engineering, Nanjing University, Nanjing, China, 210023.
†School of Information and Electronic Engineering, Zhejiang Gongshang University, Hangzhou, China, 314423.
‡Department of Electrical, Computer, Biomedical Engineering, Toronto Metropolitan University, Toronto, Canada, ON M5B 2K3.
Email: xyliu0119@163.com, zitian.zhang@mail.zjgsu.edu.cn, xhderemail@smail.nju.edu.cn, haibozhou@nju.edu.cn, l5zhao@torontomu.ca

***Abstract*—The ultra-dense low earth orbit (UD-LEO) satellite network is being vigorously developed due to its great potential in providing global coverage and services. For the sake of improved network performance in resource-constrained satellite networks, multipath schemes are being explored. However, state-of-the-art multipath routing algorithms face the challenge when dealing with highly dynamic satellite network features (i.e., frequent traffic variation, link failures) and fail to exploit the simple grid topology to design fast yet efficient traffic engineering (TE) approaches. In this paper, we propose a novel distributed TE scheme called Flexible Satellite Traffic Engineering (FlexSATE), which leverages global path computation coupled with distributed local routing decisions to improve the overall load balancing performance for ultra-dense LEO satellite networks. By constructing a minimum-hop binary tree (MHBT), we propose an MHBT-based k-segment Routing algorithm, which is capable of promptly discovering routing paths with low latency, high diversity, and good load balancing. To further enhance network transmission performance, we employ supervised learning into dynamic rate adaption, where FlexSATE employs centralized offline learning to derive insights from the globally optimal routing strategy and utilizes distributed deployment to predict the optimal distribution of traffic in real time. Our simulation results on a real-world typical Walker-delta type LEO constellation with 720 satellites show that FlexSATE outperforms some existing approaches with superior robustness and flexibility.**

***Index Terms*—LEO satellite network, multipath routing algorithm, rate adaptation, traffic engineering, supervised learning.**

I. INTRODUCTION

DUE to its great potential in providing services of low latency, high reliability, and ubiquitous connectivity, the UD-LEO satellite network is promoted to be a nontrivial infrastructure in constructing future space-terrestrial integrated networks [1]. Considering its high mobility, resource-constrained and fault-prone nature, which is fundamentally different from terrestrial networks, how to implement effective network management and control in LEO satellite networks becomes an open issue. In traditional terrestrial networks, Internet Service Providers use a significant network operation of traffic engineering to enhance network performance and optimize resource utilization by configuring routing paths and controlling traffic distribution. Considering more limited resources, such as link bandwidth, compared to terrestrial networks, applying TE techniques to LEO satellite networks becomes particularly valuable to achieve efficient flow management and resource utilization. However, most previous studies only design TE systems in traditional terrestrial networks [2], which cannot be directly applied to LEO satellite networks.

Due to dynamic network states such as frequent traffic fluctuations and intermittent transmission links in the context of LEO satellite networks [3], new challenges are brought to TE system design in terms of path selection and rate adaptation. In UD-LEO satellite networks, it is a relatively slow and costly operation of updating end-to-end forwarding paths, which should be executed infrequently, such as when topology changes occur. Hence, the key challenge is how to compute a set of candidate paths for each satellite pair in the first phase, capable of flexibly adjusting to different traffic scenarios. Otherwise, regardless of the traffic distribution solutions used, the network performance and resource utilization may significantly deviate from the optimal value, leading to suboptimal results. The rate adaptation phase involves utilizing information about current demands and failures to schedule incoming traffic onto candidate paths, aiming to optimize network performance. Since updating path weights is a relatively fast operation, rate adaptation can be performed continuously to accommodate changes in states. Centralized schemes [4], [5] typically employ periodic routing to achieve load balancing on links by formulating and solving optimization problems. However, these approaches encounter challenges in terms of scalability and timeliness, especially when dealing with large-scale networks. As an alternative solution, distributed schemes [6] is introduced to the control scalability issue, however, they can only obtain locally optimal but globally suboptimal solutions. Therefore, the key challenge is how to effectively optimize the global TE objective in a distributed TE problem, where each satellite in the network independently determines its own local routing strategy.

Motivated by the aforementioned challenges, this paper aims to develop a Flexible and Distributed Traffic Engineering approach for improving the network performance and resource utilization of LEO satellite networks. Our main contributions can be summarized as follows:

- We propose a new TE system called FlexSATE for large-scale LEO satellite networks, which aims to achieve efficient network management and control. The centralized controller computes forwarding paths considering cumulative path occupation from a global view. The rate adaptation is performed distributedly in each LEO satellite, which enables more flexible adaptation to dynamic network state changes.
- We develop a simple yet efficient multipath routing algorithm for the grid topology of Walker-delta type LEO satellite networks to obtain forwarding paths of low latency, high diversity, and good load balancing. By constructing a MHBT, we propose an MHBT-based k-segment Routing algorithm, which flexibly restricts the path search to the minimum hop path (MHP) region and is capable of promptly discovering multiple paths.
- We propose a customized supervised learning (SL) approach coupled with Graph Neural Network (GNN) architecture that predicts the optimal multipath traffic distribution for each satellite. To achieve global optimal performance, FlexSATE utilizes offline centralized learning guided by optimal multipath traffic split ratios obtained from a modified Multi-Commodity Flow (MCF) problem. Each LEO satellite deploys a well-trained model to make local routing decisions in a distributed manner, enabling near-optimal TE performance with low computation and communication overhead.

The rest part is organized as follows. Section II describes the system model. Section III and IV give the design of routing selection and rate adaptation, respectively. Simulation results are provided in Section V. Section VI concludes the paper.

II. SYSTEM MODEL

Most of the LEO constellation networks adopt the Walker-type constellation, where all satellites have the same altitude h and inclination α. M orbit planes are evenly distributed along the equator and N satellites are evenly distributed in each plane. Satellites are connected through inter-satellite links (ISLs) to form communication network. We model the LEO constellation networks as a graph $G = (V, E)$, where V denotes the set of satellite nodes and E denotes the set of ISLs. Each satellite establishes 4 links, two for intra-orbit links and two for inter-orbit links. For Walker-delta type LEO constellation, a mesh-like network topology with a relatively stable ISL connection relationship can be formed, unless there is an unpredictable link outage. Furthermore, we consider scenarios with many traffic demands. In this context, a traffic demand represents the aggregate traffic entering the network from an entry satellite and exiting the network from an exit satellite. A traffic matrix (TM) encompasses the collection of demands between all possible pairs of different satellites, with the traffic volume for each demand varying over time. The objective of TE is to efficiently deliver these dynamic traffic demands, aiming to optimize a specific objective, such as minimizing the maximum link utilization (MLU).

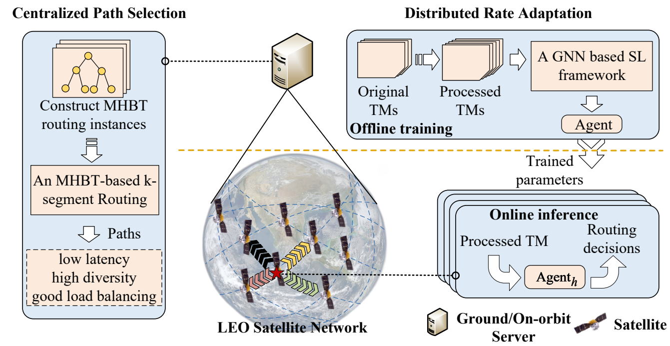


Fig. 1: Overview of FlexSATE's system design.

As shown in Fig. 1, the proposed FlexSATE consists of two main building blocks: path selection and rate adaption. We deploy the relatively slow and costly operation of path selection in the centralized controller (e.g., MEO satellite, terrestrial server) in order to select robust forwarding paths for all possible satellite pairs from a global perspective. By constructing a minimum-hop binary tree (MHBT), we propose an MHBT-based k-segment Routing algorithm, which is capable of promptly discovering routing paths with low latency, high diversity, and good load balancing, as later shown in Section IV. Due to the relatively fast operation of rate adaptation, it can be performed continuously as network conditions evolve. The system has the capability to update weights to rebalance the load when traffic demands change. We focus on tackling the distributed TE problem, where each satellite in the network independently makes local routing decisions while optimizing the global TE objective. To effectively solve it, we employ a data-driven approach rather than relying on a model-based one. Satellites in the network measure the local traffic demand and flood the information over the network. A GNN-based model, denoted as an agent, is distributed on each satellite in the network. The global optimal routing solutions are obtained by solving a variation of MCF as the training target. Each satellite agent maintains a GNN that approximates the mapping from network status inputs to routing decisions outputs. In essence, routing decisions determine the allocation of traffic demands across multiple forwarding paths that connect the ingress satellite to the egress satellite.

III. SIMPLE YET EFFICIENT MULTIPATH ROUTING

A. Minimum-hop Path Region

Using the four-ISL pattern, the LEO satellite network establishes a resilient Mesh-like topology that can be maintained even as the distances between ISLs vary. When considering two satellite nodes, A and B, packets traveling along the minimum hop path from A to B have up to two alternative directions at each node. The minimum hops, denoted as

H_{MH}, between A and B are determined by the sum of inter-plane hops (H_x) and intra-plane hops (H_y), such that $H_{MH} = H_x + H_y$. Multiple minimum hop paths may exist between A and B, forming a region known as the MHP region.

In a bidirectional Manhattan street network (MSN), the minimum hop count can be obtained by comparing the orbit numbers and satellite numbers of the two satellites. Specifically, when the satellite ID U that ranges from 0 to $M*N-1$ is known, the orbit number and satellite number can be obtained by $(U/N, U \bmod N)$. From the source (s_x, s_y) to the destination (d_x, d_y), we have $H_x = \min\{|s_x - d_x|, M - |s_x - d_x|\}$, where, when $|s_x - d_x| \le \frac{M}{2}$, if $d_x > s_x$, the path direction of H_x is $\vec{H}_x = +x$, i.e., the positive direction of x-axis; if $d_x < s_x$, the path direction of H_x is $\vec{H}_x = -x$, i.e., the negative direction of x-axis. When $|s_x - d_x| > \frac{M}{2}$, if $d_x > s_x$, the path direction of H_x is $\vec{H}_x = -x$; if $d_x < s_x$, the path direction of H_x is $\vec{H}_x = +x$. We have $H_y = \{|s_y - d_y|, N - |s_y - d_y|\}$, where, when $|s_y - d_y| \le \frac{N}{2}$, if $d_y > s_y$, the path direction of H_y is $\vec{H}_y = +y$, i.e., the positive direction of y-axis; if $d_y < s_y$, the path direction of H_y is $\vec{H}_y = -y$, i.e., the negative direction of x-axis. When $|s_y - d_y| > \frac{N}{2}$, if $d_y > s_y$, the path direction of H_y is $\vec{H}_y = -y$; if $d_y < s_y$, the path direction of H_y is $\vec{H}_y = +y$.

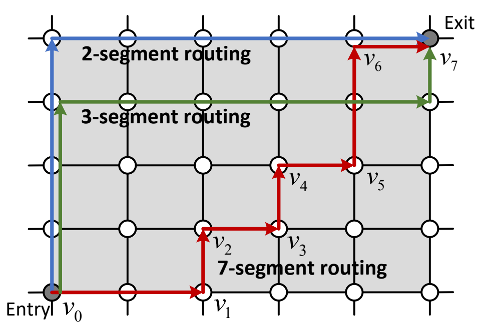


Fig. 2: k-segment routing in LEO satellite network.

B. k-segment Routing

In the MHP region, we present a k-segment routing scheme, which allows flexible and prompt discovery of multiple forwarding paths of low latency, high diversity, and good load balancing. A segment routing path with k segment paths will be referred to as a k-segment routing path. In the grid topology of LEO satellite networks, we define a k-segment routing path as a minimum hop path in the MHP region and consisting of k linear segments. Each linear segment must be vertical to each other in a different direction. As shown in Fig. 2, we give examples of constructing 2-segment routing paths, 3-segment routing paths and 7-segment routing paths. According to the above definition, for a MHP region with minimum hop H, the maximum segment number of k-segment routing paths is H. As the size of the LEO satellite network becomes larger, if all k-segment routing paths are to be enumerated, the storage and calculation overhead will become very large. Therefore, in the context of k-segment routing, we specifically focus on considering only l-segment $(1 \le l \le k)$ routing paths as candidate paths between the entry and exit satellites.

C. Minimum-hop Binary Tree

In order to search for k-segment routing paths conveniently and promptly in the MHP region, we construct a MHBT to characterize the connectivity between LEO satellites. In particular, as shown in Fig. 3, there exists a unique root node that represents the source satellite, along with several leaf nodes that represent the destination satellites. The leaf node number is equivalent to the number of paths connecting the source to the destination. In MHBT, each node's left branch corresponds to movement along the x-axis, indicating that the left child-node can be reached after a single hop along the x-axis. Similarly, the right branch indicates that the right child-node can be reached after a single hop along the y-axis. Moreover, the level H_T of destination node can be obtained by $H_T = H_{MH} + 1$.

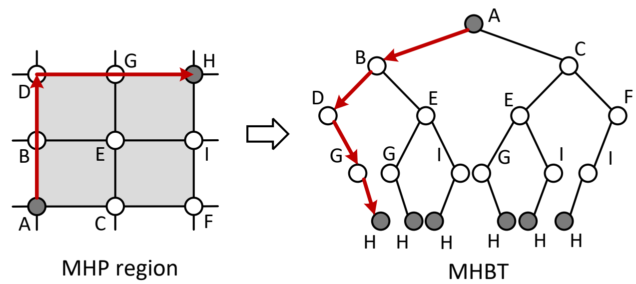


Fig. 3: Minimum-hop binary tree.

We propose an MHBT-based k-segment Routing algorithm, the details of which are summarized in Algorithm 1. The MHBT is constructed level by level, including the H_T levels in total. In the given scenario, the source node s is positioned at the top of the tree as the root, while the destination node d is located at the bottom as a leaf. Due to the nature of minimum-hop paths having an equal hop count, the level of the destination node remains consistent across all possible paths. Each intermediate node has extensions of left and right subtrees, which respectively correspond to the x and y directions in the grid topology. We first find the nodes set VH_h at the hth level. $V_{h,i}$ is the ith node of the hth level and $v_{h,i}$ is the coordinate of node $V_{h,i}$ in MHBT. Then, we record l-segment $(1 \le l \le k)$ routing paths $P^k_{s,d}$ from the source node to each intermediate node, where k represents the maximum number of linear segments the path contains. By executing the algorithm once, we can find l-segment $(1 \le l \le k)$ routing paths from the source node to each node in the MHP region. Finally, by filtering these enumerated paths, we can select multiple paths with low latency, high diversity and good load balancing for each node pair. Specifically, the minimum-hop path can guarantee the low-latency property. Diversity requirements can be achieved by choosing as many paths as disjoint as possible. The cumulative usage of that edge contained in the selected path is factored into the path cost, ensuring good load balancing.

Algorithm 1: MHBT-based k-segment Routing.

Input: M, N, H_{MH}, H_x, H_y, $\vec{H}_x$, $\vec{H}_y$, s, d, k.
Output: l-segment $(1 \le l \le k)$ routing paths $P^k_{s,d}$.

Initialization: $MHBT = \emptyset$, $V_{1,1} = \{1, s, H_x, H_y, \emptyset\}$, $VH_1 \leftarrow V_{1,1}$, $MHBT \leftarrow VH_1$, $H_T = H_{MH} + 1$.
for $h = 2$ to H_T **do**
  $VH_h = \emptyset$, $j = 0$
  for each $V_{h-1,i} \in VH_{h-1}$ **do**
    if $H_{x,v_{h-1,i}} > 0$ **then**
      $j = j + 1$
      $v_{h,j} = \Big(\big(x_{v_{h-1,i}} + M + \vec{H}_x / |\vec{H}_x|\big) \% M, y_{v_{h-1,i}} \Big)$
      $V_{h,j} = \Big\{ h, v_{h,j}, H_{x,v_{h-1,i}} - 1, H_{y,v_{h-1,i}}, v_{h-1,i} \Big\}$
      $VH_h \leftarrow V_{h,j}$
    end if
    if $H_{y,v_{h-1,i}} > 0$ **then**
      $j = j + 1$
      $v_{h,j} = \Big(x_{v_{h-1,i}}, \big(y_{v_{h-1,i}} + N + \vec{H}_y / |\vec{H}_y|\big) \% N \Big)$
      $V_{h,j} = \Big\{ h, v_{h,j}, H_{x,v_{h-1,i}}, H_{y,v_{h-1,i}} - 1, v_{h-1,i} \Big\}$
      $VH_h \leftarrow V_{h,j}$
    end if
  end for
  for each $V_{h,i} \in VH_h$ **do**
    for each $p \in P_{s,v_{h,Fa}}$ **do**
      $p \leftarrow v_{h,i}$
      if p's segment $\le k$ **then**
        $P_{s,v_{h,i}} \leftarrow p$
      end if
    end for
  end for
  Eliminate the duplicate nodes in VH_h
  $MHBT \leftarrow VH_h$
end for

IV. FLEXIBLE AND DISTRIBUTED TRAFFIC ENGINEERING WITH SUPERVISED LEARNING

A. Customized SL Approach Coupled with GNN Architecture

We propose a customized SL approach coupled with GNN architecture that predicts the optimal multipath traffic distribution for each satellite to perform local rate adaptation distributedly. Graph representation learning techniques and message passing frameworks [7], as provided by GNNs [7], are utilized to perceive topology changes and model message exchange procedures. Fig. 4 shows the model architecture. FlexSATE consists of a encoder that performs node-wise message exchange, and a decoder interprets desired multipath traffic distribution from updated and encoded node features. In our approach, we develop a dedicated encoder designed to model and encode both network topology and traffic information. The encoder takes two inputs into consideration: node features and a node adjacency matrix. The node features represent a series of demands originating from each network node, while the adjacency matrix provides information about the neighbors of each node. The encoder employs a shared feed-forward layer to compute an initial node embedding. Subsequently, each node's embedding is updated through message exchange with its neighboring nodes. The embedding update module, inspired by the transformer model introduced in [8], consists of a stack of H identical attention layers. Once message exchange phases are done, we need to concatenate all node embeddings outputted by the encoder to form a graph embedding h_G, because each node's embedding h_v only includes partial information of the whole network. As shown in Fig. 4, a decoder consists of one readout function R, which is used to decode the corresponding multipath traffic distribution from the graph embedding. For a given network node v, R interprets the graph embedding as multipath traffic split ratio $\sigma_p^{v,d}, d \in D_v, p \in P^{v,d}$, where D_v is the destination node set of node v, $P^{v,d}$ is the preconfigured path set of pair $< v, d >$, satisfying $\sum_{p \in P^{v,d}} \sigma_p^{v,d} = 1$. It is worth noting that in our approach, each node can share and reuse the encoder and decoder model, thereby significantly simplifying the complexity and substantially reducing training and inference time.

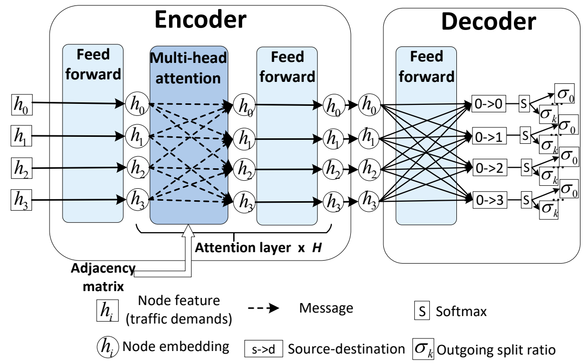


Fig. 4: GNN based supervised learning model of FlexSATE.

B. Offline Training

To learn from an optimal multipath traffic scheduling policy, we address a modified MCF problem to obtain the ground truth optimal path traffic split ratios. Leveraging the multipath routing algorithms proposed earlier, we can readily obtain the optimal traffic distribution in response to varying traffic and connectivity conditions. The modified MCF problem can be formulated as follows: Given a network $G(V, E)$ with a set of traffic demands and preconfigured paths for each satellite pair, our goal is to determine the explicit path traffic split ratios that optimize the traffic flow, aiming to minimize the MLU. Upon solving the aforementioned problem using LP solvers (e.g., Gurobi [5]), the optimal explicit path traffic split ratios for each flow can be obtained. Then, for distributed TE to reach a global optimal performance, the offline centralized learning of FlexSATE can be guided by the global optimal routing solutions accordingly.

We generate a dataset of training samples that includes different input network states and their corresponding target output distributions. Each training sample includes the network-wide traffic matrix and the adjacency matrix as input.

The output comprises the target traffic distribution which is determined by calculating the global optimal path traffic split ratios based on the given inputs. Due to the symmetrical and uniform characteristics of the constellation configuration, each satellite plays an equivalent role in constructing the network topology. That is, the network-wide traffic demand distribution and topology connectivity seen from any satellite at one time may be seen by any other satellite at other times. Inspired by the above findings, we can design and modify the input and output of the neural network from the perspective of satellite observation and decision-making, respectively. Compared with the scheme of outputting the network-wide traffic distribution solutions, we can only predict the distribution of traffic demands originated from the current node based on the satellite homogeneity, which can greatly simplify the design of the neural network model, reduce its size and improve the efficiency of model training. What's more, model distributed deployment and real-time routing decisions are also more practical and feasible. Each satellite generates the traffic matrix and adjacency matrix from its own view. For building new views, the current satellite label is 0 by default, and the other satellite labels are changed to new labels based on the symmetrical and uniform constellation configuration. It is important to mention that we can periodically conduct the process of offline training, thereby updating the FlexSATE model with newly collected samples.

C. Online Agent Plug

Once the offline centralized training is completed, FlexSATE deploys the trained model in each satellite to predict the optimal traffic distribution to perform local routing decision distributedly, so as to achieve near-optimal TE performance with low computation/communication overhead. Each LEO satellite floods traffic demand messages periodically over the entire network and connectivity messages once the link interruption/reestablishment occurs. Each satellite inputs the converted state matrix into FlexSATE to output the predicted optimal traffic distribution. In this way, each LEO satellite performs the local routing decision in a distributed manner.

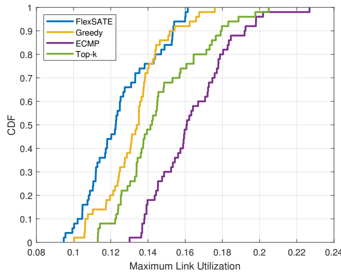
V. NUMERICAL RESULTS AND DISCUSSION

We simulate a LEO satellite network mirroring Starlink shell 2's orbit parameters—a standard Walker-delta LEO constellation with uniform and symmetric setup. At 570 km altitude, 720 satellites are evenly distributed on 36 orbits at a 70-degree orbit inclination. Random link failures are introduced at specific times to simulate dynamic topology changes. For each TM generation, we randomly select satellite pairs to generate traffic demand between 10 Mbps and 400 Mbps. In addition, we set the capacity of inter-satellite laser link to 11 Gbps. The encoder of the prediction model is configured with an embedding dimension of 128 and attention heads of 4. The feed-forward sub-layer in each attention layer has a dimension of 256, and the model consists of 4 attention layers H. The decoder's readout function R is designed with an output dimension equal to $|V| * |P|$. During training, we set the initial learning rate to 10^{-4} and the batch size to 64.

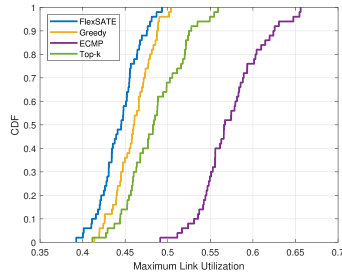
To demonstrate the advantages of FlexSATE, we consider to compare the following schemes: (1) Top-K [2]: This heuristic method selects K flows with the highest demand volume from a given TM and reroutes these flows by solving the modified MCF problem, which is based on the assumption that elephant flows would have a dominant impact on network performance. In the experiment, we let the value of K be one quarter of the demand number. (2) ECMP: evenly distributes traffic among the available shortest paths. (3) Greedy: The shortest paths with sufficient capacity are greedily assigned to each flow in order of arrival and the path traffic split ratios of each flow are solved by the MCF problem sequentially.

In order to simulate different traffic distributions and patterns, we generate different flow numbers in each experiment. After extensive experiments, we present the Cumulative Distribution Function (CDF) of MLU in the satellite network, as shown in Fig. 5. FlexSATE performs optimally because its ability to flexibly adapt to different traffic scenarios. Since the Greedy scheme greedily utilizes network resources, it can only achieve a local optimal solution and may also be affected by the arrival order of flows. For the Top-k scheme, it is affected by the selected flow number K. Increasing the K value will bring better load balancing performance, but at the expense of higher computing overhead. Note that if the K value is equal to the total number of flows, it is the KMCF scheme. Unsurprisingly, ECMP performs worst because it implements a static routing policy and is not aware of load changes. In particular, when 900 flows are injected into the network, ECMP will cause severe network congestion. Then, we compare the performance of average end-to-end delay. In Fig. 6, we can see that with the growth of flow number, the average end-to-end delay increases. As the number of flows increases, unbalanced traffic load distribution may lead to network congestion, resulting in a significant increase in average end-to-end delay. Fig. 6 shows that the average end-to-end delays of FlexSATE and Greedy increase more slowly than ECMP and Top-k, delays of which are affected by network congestion, with a significant increase in the presence of 900 and 1100 flows, respectively. It means that ECMP and Top-k cannot adapt well to dynamic traffic changes, especially in large-scale scenarios with a large amount of flows.

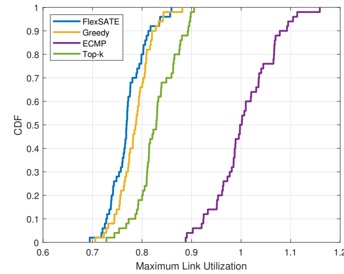
We use the same 4-core Intel 2.4GHz CPU to measure all the computation overheads, with Gurobi utilized as an LP solver to calculate the optimal path traffic split ratios. As shown in Fig. 7, we give the measured computation overheads for comparison. Since ECMP is a static routing solution, we do not evaluate the computation overhead for it. With an inference time of only tens of milliseconds, FlexSATE's GNN architecture enables it to promptly predict the optimal traffic distribution in less than one second. This is reasonable since we divide the large global routing problem into node-wise small local routing subproblems that are solved simultaneously in FlexSATE, thereby enabling better scalability and flexibility. In contrast, centralized TE schemes such as KMCF, Greedy,



(a) The MLU under 100 flows.



(b) The MLU under 500 flows.



(c) The MLU under 900 flows.

Fig. 5: Network performance comparison under different flow numbers.

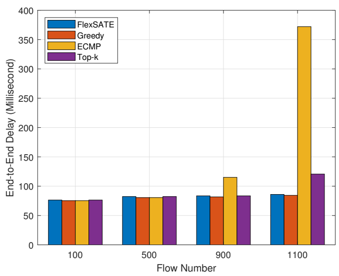


Fig. 6: Average end-to-end delay.

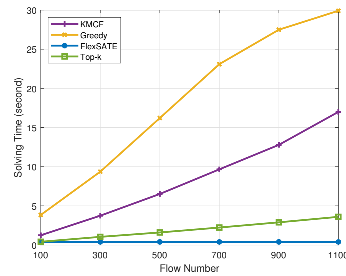


Fig. 7: Computation overheads.

and Top-k take several minutes to solve global optimal routing due to the large problem scale. By considering a global view, KMCF can achieve near-optimal MLU performance, however, it faces limitations in accommodating traffic fluctuations in time as the network size increases. In Greedy, we consider the total time to give the path traffic split ratio of each flow in the order of arrival as the solving time, which is smaller than KMCF but still much larger than our FlexSATE. The solving time of Top-k mainly depends on the number of flows selected for LP problem solving. That is, the fewer flows selected, the shorter the solving time, but the performance is poorer. When more flows are selected, better network performance will be at the expense of large computation overhead. Overall, our FlexSATE can flexibly accommodate network states changes in a responsive manner in large-scale LEO satellite networks.

VI. Conclusion

In this paper, we have proposed a distributed TE system called FlexSATE for UD-LEO satellite networks. We have developed a centralized MHBT-based k-segment Routing algorithm, which is capable of promptly discovering routing paths with low latency, high diversity, and good load balancing from a global view. Furthermore, we have integrated SL into rate adaptation, allowing FlexSATE to learn the global optimal routing strategy in a centralized offline manner and employ distributed deployment for real-time prediction of the optimal traffic distribution. Simulation results have shown that FlexSATE outperforms existing approaches with superior robustness and flexibility.

Acknowledgment

This work was supported by the National Key R&D Program of China under Grant 2020YFB1806104.

References

[1] X. Liu, T. Ma, Z. Tang, X. Qin, H. Zhou, and X. Shen, "Ultrastar: A lightweight simulator of ultra-dense leo satellite constellation networking for 6G," *IEEE/CAA Journal of Automatica Sinica*, vol. 10, no. 3, pp. 632–645, 2023.

[2] M. Ye, J. Zhang, Z. Guo, and H. J. Chao, "Flexdate: Flexible and disturbance-aware traffic engineering with reinforcement learning in software-defined networks," *IEEE/ACM Transactions on Networking*, 2022.

[3] X. Liu, T. Ma, X. Qin, H. Zhou, and L. Zhao, "A DRL empowered multipath cooperative routing for ultra-dense LEO satellite networks," in *GLOBECOM 2023-2023 IEEE Global Communications Conference*. IEEE, 2023, pp. 5961–5966.

[4] Z. Xu, J. Tang, J. Meng, W. Zhang, Y. Wang, C. H. Liu, and D. Yang, "Experience-driven networking: A deep reinforcement learning based approach," in *IEEE INFOCOM 2018-IEEE conference on computer communications*. IEEE, 2018, pp. 1871–1879.

[5] M. Ye, Y. Hu, J. Zhang, Z. Guo, and H. J. Chao, "Mitigating routing update overhead for traffic engineering by combining destination-based routing with reinforcement learning," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 9, pp. 2662–2677, 2022.

[6] Y. Huang, X. Jiang, S. Chen, F. Yang, and J. Yang, "Pheromone incentivized intelligent multipath traffic scheduling approach for LEO satellite networks," *IEEE Transactions on Wireless Communications*, 2022.

[7] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," *Advances in neural information processing systems*, vol. 30, 2017.

[8] W. Kool, H. Van Hoof, and M. Welling, "Attention, learn to solve routing problems!" *arXiv preprint arXiv:1803.08475*, 2018.

[9] N. Geng, M. Xu, Y. Yang, C. Liu, J. Yang, Q. Li, and S. Zhang, "Distributed and adaptive traffic engineering with deep reinforcement learning," in *2021 IEEE/ACM 29th International Symposium on Quality of Service (IWQOS)*. IEEE, 2021, pp. 1–10.