In-Orbit Aggregator-Empowered Federated Learning Framework for Satellite and Terrestrial-Integrated Networks

Hongrok Choi
School of Electrical Engineering
Korea University
Seoul, Korea
ghdfhrooo@korea.ac.kr

Hochan Lee
School of Electrical Engineering
Korea University
Seoul, Korea
ghcks1000@korea.ac.kr

Dongkyun Ryoo
School of Electrical Engineering
Korea University
Seoul, Korea
rdk0610@korea.ac.kr

Heewon Kim
School of Electrical Engineering
Korea University
Seoul, Korea
harry0475@korea.ac.kr

Haneul Ko
Department of Electronic Engineering
Kyung Hee University
Yongin, Korea
heko@khu.ac.kr

Sangheon Pack
School of Electrical Engineering
Korea University
Seoul, Korea
shpack@korea.ac.kr

***Abstract*—In this paper, we introduce a federated learning (FL) framework tailored for a satellite and terrestrial-integrated network (STIN), which employs a semi-asynchronous FL algorithm and in-orbit aggregations (IOA) to mitigate the straggler issue and enhance energy efficiency. Our goal is to optimize IOA-aware routing to enable energy-efficient model aggregation with uncertain ground stations (GSs) in terms of upload-ready timing. To this end, we utilize a time-expanded directed graph (TEDG) to effectively account for the network's connectivity and energy demands. Furthermore, we propose a predictive algorithm to cope with the uncertainty of GSs. A preliminary result demonstrates the robustness of our approach even under inaccurate predictions, achieving a marginal gap of 2% of the cost compared to the optimal scheme.**

***Index Terms*—satellite networks, federated learning, time-expanded graph, algorithm with prediction**

I. Introduction

Recently, the role of low Earth orbit (LEO) satellites has gained attention in various scenarios such as 6G and the Internet of Things (IoT) [1]. In particular, the wide coverage and computing resources provided by LEO satellites have spurred interest in research related to task offloading for IoT in remote areas [2], [3], deployment and routing of virtual network functions (VNFs) in satellite networks [4], [5], and high-speed packet processing based on programmable switches-deployed LEO satellites [6].

Meanwhile, machine learning (ML)-based data processing and network management techniques have gained attention to manage the complexity of large-scale networks integrated with satellites. However, the approach of transferring vast amounts of data to a central server, such as a cloud, for learning is fraught with constraints due to bandwidth limitations of the communication link and intermittent connections caused by high mobility. As a result, federated learning (FL) techniques, which allow for the distributed training of models and the exchange of trained models instead of raw data, thereby reducing the costs of satellite networks associated with transferring large amounts of data, have gained significant attention. Chen *et al.* [7] and Zhao *et al.* [8] have outlined the potential roles that LEO satellites able to process on board can play in an FL framework. These roles include serving as a relay node, a learning agent, and a local and global parameter aggregator.

In contrast to the usual network environments, synchronous FL procedures could worsen the straggler problem in large-scale networks such as satellite networks. To tackle this issue, Razmi *et al.* [10] proposed synchronous and asynchronous FL algorithms for the satellite network where ISL can be used and aggregation at intermediate satellites is feasible. They also demonstrated the potential when PS is possible on satellites. However, they did not actively consider inter-orbit communication and the more complicated network routing that arises from it. Wang *et al.* [11] introduced an FL-aware routing and resource allocation scheme to minimize the delay in transmitting the FL model to the parameter server (PS) on the ground. However, they did not address the energy consumption issue inherent in asynchronous FL algorithms. Lin *et al.* [12] proposed a dynamic FL model aggregation technique that considers periodic and buffer-based aggregation methods, highlighting the potential of a semi-asynchronous approach that can alleviate straggler and energy issues. Nevertheless, they only considered model exchanges of satellites through direct communication with GS without considering inter-satellite links (ISLs).

In this paper, we explore a framework of FL tailored for

This research was supported by National Research Foundation (NRF) of Korea Grant funded by the Korean Government (MSIT) (No. 2020R1A2C3006786).

a satellite and terrestrial-integrated network (STIN) adopting a semi-asynchronous FL algorithm, which aggregates trained models periodically. Within this framework, geo-distributed GSs train and upload local models asynchronously. Concurrently, LEO satellites operate as a backhaul network, performing in-orbit aggregations (IOAs). These aggregations encompass both global model aggregation and partial model aggregation. The former is executed by a satellite operating as a global model aggregator (GA) periodically, while the latter is carried out by intermediate on-path satellites. Meanwhile, the asynchrony of GSs presents a challenge for model upload routing due to the uncertain timing of their readiness to upload models.

To facilitate energy-efficient FL in the presence of this uncertainty, we solve a problem optimizing IOA-empowered routing for trained model upload by leveraging prediction. To this end, we first construct a time-expanded directed graph (TEDG) of the network over the time horizons for each interval. In the constructed TEDG, we find a directed rooted tree in which the vertex at the end of each interval, representing the GA, is designated as the root, and it includes vertices that signify the upload-ready GSs. Lastly, to address the challenge originating from the uncertainty of the GSs, we propose an algorithm that considers predictions on upload-ready GSs identifying which vertices in the TEDG need to be included in the tree.

Through a preliminary result, we demonstrate that our proposed algorithm operates effectively even in situations where the predictions are not accurate.

II. System Model

Fig. 1 shows the STIN-based FL framework considered in this paper consisting of a set of GSs G and a set of LEO satellites L. The overall training process, in alignment with the semi-asynchronous FL technique that follows periodical aggregation, is divided into multiple intervals, with each interval subdivided into T time slots. At the beginning of each interval, a LEO satellite, which acts as a GA (i.e., PS), transmits a global model to GSs (i.e., FL clients). Once the GSs download the global model, they utilize their individual datasets to update it. Upon completion of the update, these GSs transition into an upload-ready state. It is worth noting that the GSs are distributed across a broad area and possess diverse computing capabilities. As a result, the timing of their readiness for uploading, which is followed by the completion of model updates, can vary. Some GSs may complete their updates earlier, while others may not finish within a single interval. Following this, GSs in the upload-ready state asynchronously send their models to the GA. During this phase, the ISLs of LEO satellites facilitate multi-hop transmissions, and satellites overlapping in the transmission routes conduct partial aggregations. Note that the partial aggregation not only decentralizes the computational and energy demands from the GA to on-path satellites but also consolidates multiple incoming updated models into a singular-sized model. By the end of the interval, the GA aggregates the received models

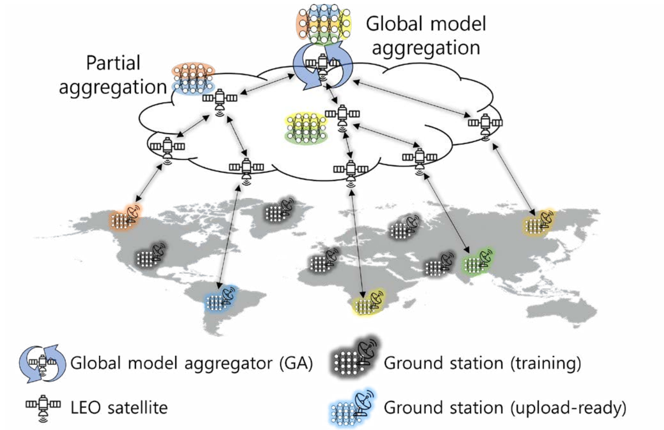

Fig. 1. System model.

and sends the newly aggregated global model back to the GSs that uploaded. For the sake of simplicity, we assume that the model upload and download processes operate independently and mainly focus on the upload phase.

III. Problem Formulation

In this section, we first construct a TEDG for the STIN to capture the asynchronous behavior of upload-ready GSs and energy consumption patterns of the network. Then, we present a formulation for constructing a directed tree within the TEDG.

A. Time-Expanded Directed Graph Construction

To effectively model the dynamics and energy-consuming patterns of STIN, we construct a TEDG for the STIN. Each interval of the semi-asynchronous FL procedure is defined by durations T. The TEDG for every interval, denoted as $\mathcal{G} = (\mathcal{V}, \mathcal{A}, \mathcal{M}, \mathcal{N})$, is constituted by sets of vertices $\mathcal{V}$, arcs $\mathcal{A}$, vertex features M and arc features $\mathcal{N}$.

Specifically, the vertex set $\mathcal{V} = \mathcal{V}_G \cup \mathcal{V}_L$ is formed by time-expanded vertices of GSs and LEO satellites from STIN, which is replicated along the time horizon, represented by $\mathcal{V}_G = \{v_{g,t} | 1 \leq t \leq T, g \in G\}$ and $\mathcal{V}_L = \{v_{l,t} | 1 \leq t \leq T, l \in L\}$, respectively.

The arc set $\mathcal{A} = \mathcal{A}_G \cup \mathcal{A}_{GL} \cup \mathcal{A}_L \cup \mathcal{A}_{LL}$ consists of storage and transmission arcs linking vertices across consecutive slots. The storage arcs of GSs $\mathcal{A}_G = \{(v_{g,t}, v_{g,t+1}) | 1 \leq t < T, g \in G\}$ and LEO satellites $\mathcal{A}_L = \{(v_{l,t}, v_{l,t+1}) | 1 \leq t < T, l \in L\}$ both represent the connections between vertices in consecutive time slots pointing to the identical GS and LEO, respectively, which implies the utilization of their respective storage capabilities. The transmission arcs for vertices of GSs and LEO satellites $\mathcal{A}_{GL} = \{(v_{g,t}, v_{l,t+1}) | 1 \leq t < T, g \in G, l \in L\}$ and LEO satellites $\mathcal{A}_{LL} = \{(v_{l,t}, v_{l',t+1}) | 1 \leq t < T, l, l' \in L\}$ indicate the transmission from GSs to LEO satellites and among LEO satellites, respectively. For simplicity, we assume that both GSs and satellites transmit a single data unit in each time slot, equivalent to the size of the model. Meanwhile, owing to the partial aggregation, any intermediate satellites that receive multiple models will transmit an aggregated model. Reflecting these considerations, we set the capacities for both transmission and storage arcs to one.

The vertex feature set $\mathcal{M} = \mathcal{M}_G \cup \mathcal{M}_L$ captures the features associated with each type of vertex. The vertex feature indicates an aggregation cost in terms of energy consumption, which is proportional to the number of models received, where $\mathcal{M}_G = \{m_{g,t} | 1 \leq t \leq T, g \in \mathcal{V}_G\}$ represents the aggregation cost of vertices for GSs and $\mathcal{M}_L = \{m_{l,t} | 1 \leq t \leq T, l \in \mathcal{V}_L\}$ represents the aggregation cost of vertices for LEO satellites.

The arc feature set $\mathcal{N} = \mathcal{N}_G \cup \mathcal{N}_{GL} \cup \mathcal{N}_L \cup \mathcal{N}_{LL}$ captures the features associated with each type of arc. The arc feature sets $\mathcal{N}_{GL} = \{n_{g,l,t} | (v_{g,t}, v_{l,t+1}) \in \mathcal{A}_{GL}\}$ and $\mathcal{N}_{LL} = \{n_{l,l',t} | (v_{l,t}, v_{l',t+1}) \in \mathcal{A}_{LL}\}$ represent a transmission cost in terms of energy consumption. $\mathcal{N}_G = \{n_{g,g,t} | (v_{g,t}, v_{g,t+1}) \in \mathcal{A}_G\}$ and $\mathcal{N}_L = \{n_{l,l,t} | (v_{l,t}, v_{l,t+1}) \in \mathcal{A}_L\}$ represent a storing cost in terms of the storage capacity.

B. Tree Construction Problem Formulation

In the context of TEDG, our goal is to construct a directed tree rooted at vertex r, which represents GA. In doing so, we aim to minimize both aggregation and transmission costs while maximizing the number of upload-ready GSs included in the tree. To this end, let $x_a \in \mathbf{x} = \{x_1, x_2, ..., x_{|\mathcal{A}|}\}$ be a binary variable that is 1 if arc $a \in \mathcal{A}$ is in the tree, 0 otherwise. Similarly, let $y_v \in \mathbf{y} = \{y_1, y_2, ..., y_{|\mathcal{V}|}\}$ be a binary variable that is 1 if vertex $v \in \mathcal{V}$ is in the tree, 0 otherwise. To begin, The transmission cost $c^{\text{tx}}(\mathbf{x})$ in the tree can be defined as

$$c^{\text{tx}}(\mathbf{x}) = \sum_{a \in \mathcal{A}_{GL} \cup \mathcal{A}_{LL}} n_a \cdot x_a, \tag{1}$$

where $n_a \in \mathcal{N}$. Subsequently, the aggregation cost $c^{\text{agg}}(\mathbf{y})$ in the tree can be defined proportionally to the number of in-degree vertices of the selected vertices $y_v \in \mathbf{y}$ and the cost for aggregating models, which can be derived as

$$c^{\text{agg}}(\mathbf{y}) = \sum_{v \in \mathcal{V}_L} m_v \cdot y_v \cdot (|d(v)| - 1), \tag{2}$$

where $m_v \in \mathcal{M}_L$, $d(v)$ is the set of in-degree vertices of vertex v, and $|\cdot|$ return the size of given set. Lastly, the number of upload-ready GSs included in the tree can be defined as

$$c^{\text{GS}}(\mathbf{y}) = \sum_{u \in \mathcal{U}} y_u, \tag{3}$$

where $\mathcal{U} \subseteq \mathcal{V}_{\mathcal{G}}$ denotes the set of GSs becoming upload-ready in the current interval.

Therefore, the objective function can be defined as follows

$$\min_{\mathbf{x},\mathbf{y}} \quad c^{\text{tx}}(\mathbf{x}) + c^{\text{agg}}(\mathbf{y}) + c^{\text{GS}}(\mathbf{y}), \tag{4}$$

subject to

$$r \in \{v_{l,T} | l \in L\}, \tag{5}$$

$$y_r = 1, \tag{6}$$

$$x_a \in \{0, 1\}, \forall a \in \mathcal{A}, \tag{7}$$

$$y_v \in \{0, 1\}, \forall v \in \mathcal{V}, \tag{8}$$

$$x_a \leq y_v, x_a \leq y_{v'}, \forall a = (v, v') \in \mathcal{A}. \tag{9}$$

The constraints (5) and (6) denote that the vertex r belongs to the set of vertices corresponding to the LEO satellites in the last interval in the TEDG and should be included in the tree. The constraints (7) and (8) confirm that the decision variables x_a and y_v are binary. The constraint (9) represents the fundamental condition for tree construction, capturing the relationship between the decision variables x_a and y_v.

Meanwhile, constructing the optimal directed tree is NP-hard [13]. Additionally, due to the large scale of STIN and its heterogeneous computing capabilities and resource constraints, each model of GS is updated and uploaded asynchronously. Consequently, it is challenging to specify the set of GSs $\mathcal{U}$ that are ready for upload and their precise timing and corresponding vertices in the TEDG in advance. Furthermore, constructing the optimal directed tree is NP-hard [13], and under these uncertainties, it becomes a more intricate problem. Lastly, the vertex indicating the upload-ready GS can only be discerned at the time of readiness, necessitating online problem-solving approaches.

IV. Predicted Upload-Ready-based Algorithm

To address the aforementioned challenges, we propose an algorithm leveraging prediction results on which and when GS will become upload-ready to construct the tree, which is inspired by the algorithm outlined by [14]. The detailed steps of the algorithm are depicted in Algorithm 1.

Algorithm 1 Predicted Upload-Ready GS-based Algorithm

Input: TEDG $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{M}, \mathcal{N})$ and GA vertex r
Output: Predicted upload-ready GS-based tree $\bar{\mathcal{T}}$

1: Initialize tree $\bar{\mathcal{T}}$
2: Predict to obtain $\hat{\mathcal{U}}$
3: Construct tree $\hat{\mathcal{T}}$ based on $\hat{\mathcal{U}}$
4: **while** a vertex u appears **do**
5:     Add the vertex u into $\mathcal{U}$
6:     **if** $u \in \hat{\mathcal{U}}$ **then**
7:         Retrieve the path $p_{u,r}^{\hat{\mathcal{T}}}$ from $\hat{\mathcal{T}}$
8:         Add the path $p_{u,r}^{\hat{\mathcal{T}}}$ into $\bar{\mathcal{T}}$
9:     **else**
10:         Find the shortest path $p_{u,v'}^{*}$ from v to $v' \in \mathcal{V}_{\hat{\mathcal{T}}}$
11:         Retrieve the path $p_{v',r}^{\hat{\mathcal{T}}}$ from $\hat{\mathcal{T}}$
12:         Add path $p_{u,v'}^{*} \cup p_{v',r}^{\hat{\mathcal{T}}}$ into $\bar{\mathcal{T}}$
13:     **end if**
14: **end while**

At the beginning of each interval, the algorithm initializes the prediction-based tree, denoted as $\bar{\mathcal{T}} = (\mathcal{V}_{\bar{\mathcal{T}}}, \mathcal{A}_{\bar{\mathcal{T}}}, r)$. Here, both $\mathcal{V}_{\bar{\mathcal{T}}}$ and $\mathcal{A}_{\bar{\mathcal{T}}}$ are set to $\emptyset$, and r represents the root of the tree (see line 1 in Algorithm 1). Then, the algorithm predicts a set of vertices $\hat{\mathcal{U}}$ of TEDG, representing both the GSs anticipated to be ready for upload and their respective timings (see line 2 in Algorithm 1). Subsequently, the algorithm constructs a preliminary tree $\hat{\mathcal{T}} = (\mathcal{V}_{\hat{\mathcal{T}}}, \mathcal{A}_{\hat{\mathcal{T}}}, r)$ that minimizing (4), assembled from the predicted vertex set $\hat{\mathcal{U}}$ to GA r (see line 3 in Algorithm 1). As time advances, a vertex

$u \in \mathcal{U}_G$ appears, signifying a GS in STIN that just completed local model training and is now upload-ready (see line 4 in Algorithm 1). If the prediction is accurate (i.e., the vertex is in the predicted set $\hat{\mathcal{U}}$), the path $p_{u,r}^{\hat{\mathcal{T}}} = \{(v^{(1)}, v^{(2)}, ..., v^{(k)}) | v^{(1)} = u, v^{(k)} = r, \forall i \in \{1, 2, ..., k-1\}, (v^{(i)}, v^{(i+1)}) \in \mathcal{A}_{\hat{\mathcal{T}}}\}$ connecting u with r is retrieved from the preliminary tree $\hat{\mathcal{T}}$ (see lines 6-7 in Algorithm 1). Then, the vertices and arcs that constitute the path are added to $\bar{\mathcal{T}}$ (see line 8 in Algorithm 1). Otherwise, the algorithm finds the shortest path $p_{u,v'}^{*} = \{(v^{(1)}, v^{(2)}, ..., v^{(k)}) | v^{(1)} = u, v^{(k)} = v', \forall i \in \{1, 2, ..., k-1\}, (v^{(i)}, v^{(i+1)}) \in \mathcal{A}_{GL} \cup \mathcal{A}_{LL}\}$ with the Dijkstra algorithm where $v' \in \mathcal{V}_{\hat{\mathcal{T}}}$ denotes a vertex consisting of the tree $\hat{\mathcal{T}}$ and retrieves a path $p_{v',r}^{\hat{\mathcal{T}}}$ connecting v' with r from the tree $\hat{\mathcal{T}}$ (see lines 10-11 in Algorithm 1). Then, the algorithm adds the vertices and arcs of the merged path $p_{u,v'}^{*} \cup p_{v',r}^{\hat{\mathcal{T}}}$ into $\bar{\mathcal{T}}$ (see line 12 in Algorithm 1).

V. Evaluation Result

For performance evaluation, we compare our algorithm PREDICTIVE with the following schemes: 1) OPTIMAL where the tree is constructed with paths having minimum cost based on the precise predictions; and 2) GREEDY where upon identifying an upload-ready GS vertex over time the tree is constructed with paths having the least cost to the GA without considering vertices that will be determined in the future. The performance is evaluated by the cost of the tree constructed by each scheme. We consider a specific region defined by certain latitudes and longitudes. In this region, we assume a STIN formed by a 5-by-5 grid network of the satellite constellation capturing 5 orbital planes with 5 satellites in each plane and the GSs corresponding to each satellite's coverage.

We set the length of each interval $T = 6$ and both transmission cost $n_a \in \mathcal{N}$ and aggregation cost $m_v \in \mathcal{M}$ are set to one unit cost. The accuracy of the prediction is defined as the ratio of the predicted vertices that match with the actual appearing vertices. For simplicity, it is assumed that predictions are made through trained models that guarantee each accuracy level.

Fig. 2 presents the costs of constructed trees, which are normalized by the cost of OPTIMAL scheme, depending on the accuracy of the prediction. Across all the cases of accuracy, PREDICTIVE shows a marginal performance gap with OPTIMAL at most around 2% and this gap tends to decrease as the accuracy increases. This observation implies that our scheme can proficiently construct trees even where the prediction model is incomplete or the prediction is challenging. In contrast, GREEDY shows higher costs than our proposed scheme. GREEDY incurs 1.12 to 1.14 times higher cost than our proposed scheme and 1.17 times higher cost than OPTIMAL. This is because it constructs trees without considering GSs that might be ready for upload in the future. By greedily forming trees, it excludes potential opportunities for model aggregations, which could lead to reduced data transmission and, consequently, lower energy consumption.

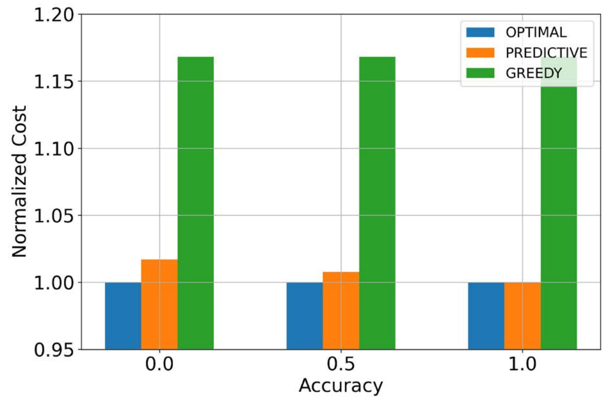

Fig. 2. Normalized cost.

VI. Conclusion and Future Work

In this paper, we introduced an FL framework for STIN, adopting a semi-asynchronous FL algorithm and leveraging IOA. Our proposed algorithm found energy-efficient routing paths even in situations with uncertain upload-ready timing of GSs. This was supported by our preliminary result, demonstrating the robustness of our approach with performance close to optimal, even under inaccurate predictions. In our future work, we will validate our proposed method in larger satellite network environments.

Acknowledgment

This research was supported by National Research Foundation (NRF) of Korea Grant funded by the Korean Government (MSIT) (No. 2020R1A2C3006786).

References

[1] Z. Shen *et al.*, "A Survey of Next-generation Computing Technologies in Space-air-ground Integrated Networks," *ACM Computing Surveys*, vol. 56, no. 1, pp. 1–40, January 2024.

[2] F. Tang *et al.*, "A Deep Reinforcement Learning-Based Dynamic Traffic Offloading in Space-Air-Ground Integrated Networks (SAGIN)," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 1, pp. 276–289, January 2022.

[3] D. Han *et al.*, "Two-Timescale Learning-Based Task Offloading for Remote IoT in Integrated Satellite–Terrestrial Networks," *IEEE Internet of Things Journal*, vol. 10, no. 12, pp. 10131–10145, June 2023.

[4] D. Bhattacherjee, S. Kassing, M. Licciardello, and A. Singla, "In-orbit Computing: An Outlandish thought Experiment?," in *HotNets 2020*, New York, NY, USA, November 2020.

[5] Z. Jia *et al.*, "Joint Optimization of VNF Deployment and Routing in Software Defined Satellite Networks," in *Proc. VTC-Fall 2018*, Chicago, IL, USA, August 2018.

[6] H. Pan, H. Yao, T. Mai, N. Zhang, and Y. Liu, "Scalable Traffic Control Using Programmable Data Planes in a Space Information Network," *IEEE Network*, vol. 35, no. 4, pp. 35–41, July/August 2021.

[7] H. Chen, M. Xiao, and Z. Pang, "Satellite-Based Computing Networks with Federated Learning," *IEEE Wireless Communications*, vol. 29, no. 1, pp. 78–84, February 2022.

[8] M. Zhao, C. Chen, L. Liu, D. Lan, and S. Wan, "Orbital Collaborative Learning in 6G Space-Air-Ground Integrated Networks," *Neurocomputing*, vol. 497, pp. 94–109, August 2022.

[9] C. Chen, L. Shen, K. Feng, L. Yang, and J. Wu, "Edge Selection and Clustering for Federated Learning in Optical Inter-LEO Satellite Constellation," *arXiv preprint arXiv:230316071*, 2023.

[10] N. Razmi, B. Matthiesen, A. Dekorsy, and P. Popovski, “On-board Federated Learning for Satellite Clusters with Inter-Satellite Links,” *arXiv:2307.08346v1*, July 2023.
[11] P. Wang, H. Li, and B. Chen, “FL-Task-aware Routing and Resource Reservation over Satellite Networks,” in *Proc. GLOBECOM 2022*, Rio de Janeiro, Brazil, December 2022.
[12] J. Lin, J. Xu, Y. Li, and Z. Xu, “Federated Learning with Dynamic Aggregation Based on Connection Density at Satellites and Ground Stations,” in *Proc. 2022 IEEE International Conference on Satellite Computing (Satellite)*, Shenzhen, China, November 2022.
[13] D. Watel and M.-A. Weisser, “A Practical Greedy Approximation for the Directed Steiner Tree Problem,” *Journal of Combinatorial Optimization*, vol. 32, pp. 1327–1370, September 2016.
[14] C. Xu and B. Moseley, “Learning-Augmented Algorithms for Online Steiner Tree,” in *Proc. AAAI 2022*, February/March 2022.