# SemSAN: Semantic Satellite Access Network Slicing for NextG Non-Terrestrial Networks

Chaoqun You[*†], Xingqiu He[*†], Yajing Zhang[‡], Kun Guo[§], Yue Gao[†], Tony Q. S. Quek[*]
[*]Singapore University of Technology and Design
[†]Fudan University
[‡]Shanghai Jiao Tong University
[§]East China Normal University

***Abstract*—Satellites equipped with computing capabilities serve as invaluable access platforms for 5G and beyond (NextG) non-terrestrial networks (NTNs). They facilitate the continuous execution of resource-intensive edge-assisted deep learning (DL) tasks that are offloaded from Internet-of-Things (IoT) user equipment (UEs) in remote areas. To this end, satellite access network (SAN) resources need to be carefully "sliced", considering both the constrained energy availability and the scarcity of SAN resources. Existing SAN slicing approaches tend to treat offloaded tasks conventionally, overlooking the intricate *semantics* associated with DL tasks. In this paper, we propose semantic SAN (SemSAN), the first semantic SAN slicing algorithm for NextG AI-native NTNs. Our keen observations reveal that various DL tasks (i) can tolerate different degrees of image compression, and (ii) may yield equivalent model accuracy when employing DNN models with different sizes. These observations inspire us to further exploit the computation capability of a SAN to support more tasks while still minimizing overall energy consumption. After analyzing the characteristics of this optimization problem, we propose an online greedy SemSAN slicing algorithm to approximate its optimal solution. Extensive experiments verify the effectiveness of SemSAN in energy saving and its ability to support a substantial number of tasks, compared with other baselines.**

***Index Terms*—SAN, semantics, slicing, DL tasks**

## I. Introduction

SANs, which are traditionally used for a limited set of applications, such as TV broadcasting and disaster management, have regained their shine in recent developments of NextG NTNs [1–3]. The rise of emerging applications, such as autonomous vehicles, drone-based delivery, precision agriculture, and numerous yet-to-be-uncovered use cases, has spurred extensive research in the realm of SAN to deliver artificial intelligence (AI) services. Such AI-native SANs in conjunction with the existing terrestrial infrastructures could provide a seamless and ubiquitous global coverage, which is precisely the vision of NextG networks.

While AI has certainly expanded the horizons of SAN applications with diverse use cases, the integration of DL tasks has substantially strained the SAN, particularly in terms of power consumption and resource allocation. Commonly, *network slicing* is a fundamental tool to alleviate this issue. It allows network operators to virtualize and allocate the computation and communication resources of SAN based on their needs, leading to a substantial enhancement in SAN resource utilization. Additionally, akin to the radio access network (RAN) slicing technology for terrestrial infrastructures, SAN slicing is fully supported by the Open RAN (O-RAN) framework [4], which has been envisioned as the future for mobile industry.

Nevertheless, in the AI-native NextG SAN, the scarcity of power and network resources is notably exacerbated. Compared to terrestrial infrastructures, satellites face limitations not only in terms of their number but also in the finite energy supplies on each satellite, which cannot be promptly replenished. This makes it insufficient to rely solely on network slicing to address the substantial computational burden and energy consumption brought by DL tasks to the SAN. To this end, in this paper, we delve into the *semantics* of the DL tasks to further reduce network overhead by compressing the task data. For instance, tasks like classifying cars are semantically less complex than those involving bicycles, allowing for more aggressive image compression if classifying cars is the priority [5]. Furthermore, compressed images often necessitate lightweight DNN models to maintain acceptable model inference accuracy.

The introduction of semantics brings both opportunities and challenges. On the one hand, it makes fuller use of the computing power of SAN by allowing more DL tasks to be processed at the same time. On the other hand, in the context of semantic SAN slicing, the conflict between the number of tasks processed and system energy consumption becomes more pronounced. Solving this confliction, however, is particular challenging due to (i) the dynamic SAN topology leads to a constantly *changing* set of satellites visible to UEs and their tasks, (ii) the *unclear* relationship between the DL task compression degree, the choice of DNN model sizes and the model inference accuracy.

In this paper, we propose Semantic Satellite Access Network (SemSAN), the inaugural *semantic* and *non-terrestrial* slicing approach to support SAN-assisted DL task processing in NextG AI-native NTNs. SemSAN strives to maximize the number of accepted DL tasks while still minimizing the system energy consumption. This endeavor precisely encapsulates the objective of our proposed optimization problem. Meanwhile, the changing set of visible satellites to DL tasks is quantified as placement constraints within the optimization problem. Moreover, the relationships between task compression level

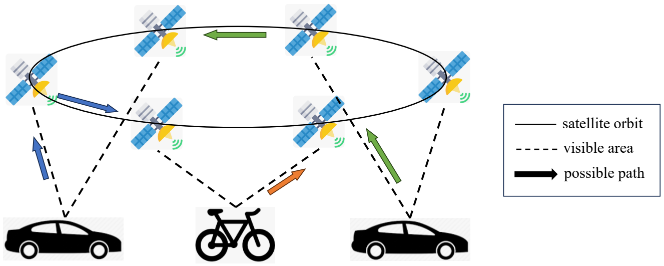

Fig. 1: SemSAN system model

and DNN model size, as well as task compression level and model inference accuracy, are modeled as piecewise functions attainable through data-driven approaches. After analyzing the characteristics of the SemSAN slicing optimization problem, we propose a greedy online SemSAN slicing algorithm, which not only approximates the optimal solutions to the optimization problem, but is also practical for real-world execution.

The contributions are summarized as follows,

- We propose SemSAN, the first semantic and non-terrestrial slicing approach for NextG AI-native SANs.
- We strike a balance between the number of tasks accepted by SemSAN and its energy consumption, ensuring SemSAN not only meets the QoS requirements of UEs in dynamic topologies but also leverages the benefits brought by DL task' semantic compression. (Section IV)
- We propose an online greedy SemSAN slicing algorithm to implement SemSAN in practice. (Section IV)
- We conduct extensive experiments to testify the effective of SemSAN in both energy saving and its ability to support a substantial number of tasks. (Section V)

## II. System Model

### A. Network Model

We consider a SAN shown in Fig 1, which consists of remote UEs and satellites. The SAN is described as a graph $\mathcal{G} = (\mathcal{S}, \mathcal{E})$, where $\mathcal{S} = \{1, \ldots, s, \ldots, S\}$ represents satellites, while $\mathcal{E} = \{(s, s')|s, s' \in \mathcal{S}, s \neq s'\}$ represents the set of transmission links. A task generated from a certain UE will be submitted to SAN $\mathcal{G}$ for processing.

In order to reflect the dynamism of the SAN topology, we consider a time horizon with $T$ time slots, each of which has the same time duration $\tau$. The time slots are indexed by a set $\mathcal{T} = \{1, \ldots, t, \ldots, T\}$. The network topology is assumed to be invariant in each time slot and may change between different time slots. As a result, the network topology $\mathcal{G}$ is a graph composed of $T$ layers, where each layer $\mathcal{G}(t)$ corresponds to a snapshot of the network. In practice, the time slot duration is around 1 minute, while the DL tasks sent to the SAN for inference are often delay sensitive, requiring a maximum delay typically in the range of milliseconds to seconds [6]. This fact implies that, compared to the variation of SAN topology, the procedure of task scheduling and processing is transient. Therefore, in this paper, the SAN topology is considered to be fixed within one task scheduling period.

### B. DL Task Model

The workloads are generated by applications running on UEs. We define an *application class* as a high-level goal that must be achieved through the execution of one or more *DL tasks* with specific requirements. For example, a monitoring application class could require the detection and tracking of person or vehicle objects located in a remote area with a minimum expected accuracy of 0.50 and maximum end-to-end (E2E) delay of 800 ms [5].

Let $\mathcal{A} = \{1, \ldots, a, \ldots, A\}$ be the set of all application classes, and $\mathcal{U} = \{1, \ldots, u, \ldots, U\}$ be the set of remote UEs distributed in an area without cellular coverage. Compared to satellites, the UEs' movements are much slower. Therefore, the relative motion between UEs and satellites mainly relies on the movement of satellites, and the UEs can be regarded as quasi-static. A task $k$ can be uniquely identified at the system level by a tuple $(a, u, \kappa)$, where $\kappa$ denotes the task index generated by a UE. That is, a generic task is defined as $k = (a, u, \kappa) \in \mathcal{K}$, where $\mathcal{K}$ is the set of all generated tasks.

For a given $k \in \mathcal{K}$, we define its compression scaling factor at time slot $t$ as $\sigma_k(t)$, where $0 \leq \sigma_k(t) \leq 1$. Therefore, if $\phi_k$ denotes the number of packets in each task $k$, and $q_k$ denotes the number of bits in each packet, then original the data amount (in bits) of task $k$ is $\phi_k q_k$, whereas the submitted data amount (in bits) of task $k$ with compression is $\sigma_k(t)\phi_k q_k$.

Let $\rho_k$ denote the expected inference accuracy for task $k$. Intuitively, the expected accuracy $\rho_k$ is positive related to the compression factor $\sigma_k(t)$. In this paper, we adopt the data-driven approach used in [5], where $\rho_k$ can be determined through a regression model. This approach takes into account for the explicit dependencies of the accuracy function $\rho_k(\sigma)$ on the compression scaling factor, and assumes that this function is provided as part of the problem input. In our performance evaluation, we will adhere to the function defined in [5], which treats accuracy as a piecewise function applicable only to discrete solution values.

### C. Communication Model

*1) UE-Satellite Data Transmission:* The communication between UEs and satellites is achieved by wireless channels, which is susceptible to effects of carrier frequency, noise, transmission distance and bandwidth capacity. To describe the communication model between UEs and satellites, we first introduce the signal-to-noise ratio (SNR) from UE $u$ to satellite $i$ for task $k$ at time slot $t$ as follows,

$$\gamma^k_{u,s}(t) = \frac{p^k_{u,s}(t) H^k_{u,s}(t)}{b^k_{u,s}(t) I_0}, \tag{1}$$

where $p^k_{u,s}(t)$ denotes the transmit power from UE $u \in \mathcal{U}$ to satellite $s$ at time slot $t$, $H^k_{u,s}(t)$ is the channel gain, $b^k_{u,s}(t)$ is the allocated uplink bandwidth between UE $u$ and satellite $s$, and $I_0$ is the noise power spectral density. Naturally, there exist a transmit power constraint,

$$p_{\text{th}} \leq p^k_{u,s}(t) \leq p_{\max}, \tag{2}$$

where $p_{\text{th}}$ is the threshold to guarantee that data can be transmitted, and $p_{\max}$ is the maximal transmit power of an UE. According to the Shannon-Hartley Theorem, the achievable data rate of uplink transmission from UE $u$ to satellite $s$ at time slot $t$ is computed by,

$$r_{u,s}^k(t) = b_{u,s}^k(t)\log_2(1+\gamma_{u,s}^k(t)). \tag{3}$$

Note that in this paper we assume that during the data transmission process, for the same UE, its communication-related parameters' values keep the same across tasks. That is, $\gamma_{u,s}^k(t) = \gamma_{u,s}(t)$, $r_{u,s}^k(t) = r_{u,s}(t)$.

As a result, the uplink transmission time $D_{u,s}^{\text{tr},k}(t)$ of task $k$ from UE $u$ to satellite $s$ at time slot $t$ is computed by,

$$D_{u,s}^{\text{tr},k}(t) = \frac{\sigma_k(t)\phi_k q_k}{r_{u,s}^k(t)}. \tag{4}$$

Meanwhile, the total energy consumption of uplink transmission is computed by,

$$E_{u,s}^{\text{tr},k}(t) = E_{u,s}^k(t) + E_{u,s}^{\text{RX},k}(t) = (p_{u,s}(t)+p_{\text{RX}})D_{u,s}^{\text{tr},k}(t), \tag{5}$$

where $E_{u,s}^k(t)$ and $E_{u,s}^{\text{RX},k}(t)$ are the energy consumption of UE $u$'s data transmission and satellite $s$'s data receiving, respectively. $p_{\text{RX}}$ is the receiving power of a satellite. $p_{\text{RX}}$ is often a fixed value and is assumed to be unified across satellites in this paper.

*2) Data Forwarding Between Satellites:* The communication between satellites is achieved by inter-satellite links (ISLs). Assume that data traffic starts from $s_{\min}$ to $s_{\max}$, where the former is selected from the visible set of satellites of the UE that generates the data, and the later is selected to process the task. Let $p_{\text{ISL}}$ be the transmit and receive power of satellites when delivering data by ISLs, and $r_{\text{ISL}}$ be the achievable data rate of ISL, which are fixed values. Then the total forwarding delay during uploading procedure can be computed by

$$D_{s_{\min},s_{\max}}^k(t) = \frac{n_k^{\text{hop}}\sigma_k(t)\phi_k q_k}{r_{\text{ISL}}}, \tag{6}$$

where $n_k^{\text{hop}}$ is the number of hops for data transfer from $s_{\min}$ to $s_{\max}$. The energy consumption of the communication relay executed along the path from $s_{\min}$ to $s_{\max}$ is computed by,

$$E_{s_{\min},s_{\max}}^k(t) = 2p_{\text{ISL}}D_{s_{\min},s_{\max}}^k(t). \tag{7}$$

Given that the forwarding delay is severe in SAN, we adopt the shortest path routing method in this paper, which is most likely to approach the optimal method.

### D. Computation Model

The computational complexity of DL models is typically measured using floating point operations per second (FLOPS) [7]. According to the compression level $\sigma_k(t)$ of task $k$, it chooses a suitable DNN model for inference according to a pre-defined piecewise function to achieve the required accuracy. The model inference/computation latency and energy consumption are all determined by this chosen model. Assume there are $\mathcal{N}_a = \{1,\ldots,n_a,\ldots,N_a\}$ models available for application class $a$ to choose, each of which requires $f_{n_a}$ FLOPS of computation and $D_{n_a}$ s for model reference.

$$E_{u,s}^{cmp,k}(t) = f_{n_a}D_{n_a} \tag{8}$$

## III. Problem Formulation

In order to formulate the optimization problem, we first introduce a binary matrix $\mathbf{x} = \{x_s^k(t)|k \in \mathcal{K}, s \in \mathcal{S}, t \in \mathcal{T}\}$ to describe the *task association strategy*. $x_s^k(t) = 1$ indicates that task $k$ accesses node $s$ for data processing, and $x_s^k(t) = 0$ otherwise. Specifically, at time slot $t$, we use $x_{s_{\min}}^k(t)$ to indicate whether task $k$ chooses satellite node $s_{\min}$ to start its data transmission, while $x_{s_{\max}}^k(t)$ to indicate whether task $k$ chooses satellite node $s_{\max}$ for model inference.

### A. Fundamental Constraints

There are six types of fundamental constraints in the SemSAN slicing problem, one of which is the placement constraint, three of which are resource capacity constraints, two of which are UE QoS guarantee constraints. The three resource capacity constraints are communication, computation and energy capacity constraints. And the two QoS guarantee constraints are delay constraints and inference accuracy constraints. In the following we will specifically introduce these four constraints.

*1) Placement Constraints:* There are two types of placement constraints. The first is applied to $s_{\min}$, that the dynamic satellite topology naturally constrains the satellite clusters that UEs can potentially directly arrive. Let $\mathcal{S}_u^{\text{IN}}(t)$ be the set of satellites that are visible to UE $u$ at time slot $t$, then the placement constraint on $x_{s_{\min}}^k(t)$ is $\sum_{t\in\mathcal{T}}\sum_{s_{\min}\in\mathcal{S}_u^{\text{IN}}(t)} x_{s_{\min}}^k(t) \in \{0,1\}$.

The second placement constraint is applied to $s_{\max}$, that there may not be the required DNN models on certain satellites. Let $\mathcal{S}_a^{\text{MOD}}$ be the set of satellites that possess the DNN models for application class $a$, then the placement constraint on $x_{s_{\max}}^k(t)$ is $\sum_{t\in\mathcal{T}}\sum_{s_{\max}\in\mathcal{S}_a^{\text{MOD}}} x_{s_{\max}}^k(t) \in \{0,1\}$.

These two placement constraints are interrelated, that for any task $k \in \mathcal{K}$, once it is assigned to access the SAN via a particular satellite $s_{\min}$ (i.e., $x_{s_{\min}}^k(t) = 1$), it must simultaneously selects one of the satellites $s_{\max}$ for model inference (i.e., $x_{s_{\max}}^k(t) = 1$). Therefore, we further combine the two placement constraints as follows,

$$\sum_{t\in\mathcal{T}}\left(\sum_{s_{\min}\in\mathcal{S}_u^{\text{IN}}(t)} x_{s_{\min}}^k(t)\right)\left(\sum_{s_{\max}\in\mathcal{S}_a^{\text{MOD}}} x_{s_{\max}}^k(t)\right) \in \{0,1\}. \tag{9}$$

*2) Resource Capacity Constraints:* As for the communication capacity, we mainly consider the G2S links. Let $B_{\text{G2S}}$ be the bandwidth available for the uplink data transmission from UEs to satellites, then bandwidth constraints are formalized as follows,

$$\sum_{k\in\mathcal{K}}\sum_{s_{\min}\in\mathcal{S}} x_{k,s_{\min}}(t)b_{k,s_{\min}}(t) \le B_{\text{G2S}}. \tag{10}$$

Meanwhile, given that tasks are offloaded to SAN for processing, each SAN node $s$ should make sure that the computation resource required by its workload is no larger than its computation capacity. Let $F_s$ be the floating point operations per second (FLOPS) of node $s$. Therefore, for each SAN node $s \in \mathcal{S}$, its computation capacity constraint is formalized as follows,

$$\sum_{a \in \mathcal{A}} \sum_{k \in \mathcal{K}_a} x_{k,s_{\max}}(t) f_{n_a} \leq F_s. \tag{11}$$

At last, for the energy consumption constraint, that for each UE $u \in \mathcal{U}$, its overall energy consumption during $T$ time slots is no larger than its electricity capacity $E_u^{\max}$. The energy consumption constraint of each UE is formulated as follows,

$$\sum_{t \in \mathcal{T}} \sum_{k \in \mathcal{K}_u} \sum_{s \in \mathcal{S}} x_{k,s}(t) E_{u,s}^k(t) \leq E_u^{\max}. \tag{12}$$

*3) QoS Constraints:* The end-to-end delay of a task consists of two parts: transmission and computation. Let $D_a$ be the maximum latency tolerable for class $a$ tasks, then based on the transmission model and the computation model we introduce in Section III, for any task in application class $a \in \mathcal{A}$, the expected E2E delay constraint of task $k$ at time slot is formalized as follows,

$$\begin{aligned} & x_{s_{\min}}^k(t) D_{u,s_{\min}}^{\text{tr},k}(t) + x_{s_{\min}}^k(t) x_{s_{\max}}^k(t) D_{s_{\min},s_{\max}}^k(t) \\ & + x_{s_{\max}}^k(t) D_{a,s_{\max}}^{\text{cmp},k}(t) \leq D_a^{\max}. \end{aligned} \tag{13}$$

Now we come to another QoS guarantee constraint, inference accuracy constraint. Let $\epsilon_a$ be the minimum expected prediction accuracy on the selected object class $a$, then the expected inference accuracy $\rho_k(t)$, which is a function of $\sigma_k(t)$, is only acceptable if

$$\rho_k(t) x_{s_{\min}}^k(t) x_{s_{\max}}^k(t) \geq \epsilon_a, \quad \forall k = (u, a, \kappa) \in \mathcal{K}. \tag{14}$$

### B. Problem Formulation

We first review the decision variables of the optimization problem: task association matrix $\mathbf{x}$; UE transmit power matrix $\mathbf{p}$; UE bandwidth allocation matrix $\mathbf{b}$; and task compression scaling factor matrix $\sigma$.

As we mentioned in the introduction, there exists a tradeoff between the system energy consumption and the total number of accepted tasks. The SemSAN system energy consumption consists of two parts, transmission and computation energy consumptions. The total transmission energy consumption over all time slots is formalized as $E_{\text{tr}} = \sum_{t \in \mathcal{T}} \sum_{u \in \mathcal{U}} \sum_{s_{\min} \in \mathcal{S}_u^{\text{IN}}} \sum_{k \in \mathcal{K}_u} x_{s_{\min}^k} E_{u,s_{\min}}^{\text{tr},k}(t)$, while the total computation energy consumption over all time slots is formalized as $E_{\text{cmp}} = \sum_{t \in \mathcal{T}} \sum_{s_{\max} \in \mathcal{S}_a^{\text{MOD}}} x_{s_{\max}}^k(t) E_{u,s}^{\text{cmp},k}(t)$.

Meanwhile, the total number of admitted tasks within $T$ time period is formalized by,

$$M = \sum_{t \in \mathcal{T}} \sum_{k \in \mathcal{K}} \sum_{s_{\min}, s_{\max} \in \mathcal{S}} x_{s_{\min}}^k(t) x_{s_{\max}}^k(t) \tag{15}$$

At this point, our optimization problem can be described as *how to process as many as DL tasks as possible within the time period T, ensuring that the SAN not only meets the QoS requests of the selected tasks but also minimizes the system's energy consumption.* It can be formalized as follows,

$$\begin{aligned} \max_{\mathbf{x},\mathbf{p},\mathbf{b},\sigma} \quad & \eta \frac{M}{K} + (1-\eta) \frac{E_{\text{tr}} + E_{\text{cmp}}}{E_{\max}} \\ \text{s.t.} \quad & (2), (9) - (14), \end{aligned} \tag{P1}$$

where $\eta \in [0,1]$ is a balancing factor defining the relative weight of total admitted number of tasks and energy, $E_{\max} = \sum_{u \in \mathcal{U}} E_u^{\max} + \sum_{s \in \mathcal{S}} E_s^{\max}$ is the maximal available energy of the SAN system.

## IV. SemSAN Slicing Scheme

### A. SemSAN Slicing Problem Analysis

Four keen observations are concluded as follows to inspire our approach to solving (P1). Please note that we only provide qualitative proof for these conclusions due to space concern.

**Theorem 1.** *The optimal solution to (P1) is achieved if and only if the compression factor $\sigma$ is the minimum that satisfies the accuracy requirement $\epsilon_a$ from (14), irrespective of the values of other varibales. That is, $\sigma_k^* = \min_{\sigma_k} \sigma_k$ s.t., $\rho_k(\sigma) > \epsilon_a$.*

*Proof:* If we set $\mathbf{x}, \mathbf{b}, \mathbf{p}$ as their optimal values $\mathbf{x}^*, \mathbf{b}^*, \mathbf{p}^*$, then the objective of (P1) becomes a linear function of $\sigma_k(t)$. Meanwhile, the left-hand-sides of (10) – (13) are all monotonically increasing over $\sigma_k(t)$. As a result, to maximize the objective function, it is desired for the compression factor $\sigma_k(t)$ to take its minimum value that guarantees the expected model inference accuracy $\epsilon_a$. □

**Theorem 2.** *The SemSAN slicing problem (P1) is NP-hard.*

*Proof:* We prove the result by showing that (P1) is an instance of the binary multi-dimentional knapsack problem (0/1 d-KP), which is NP-hard [8]. Based on Theorem 1, the original problem (P1) can be simplified w.r.t. three decision variables $\mathbf{x}$, $\mathbf{p}$ and $\mathbf{b}$. Consider an extreme case when there are no placement constraints. Then the simplified (P1) is reduced to be an instance of 0/1 d-KP, where there are multiple resources (i.e., bandwidth and CPU/GPU) to be considered. □

Now that we have a simplified version of (P1), two other theorems are introduced to analyse the simplified (P1)'s characteristics.

**Theorem 3.** *Once a task is allowed to be transmitted, its delay is monotonically decreasing w.r.t. the allocated bandwidth and transmission power of that task, respectively. That is, if $x_k(t) = 1$, then the delay $D_k(t) = D_{u,s_{\min}}^{tr,k}(t) + D_{s_{\min},s_{\max}}^k(t) + D_{a,s_{\max}}^{cmp,k}(t)$ of task k at time slot t is monotonically decreasing w.r.t. the bandwidth allocation $b_{u,s}^k(t)$ and the transmit power $p_{u,s}^k(t)$, respectively.*

*Proof:* Recall the expression of delay function $D_k(t)$, we find that for a given value of $p_{u,s}(t)$, $D_k(t)$ is a function

**Algorithm 1:** Greedy SemSAN Slicing Algorithm

```
$\mathcal{K}_c \leftarrow \mathcal{K}$ ▷ Consider all tasks candidate for admission
for for all $k \in \mathcal{K}_c$ do
    if $\exists \sigma_k^*$ then
        $\sigma_k \leftarrow \sigma_k^*$
        $\mathbf{p}_k \leftarrow p_{\max}$
    else
        $\mathcal{K}_c \leftarrow \mathcal{K}_c \setminus k$
    end
end
for $t = 1$ to $T$ do
    $\mathbf{x}_{s_{\max}}, \mathcal{U}_c \leftarrow$ B&B($\mathcal{U}(t)$)
    $b_{u,s}^k \leftarrow$ Progress Filling($\mathcal{U}_c$)
end
```

of $b_{u,s}(t)$, and the first derivative of $D_k(t)$ w.r.t. $b_{u,s}(t)$ is computed by

$$\begin{aligned}\frac{\partial D_k(t)}{\partial b_{u,s}(t)} &= -\underbrace{\frac{\sigma_k^*(t)\phi_k q_k}{\left[b_{u,s}(t)\log_2\left(1+\frac{p_{u,s}(t)H_{u,s}(t)}{b_{u,s}(t)I_0}\right)\right]^2}}_{\Omega_1} \\ &\times \left[\log_2\left(1+\frac{p_{u,s}(t)H_{u,s}(t)}{b_{u,s}(t)I_0}\right) - \frac{p_{u,s}(t)H_{u,s}(t)}{b_{u,s}(t)I_0+p_{u,s}(t)H_{u,s}(t)}\right] \\ &< -\Omega_1 \times \left[\frac{\frac{p_{u,s}(t)H_{u,s}(t)}{b_{u,s}(t)I_0}}{1+\frac{p_{u,s}(t)H_{u,s}(t)}{b_{u,s}(t)I_0}} - \frac{p_{u,s}(t)H_{u,s}(t)}{b_{u,s}(t)I_0+p_{u,s}(t)H_{u,s}(t)}\right] \\ &= -\Omega_1 \times 0 = 0, \end{aligned} \tag{17}$$

where the inequality is derived from the fact that $\log_2(1+x) > \frac{x}{1+x}$ and $\Omega_1 > 0$. Therefore, $D_k(t)$ is monotonically decreasing w.r.t. $b_{u,s}^k(t)$. Follow the same logic, the first derive of $D_k(t)$ w.r.t. $p_{u,s}^k(t)$ is proved to be smaller than 0. Consequently, the theorem is proved. □

**Theorem 4.** *Once a task is allowed to be transmitted, its transmitter's energy consumption is monotonically decreasing w.r.t. the allocated bandwidth and transmission power of that task, respectively.*

*Proof:* Recall the expression of $E_{u,s}^k(t)$, we find that for a given value of $b_{u,s}^k$, $E_{u,s}^k(t)$ is a function of $p_{u,s}(t)$, and the first derivative of $E_{u,s}^k(t)$ w.r.t. $p_{u,s}^k(t)$ is computed by

$$\frac{\partial E_{u,s}^k(t)}{\partial p_{u,s}(t)} = -\frac{[\sigma_k^*(t)\phi_k q_k]^2 H_{u,s}^k(t) p_{u,s}^k(t)}{[r_{u,s}^k(t)]^3(1+\gamma_{u,s}^k(t))} < 0. \tag{18}$$

Follow the same logic, the first derive of $E_{u,s}^k(t)$ w.r.t. $b_{u,s}^k(t)$ is also proved to be smaller than 0. Consequently, the theorem is proved. □

### B. Greedy Online Algorithm for SemSAN Slicing

In practice, tasks are generated online, meanwhile their movements and visible set of satellites are unlikely to predict. As a result, it is impractical to make a plan ahead for future arriving tasks. Moreover, constraints (10), (11) and (13) are defined in each time slot, whereas the remaining (12) is defined over all time slots. The above two reasons inspire us to greedily exhaust the G2S bandwidth or satellite computation resources in each time slot to serve as many as DL tasks as possible to approximate the optimal solutions to (P1).

Such a resource exhaustion leads us to the decoupling of (P1), where $\mathbf{x}$, $\mathbf{b}$ and $\mathbf{p}$ are coupled. We decouple (P1) into three sub-problems *task processing association*, *bandwidth allocation* and *transmit power scheduling*, which are detailed as follows, meanwhile the greedy algorithm is shown in Alg. 1.

**Task processing assocation**: We first regard $\mathbf{b}$ and $\mathbf{p}$ as constants, then the task processing association is a muti-KP (MKP) that is defined on $x_{s_{\max}}^k(t)$, where there are multiple knapsacks (i.e., satellites) and multiple UEs. Each UE has multiple tasks that needs to be placed in the knapsacks such that their total weights (i.e., the objective of (P1)) can be maximized. This MKP problem can be solved by classic algorithms such as Branch-and-Bound (B&B) techniques [9], readily available within well-established solvers, e.g., CPLEX and MATLAB.

**Bandwidth allocation**: The optimal solution of $x_{s_{\max}}^k(t)$ indicates the possible UEs and tasks that may be processed in time slot $t$. With these possible tasks, our next step is to fix their values of $p_{u,s}^k(t)$ to the maximum $p_{\max}$. This is because, according to Theorem 4, $E_{u,s}^k(t)$ is monotonically decreasing w.r.t. $p_{u,s}^k(t)$, and the maximum of $p_{u,s}^k(t)$ leads to the minimum energy consumption on UE $u$. At this point, (P1) becomes a bandwidth allocation problem. From Theorem 3 we know that the larger the bandwidth $b_{u,s}^k(t)$ is, the smaller the delay $D_{u,s}^k(t)$ UE $u$ experiences. Therefore, in the bandwidth allocation problem, all participating UEs are attempting to maximize their allocated bandwidth. The optimal bandwidth solution is achieved using the progressive filling method [10], where all UEs increase their bandwidth at the same speed until the available G2S link bandwidth is fully occupied.

**Transmit power scheduling**: With the optimal solutions obtained from the above two subproblems, each UE is aware of the number of tasks that would be executed in each time slot. As a result, an UE will transmit the desired number of tasks in each time slot at its maximum power $p_{\max}$ until its battery is depleted.

## V. Performance Evaluation

### A. Setup

*1) Dataset and DNN models:* We consider the object detection problem in CV. Specifically, we consider (i) the widely known Common Objects in Context (COCO) [11] as the dataset, which is a large-scale image database containing more than $200K$ labeled examples across 80 object classes; (ii) the YOLOX classifier series [12], where tiny-YOLOX, nano-YOLOX, and regular YOLOX are considered.

*2) System Models:* Consider a SAN with 10 to 60 UEs that generate computation-intensive DL tasks randomly on the ground. Each UE generates 30 DL tasks every minute on average. There are 12 LEO satellites distributed over eight circular orbits at 600 km with the Walker constellation. In each

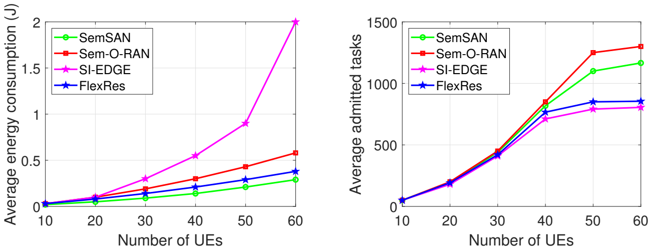

Fig. 2: System performance w.r.t. 10 to 60 number of UEs. (a) Average energy consumption in each time slot, and (b) average admitted task number in each time slot.

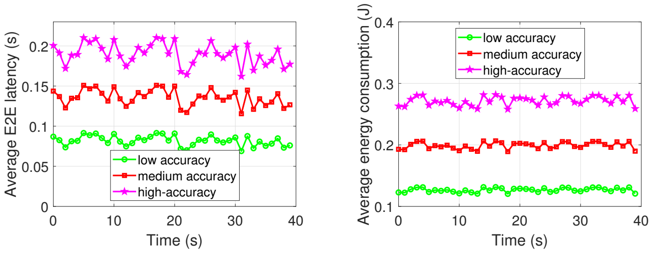

Fig. 3: System performances w.r.t. model inference accuracy where the UE number is 40. (a) Average E2E latency experienced by tasks, and (b) average system energy consumption.

time slot, an UE is only capable to access 3 of its nearest satellites, and the distance between an UE to its accessible satellites is randomly distributed between 1,000 to 2,000 km. For tasks generated by each user, their required accuracy thresholds for model inference are “Low”: 0.35, “Medium”: 0.55, “High”: 0.75, with proportions of 25%, 50%, and 25%, respectively. The length of each time slot $\tau$ is set as 1 s. The G2S link bandwidth is 1 MHz, $p_{\max}$ is 0.01 W, $I_0$ is -174 dB, $R_{\text{ISL}}$ is 10 Mbps, $\eta$ is 0.5, $D_a^{\max}$ is 0.5 s.

*3) Baselines:* Three baselines are considered in (i) SI-EDGE [13], the state-of-the-art algorithm for RAN slicing; (ii) Sem-O-RAN [5], the state-of-the-art semantic O-RAN slicing algorithm that only applies to stationary RANs; (iii) FlexRes, which implements SAN slicing following the greedy SemSAN algorithm but does not consider the semantics.

### *B. Effect of UE numbers on System Performance*

Fig. 2 shows the effect of UE numbers on the average energy consumption and admitted tasks in each time slot. We observe that SemSAN performs the best in average task energy consumption but performs the second best in accommodating task number. This is because SemSAN aims to strike a balance between energy consumption and task number accepted, whereas Sem-O-RAN aims to maximize the task number accepted. Fortunately, the advantage of Sem-O-RAN is not obvious. We attribute this phenomenon to the fact that Sem-O-RAN does not consider the size reduction of DNN models. This further validates the superiority of SemSAN.

### *C. Effect of Model Accuracy on System Performance*

Fig. 3 shows the average E2E delay and energy consumption experienced by tasks that require different accuracies. The results precisely align with our expectations, indicating that when tasks have higher accuracy requirements, they correspondingly consume more energy and take longer for transmission and processing.

## VI. Conclusions

We have proposed SemSAN, a semantic SAN slicing approach for NextG AI-native NTNs. SemSAN slicing is formalized as a combinational optimization problem, with its objective to maximize the total number of DL tasks accepted while still minimizing the total system energy consumption. Meanwhile, the effect brought by the dynamic SAN topology is quantified as placement constraints, while the relationships between task compression factor and model size, task compression factor and accuracy are quantified with data-driven approaches. Moreover, we have proposed an online greedy SemSAN slicing algorithm that not only approaches to the optimal solutions but also easily to be implemented in practice. Extensive evaluation validates the effectiveness of SemSAN in energy saving and supporting substantial number of DL tasks.

## References

[1] M. M. Azari, S. Solanki, S. Chatzinotas, O. Kodheli, H. Sallouha, A. Colpaert, J. F. M. Montoya, S. Pollin, A. Haqiqatnejad, A. Mostaani *et al.*, “Evolution of non-terrestrial networks from 5G to 6G: A survey,” *IEEE communications surveys & tutorials*, 2022.

[2] X. Lin, S. Rommer, S. Euler, E. A. Yavuz, and R. S. Karlsson, “5G from space: An overview of 3GPP non-terrestrial networks,” *IEEE Communications Standards Magazine*, vol. 5, no. 4, pp. 147–153, 2021.

[3] O. Kodheli, E. Lagunas, N. Maturo, S. K. Sharma, B. Shankar, J. F. M. Montoya, J. C. M. Duncan, D. Spano, S. Chatzinotas, S. Kisseleff *et al.*, “Satellite communications in the new space era: A survey and future challenges,” *IEEE Communications Surveys & Tutorials*, vol. 23, no. 1, pp. 70–109, 2020.

[4] M. Polese, L. Bonati, S. D’Oro, S. Basagni, and T. Melodia, “Understanding O-RAN: Architecture, interfaces, algorithms, security, and research challenges,” *IEEE Communications Surveys & Tutorials*, 2023.

[5] C. Puligheddu, J. Ashdown, C. F. Chiasserini, and F. Restuccia, “SEM-O-RAN: Semantic and flexible O-RAN slicing for NextG edge-assisted mobile systems,” in *IEEE International Conference on Computer Communications (INFOCOM)*, 2023.

[6] N. Lazarev, T. Ji, A. Kalia, D. Kim, I. Marinos, F. Y. Yan, C. Delimitrou, Z. Zhang, and A. Akella, “Resilient baseband processing in virtualized rans with slingshot,” *ACM Special Interest Group on Data Communication (SIGCOMM)*, pp. 654–667, 2023.

[7] M. Drumond, T. Lin, M. Jaggi, and B. Falsafi, “Training DNNs with hybrid block floating point,” *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 31, 2018.

[8] H. Kellerer, U. Pferschy, D. Pisinger, H. Kellerer, U. Pferschy, and D. Pisinger, *Multidimensional knapsack problems*. Springer, 2004.

[9] L. A. Wolsey, *Integer programming*. John Wiley & Sons, 2020.

[10] D. Bertsekas and R. Gallager, *Data networks*. Athena Scientific, 2021.

[11] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, “Microsoft coco: Common objects in context,” *European Conference on Computer Vision (ECCV)*, pp. 740–755, 2014.

[12] Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun, “Yolox: Exceeding yolo series in 2021,” *arXiv preprint arXiv:2107.08430*, 2021.

[13] S. D’Oro, L. Bonati, F. Restuccia, M. Polese, M. Zorzi, and T. Melodia, “Sl-EDGE: Network slicing at the edge,” *International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing (MobiHoc)*, pp. 1–10, 2020.