LEONS: Multi-Domain Network Slicing Configuration and Orchestration for Satellite-Terrestrial Edge Computing Networks

Haitham H. Esmat*, Beatriz Lorenzo*, and Jianqing Liu+
*Dept. Electrical and Computer Engineering, University of Massachusetts Amherst, USA
+Dept. of Computer Science, North Carolina State University, USA
{habdelhafez, blorenzo}@umass.edu

***Abstract*—In this paper, we present a multi-domain network slicing scheme for satellite-terrestrial edge computing networks (STECNs) that admits different slice configurations. Each slice is configured to include terrestrial-air, terrestrial-satellite, terrestrial-air-satellite, or terrestrial-air-satellite-gateway domain topologies. However, the multi-domain nature of STECNs makes slicing especially challenging since the cross-domain orchestrator has no knowledge of the resource availability in different domains. Our goal is to design an algorithm that builds a belief in resource availability to jointly optimize the slice configuration, service level agreement (SLA) decomposition, routing, and resource allocation. We model the slice/resource availability as a Markov process to track the probability of achieving the SLA per configuration. To solve the multi-domain slicing problem, the cross-domain orchestrator interacts with the configuration coordinator to define an index-based slice configuration policy based on restless multi-armed bandits (RMABs), which is aware of the network traffic. The configuration coordinator decomposes the SLA and each domain controller solves the optimum routing and resource allocation. Our slicing scheme is evaluated using five typical application scenarios for STECNs. Simulation results show that our scheme achieves six times higher reward than agnostic schemes and efficiently performs multi-domain slicing with low complexity.**

***Index Terms*—multi-domain network slicing, STECNs, energy cost, SLA, restless multi-armed bandit (RMAB).**

I. Introduction

The sixth generation of wireless networks (6G) is expected to provide global coverage by integrating terrestrial and non-terrestrial communications [1]. To realize this vision, several private and public entities are deploying mega-constellations of small-size Low Earth Orbit (LEO) satellites equipped with edge computing [2]. Satellite networks should be virtualized to be integrated with current 5G terrestrial networks and support different applications. Network slicing has been investigated in 5G as a promising virtualization technique where the infrastructure is shared by multiple tenants (operators) to serve multiple service classes simultaneously [3]. Logical slices are created through the terrestrial infrastructure and include communication, computing, and storage resources. Each slice becomes an independent virtual network expected to guarantee certain service level agreement (SLA) and provide complete network functionalities. However, slicing in STECNs is more challenging than in terrestrial networks due to: a) different service providers and administrative domains involved in creating slices to ensure coverage across large geographic areas, b) the SLA needs to be decomposed dynamically in different domains as the STECNs network topology evolves, and c) the multi-tier nature of STECNs and varied requirements increase the complexity of resource management and orchestration. Existing works ignore the multi-domain aspect of slicing in STECNs and, thus, they lack adaptability to exploit available edge computing resources at different domains and achieve different performance tradeoffs.

In [4] a satellite edge computing architecture is analyzed and a slice scheduling algorithm is proposed to access satellite servers for various IoT applications. They formulate the offloading problem as a multi-objective optimization problem with respect to latency, computational power, and transmission power attenuation. In [5], an automatic network slicing framework for ultra-dense CubeSat systems is presented to address routing and resource allocation with minimal SLA violations. Their objective is to find the optimal gateways and CubeSats required per slice, along with the corresponding allocation of resources. Drif et al. [6] integrate satellite systems as a transport network between core networks and the 5G terrestrial radio access and present primitives for slicing the space segment (LEO, GEO, and MEO). NS of STECNs applied to vehicular networks is investigated in [7], [8]. Lyu et al. [7] present an online control framework to slice spectrum in satellite-air-terrestrial networks and solve the problem of request admission and scheduling, UAV dispatching, and resource slicing using Lyapunov optimization theory. Wu et al. [8] investigate resource slicing and scheduling to create delay-sensitive and delay-tolerant slices for vehicular networks. However, these works have overlooked multi-domain orchestration and management needed to guarantee end-to-end performance.

In this paper, we present a multi-domain network slicing framework for STECNs that supports multi-providers and different slice configurations. Each slice is configured to include terrestrial-air, terrestrial-satellite, terrestrial-air-satellite, or terrestrial-air-satellite-gateway domain topologies. We formulate the slice configuration selection problem in the form

This work is partially supported by the US National Science Foundation under Grant CNS-2008309 and CNS-2225427.

ICC 2023 - IEEE International Conference on Communications | 978-1-5386-7462-8/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICC45041.2023.10278769

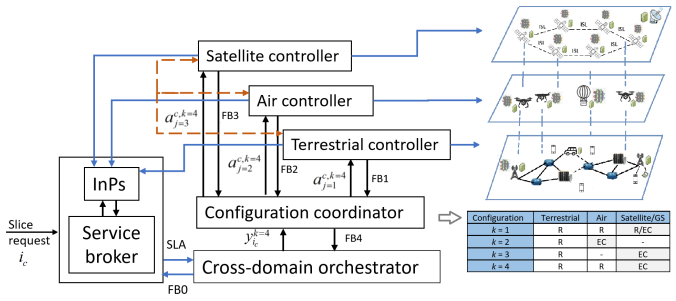

Fig. 1: Multi-domain NS scheme for STECNs and illustration of SLA decomposition $a_j^{c,k=4}$ when configuration $k = 4$ is selected ($y_{i_c}^{k=4} = 1$).

of a Restless Multi-Armed Bandit (RMAB) [9], [10], which enables efficient and low complexity implementation. The problem is solved by an online adaptive orchestration algorithm in which the cross-domain orchestrator interacts with the configuration coordinator to build a belief in resource availability and SLA achievement per configuration. The goal is to maximize the reward in terms of users' requirements, as specified in the SLA, and the energy consumption.

II. System Model

We consider a Satellite-Terrestrial Edge Computing Network (STECN) which consist of terrestrial, air, and satellite domains owned/operated by different service providers. The terrestrial domain includes IoT devices, terrestrial edge computing (EC) facilities, and base stations. The IoT devices collect data from the environment and offload their computing tasks to EC enabled nodes at the terrestrial, UAVs or satellite domains to save battery. The air domain consists of UAVs that collect and execute data from IoT devices on-demand, and the satellite domain consists of a constellation of LEO satellites that perform computing tasks, provide backhaul services to terrestrial users that cannot achieve the requested performance by an overloaded or unavailable terrestrial network, and can offload tasks using inter-satellite links (ISLs) to other satellites or to the ground station (GS) with connection to a cloud center.

A multi-domain network slicing scheme, as shown in Fig. 1, is developed to support heterogeneous service classes $c \in \mathcal{C}$ in STECN. A slice request i_c arrives at the service broker, which negotiates the price with the requesting tenant for the slice and the SLA. We assume the SLA is given in terms of throughput, delay and reliability. The request is forwarded to the cross-domain orchestrator, which analyzes the service requirements and selects the slice configuration based on the expected performance. A slice configuration defines a slicing topology which includes resources from different domains i.e., terrestrial-air, terrestrial-satellite, terrestrial-air-satellite, and terrestrial-air-satellite-GS. For example, delay-tolerant data from an IoT device can be relayed by the satellite network to the gateway, and the computing task can be served at the gateway cloud (configuration 1). Likewise, intermediate UAVs (configuration 2) and satellites (configurations 3 and 4) can process delay demanding tasks to reduce latency. Configuration decisions involve slicing communication, computing, and storage resources at different domains in STECN and should adapt to changes in the environment while saving resources for the most demanding requests. The selection of the slice configuration is forwarded to the configuration coordinator, which interacts with the domain controllers to decompose the SLA among the domains that participate on that configuration. Finally, each domain controller allocates the resources to meet the assigned SLA.

The slice request arrivals from each class c follow the Poisson distribution with rate λ_c and its network resource occupation follows the exponential distribution with rate $1/\mu_c$. The orchestrator is responsible for selecting the configuration $k \in \mathcal{K}$ that will achieve the SLA with the highest probability. Since each domain has each own service provider, the orchestrator is not aware of the state of the available resources in each domain and thus it builds a belief in resource availability based on the traffic and expected performance.

III. LEONS Framework

A. Service Level Agreement Decomposition

Let us assume that LEONS can accommodate a set of configurations $\mathcal{K}$, and each configuration $k \in \mathcal{K}$ spans a set of domains J_k. Each slice of class c has associated an end-to-end SLA $a_{e2e}^c = (T_{e2e}^c, d_{e2e}^c, \rho_{e2e}^c)$, which consist of throughput T_{e2e}^c, delay d_{e2e}^c, and reliability ρ_{e2e}^c requirements. For a given configuration $k \in \mathcal{K}$ selected by the cross-domain orchestrator, the configuration coordinator follows a decomposition rule $G^{c,k}$ to decompose a_{e2e}^c into partial SLAs $a_j^{c,k}$ attributed to each domain $j \in J_k$, $a_{e2e}^c = G^{c,k}(a_j^{c,k}; j = 1, ..., J)$. The decomposition rule indicates how the partial SLAs $a_j^{c,k}$ are combined to form a_{e2e}^c. In particular, the end-to-end throughput is the minimum throughput per domain $T_{e2e}^c = T_{min}^c = min_j\{T_j^{c,k}\}$, the end-to-end delay $d_{e2e}^c = d_{max}^c = \sum_j \xi_j^{c,k} d_{max}^c$, where $\xi_j^{c,k}$ is the decomposition of the delay in domain j under configuration k, and the end-to-end reliability probability $\rho_{e2e}^c = \rho_{min}^c = \prod_j \rho_j^{c,k}$, where $\rho_j^{c,k}$ is the reliability probability per domain j in configuration k.

B. Slice Configuration Selection

Every slice consists of radio, computing and storage resources which are allocated by the domain controllers to satisfy the end-to-end SLA a_{e2e}^c. Let n_c^k denote the number of slices from class c and configuration k served simultaneously in the network, $n_c^k \leq I_c^k$, where I_c^k is the maximum number of slices that can be served simultaneously. Even though there are slices available on a particular configuration, the selected configuration may result in outage, and thus not achieving the SLA. The outage probability $P_{out}^{c,k} = 1 - \rho_j^{c,k} = 1 - \prod_j (1 - Pr(SNR_j^{c,k} < \gamma_j^c))$ is the probability that the SNR is less than a threshold γ_j^c in any domain j in configuration k. If outage occurs, the tenant will request the slice again and the equivalent arrival rate of the new slice requests is $\lambda_c^{'} = \lambda_c/(1 - P_{out})$.

The cross-domain orchestrator dynamically selects the slice configuration k based on the belief in available resources per class and expected performance. However, the cross-domain orchestrator is unaware if transmission will be successful (i.e., no transmission outage) by using each configuration. To improve the selection of the configuration in time, we present a scheme based on RMABs model [9], [10] that builds a belief based on previous experience that the selected configuration will achieve the SLA. At the beginning of time slot t, the orchestrator selects the configuration for each class with the highest success probability and receives a reward if the configuration meets the SLA. We model the resource availability to create slices under each configuration k as a Markov process in which the state at time t is $S^{org}(t) = n_c^k$. The state of the network evolves from slot to slot as a Markov chain with transition matrix $\boldsymbol{P}^{org} = [P_{S^{org};S^{org'}}]$, as illustrated in Fig. 2a, where $x_c^k = m_c^k - z_c^k$ is the difference between the number of slice request arrivals and departures. For tractability of the model, we elaborate a reduced Markov process with two states, as shown in Fig. 2b, $S^{red}(t) = 1$ and $S^{red}(t) = 0$, depending on the probability that the slice request of class c is served successfully or unsuccessfully in configuration k. To calculate the transition probabilities $\alpha_{S^{red};S^{red'}}$, first we reduce the original model of slice availability from I_c^k states to two states '0' and '1' based on the availability or unavailability of slices with probabilities

$$\begin{aligned}
\pi_{00}^{c,k}(t+\Delta t) &= p(I_c^k, t)p(x_c^k \geq 0, \Delta t) \\
\pi_{01}^{c,k}(t+\Delta t) &= p(I_c^k, t)p(x_c^k < 0, \Delta t) \\
\pi_{10}^{c,k}(t+\Delta t) &= \sum\nolimits_{n_c^k=0}^{I_c^k-1} p(n_c^k, t)p(x_c^k \geq I_c^k - n_c^k, \Delta t) \\
\pi_{11}^{c,k}(t+\Delta t) &= \sum\nolimits_{n_c^k=0}^{I_c^k-1} p(n_c^k, t)p(x_c^k < I_c^k - n_c^k, \Delta t)
\end{aligned} \tag{1}$$

where $\pi_{00}^{c,k}(t+\Delta t)$ is the probability that all available slices have been allocated and new requests arrive, $\pi_{01}^{c,k}(t+\Delta t)$ is equal to the probability that all available slices have been allocated but some requests were served, $\pi_{10}^{c,k}(t+\Delta t)$ is the probability that there are n_c^k slices being served and more than $I_c^k - n_c^k$ requests arrive. Finally, the transition probability $\pi_{11}^{c,k}(t+\Delta t)$ is equal to the probability that there are n_c^k slices being served and the number of request arrivals is less than the available slices, $I_c^k - n_c^k$.

The transition matrix of the reduced Markov model $\mathbf{P}^{red} = [\alpha_{S^{red};S^{red'}}]$ with transition probabilities $\alpha_{S^{red};S^{red'}}$ as

$$\begin{aligned}
\alpha_{00}^{c,k}(t+\Delta t) &= \pi_{00}^{c,k}(t+\Delta t) + \pi_{01}^{c,k}(t+\Delta t)P_{out}^{c,k}(\Delta t) \\
\alpha_{01}^{c,k}(t+\Delta t) &= \pi_{01}^{c,k}(t+\Delta t)(1 - P_{out}^{c,k}(\Delta t)) \\
\alpha_{10}^{c,k}(t+\Delta t) &= \pi_{10}^{c,k}(t+\Delta t) + \pi_{11}^{c,k}(t+\Delta t)P_{out}^{c,k}(\Delta t) \\
\alpha_{11}^{c,k}(t+\Delta t) &= \pi_{11}^{c,k}(t+\Delta t)(1 - P_{out}^{c,k}(\Delta t))
\end{aligned} \tag{2}$$

where the transition probability $\alpha_{00}^{c,k}(t+\Delta t)$ from state $S^{red}(t) = 0$ to $S^{red}(t+\Delta t) = 0$ is a probability of serving the slice of class c with configuration k unsuccessfully, $\alpha_{01}^{c,k}(t+\Delta t)$ is the probability that there are slices available and no outage, $\alpha_{10}^{c,k}(t+\Delta t)$ is the probability that there are no slices available or even if they are available there is an outage, and $\alpha_{11}^{c,k}(t+\Delta t)$ is the probability that the slice is served successfully. The steady-state probability $\mathbf{p} = \mathbf{P}^{org}\mathbf{p}$, with $\mathbf{p} = [p(n_c^k)]$, that the original system is in state n_c^k is obtained

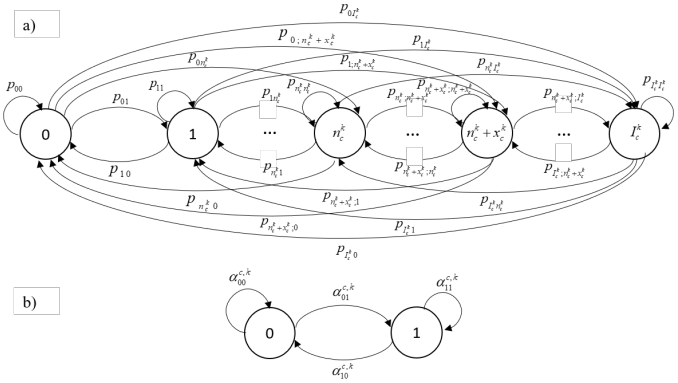

Fig. 2: Markov slice configuration state model: a) Original, b) Reduced.

$$\begin{aligned}
&p(n_c^k) = p_{0n_c^k}p(0) + p_{1n_c^k}p(1) + ... + p_{I_c^k n_c^k}p(I_c^k) = \\
&= \textstyle\sum_{n_c'^k=0}^{I_c^k} p_{n_c' n_c^k}p(n_c'^k)
\end{aligned} \tag{3}$$

where $\sum_{n_c^k} p\left(n_c^k\right) = 1$. The arrival rate of slices class c at configuration k is $\lambda_c^k = \lambda_c p_k$ where λ_c is the arrival rate of slices class c and p_k is the probability of selecting configuration k.

The belief of serving slice request i_c in configuration k successfully is defined by matrix $\boldsymbol{\Omega}(t) = [\omega_1^1(t), ..., \omega_c^k(t), ..., \omega_C^K(t)]$, where $\omega_c^k(t)$ is the conditional probability that $S^{red}(t) = 1$. It has been shown that it suffices for optimal decision-making that the conditional probability of each configuration is in state 1 given all past decisions and observations [10]. Given the observation at time *t*, the belief state in time *t* + 1 can be obtained recursively

$$\omega_c^k(t+1) = \begin{cases} \alpha_{11}^{c,k}, & S^{red}(t) = 1 \\ \alpha_{01}^{c,k}, & S^{red}(t) = 0 \\ \mathcal{T}(\omega_c^k(t)), & otherwise \end{cases} \tag{4}$$

where $\mathcal{T}(\omega_c^k(t)) = \omega_c^k(t)\alpha_{11}^{c,k} + (1 - \omega_c^k(t))\alpha_{01}^{c,k}$ is the belief operator when the slice class c has not been selected in the current slot in configuration k. If no prior information is available regarding the initial system state, the initial belief vector $\Omega(1)$ can be set to the steady state probability that the reduced system is in state 1, $\alpha_1^{c,k}$. The steady-state probabilities of the reduced system are calculated as $\alpha_0^{c,k} = \frac{\alpha_{10}^{c,k}}{\alpha_{01}^{c,k}+\alpha_{10}^{c,k}}$ and $\alpha_1^{c,k} = \frac{\alpha_{01}^{c,k}}{\alpha_{01}^{c,k}+\alpha_{10}^{c,k}}$, where $\alpha_0^{c,k}+\alpha_1^{c,k} = 1$.

C. Problem Formulation

We define the binary variable $y_{i_c}^k(t)$ to indicate the selection of the cross-domain orchestrator, where $y_{i_c}^k(t) = 1$ means it has selected configuration *k* to serve slice request i_c from class c or otherwise, $y_{i_c}^k(t) = 0$. The number of slices being served

simultaneously should not exceed the maximum number of slices, $\sum_{i_c} y_{i_c}^k(t) \leq I_c^k$. The SLA decomposition associated to each domain j is $a_j^{c,k} = (T_j^{c,k}, \xi_j^{c,k}, \rho_j^{c,k})$ with $T_j^{c,k} \geq T_{\min}^c$, and the fraction of e2e delay per domain should not exceed the maximum delay, $\sum_j \xi_j^{c,k} \leq 1$. Each domain controller allocates the communication resources and computing resources to create non-overlapping network slices. We assume that nodes in the same domain have the same computational capabilities F_j. We denote by $f_{i_c j}^k(t)$ the computing allocation ratio to slice i_c in domain j and configuration k, $\sum_{i_c} f_{i_c j}^k(t) \leq 1$. The available bandwidth per domain j is B_j. We denote by $b_{i_c j}^k(t)$ the fraction of the bandwidth allocated to slice i_c, $\sum_{i_c} b_{i_c j}^k(t) \leq 1$. The computing task of each IoT device is routed to the EC node in domain j and the routing matrix $\mathbf{H} = [\mathbf{H}_j]$ includes inter-domain and intra-domain links. Our goal is to jointly optimize slice configuration, SLA decomposition, routing, and resource allocation to maximize the expected total discounted reward,

$$\begin{aligned} \max_{\mathbf{y},\mathbf{a},\mathbf{b},\mathbf{f},\mathbf{H}} \quad & \mathbf{E}\left[\sum_{t=1}^{\infty} \sum_i \beta^{t-1} R_i(t)\right] \\ \text{subject to} \quad & \sum_{i_c} y_{i_c}^k(t) \leq I_c^k,\ \forall k \in \mathcal{K},\ y_{i_c}^k(t) \in \{0,1\} \\ & T_j^{c,k} \geq T_{\min}^c,\ \forall j \in \mathcal{J}_k \\ & \sum_j \xi_j^{c,k} \leq 1,\ \forall j \in \mathcal{J}_k \\ & \sum_{i_c} b_{i_c j}^k(t) \leq 1,\ \forall j \in \mathcal{J}_k,\ b_{i_c j}^k(t) \in \{0,1\} \\ & \sum_{i_c} f_{i_c j}^k(t) \leq 1,\ \forall j \in \mathcal{J}_k,\ f_{i_c j}^k(t) \in \{0,1\} \\ & \mathbf{H} \in \mathcal{H} \end{aligned} \tag{5}$$

where β ($0 < \beta \leq 1$) is the discount factor and $R_i(t)$ is the reward of the orchestrator described in the next section. Solving the previous optimization is complex since it is a combinatorial problem [5]. To make it tractable, we solve it in three steps. First, we solve the cross-domain orchestrator decision by deriving a configuration activation index policy based on Whittle Index [10]. Then, the SLA decomposition is optimized based on the available resources per domain, and routing and resource allocation are solved by using a heuristic.

D. Index-Based Slice Configuration Activation Policy

By modeling the slice configuration activation policy as an RMAB problem, each slice configuration k is treated as an arm. When an arm is activated, the corresponding slice is served by each network domain $j \in \mathcal{J}_k$ associated to configuration k. Since the decision of activating each arm (slice configuration) is made independently, we consider a single slice i and drop the index of the class c and configuration k. To formulate the RMAB, we define the following:

1) *Decision Epochs:* Time is divided into discrete time slots and decisions are made each time t, $t \in \{1, 2, ..., \infty\}$.

2) *State Space:* We assume that each slice tolerates a maximum admission delay $\tau_{max,i}$ and we define a binary indicator q_i that denotes if the slice has been served q_i = 1 or otherwise q_i = 0. The state of the slice admission is $s_i(t) = (\tau_{r,i}(t), q_i)$, where $\tau_{r,i}(t)$ are the remaining admission slots. We assume that one slot is enough to admit the slice. The system state at decision t consists of the states of all slice requests $\boldsymbol{s} = (s_1(t), ..., s_N(t))$.

3) *Action:* At each decision epoch, the cross-domain orchestrator takes action $y_i^k(t)$ to select configuration k to serve slice request i. Let $\boldsymbol{y} = (y_1^1(t), y_1^2(t), ..., y_i^k(t), ..., y_N^K(t))$. If the action $y_i^k(t) = 1$ the configuration k for slice i is activated, otherwise $y_i^k(t) = 0$. Since the capacity and the computing resources are limited, the action taken at any time t should satisfy the constraints $\sum_i \sum_k y_i^k(t) \leq I^k$.

4) *State Transition Probability:* The state of slice i will transfer to the next state with probability $Pr\{\ s_i(t+1)|s_i(t), y_i^k(t)\}$ depending on the current state and action taken. For $\tau_r = 1$, regardless of the action, $Pr\{\ s_i(t+1) \,|s_i(t), y_i^k(t)\} = 1$ with $s_i(t+1) = (\tau_{max,i}, 1)$. Similarly, for $\tau_{r,i} > 1$ and $q_i = 0$, no matter if configuration k is activated or not, $Pr\{\ s_i(t+1) \,|s_i(t), y_i^k(t)\} = 1$ with $s_i(t+1) = (\tau_{r,i} - 1, 0)$. For $\tau_r > 1$ and $q_i = 1$, if the configuration k is not activated, $Pr\{\ s_i(t+1) \,|s_i(t), y_i^k(t)\} = 1$ with $s_i(t+1) = (\tau_{r,i} - 1, 1)$. Meanwhile, if the configuration k is activated, $s_i(t+1) = (\tau_{r,i} - 1, 0)$ with probability ω^k or $s_i(t+1) = (\tau_{r,i} - 1, 1)$ with probability $(1 - \omega^k)$.

5) *Reward:* The reward of slice i at time t depends on the current state and the action taken,

$$R_i\left(s_i(t), y_i^k(t)\right) = \begin{cases} 0, & q_i = 0 \\ y_i^k(t)\left(\omega^k(t)\left(r_i^k(t) - \psi E_i^k(t)\right) - \left(1 - \omega^k(t)\right)\delta\right) + \left(1 - y_i^k(t)\right)\varphi, & q_i = 1, \tau_{r,i}(t) = 1 \\ y_i^k(t)\,\omega^k(t)\left(r_i^k(t) - \psi E_i^k(t)\right), & q_i = 1, \tau_{r,i}(t) > 1 \end{cases} \tag{6}$$

where r_i^k is the expected reward of configuration k, $r_i^k(t) = \left(T_i^k / T_{i,min}\right)^{\vartheta^i} / \left(d_i^k / d_{i,max}\right)^{\varepsilon^i}$, ϑ^i and ε^i are the importance of throughput and delay of the slice i, respectively, ψ is a scaling factor, $E_i^k = E_{T,i}^k + E_{P,i}^k$ is the energy consumption incurred when the slice i is served by configuration k, where $E_{T,i}^k$ and $E_{P,i}^k$ are the transmission energy consumption and processing energy consumption in configuration k to serve the slice i, respectively. $E_{P,i}^k = \sigma_{dev} f_{dev,i}^3$, where σ_{dev} is the effective capacitance coefficient of the computing node (UAV, LEO sat, GS) that computes the task in configuration k and $f_{dev,i}$ is the computing resource allocated. δ is the penalty incurred when the resources are not available, φ is a small positive value when the configuration has not been activated and ω^k is the belief state as in (4).

E. Indexability and Whittle Index Policy

Whittle index policy is obtained as the optimal solution to a Lagrangian relaxation of RMABs [10]. In fact, the RMAB can be decomposed into a single-arm activation problem and thus, it suffices to consider a single arm (i.e., configuration). In each slot, a decision is made on whether to activate the arm based on the concept of subsidy for passivity. A constant subsidy ν that is obtained when the arm is not activated. The orchestrator decides whether to activate an arm or not at

each time t to maximize the total discounted ν-subsidy reward $V_i^\nu(s) = \sum_{t=1}^{\infty} \beta^{t-1} R_i^\nu(s_i(t), y(t))$ with initial state s,

$$R_i^\nu(s_i(t), y(t)) = R_i(s_i(t), y(t)) + \nu \mathbf{1}(y(t) = 0) \quad (7)$$

where $\mathbf{1}(.)$ equals 1 if the expression in the bracket is true, and 0 otherwise. In the sequel, we drop the subscripts i and t without loss of generality. Let $V(s)$ denote the value function which is the maximum expected total discounted reward from a single-arm bandit process when the initial state is s and there are two actions, y = 0 and y = 1, $V(s) = max\{V(s, y = 0), V(s, y = 1)\}$. The expected total discounted reward $V(s; y)$ is obtained when action y is taken at the first slot followed by the optimal policy in future slots as

$$V(s, y = 0) = R(s, 0) + \nu + \sum\nolimits_{s' \in S} \beta p(s'|s, 0) V^\nu(s') \quad (8)$$

$$V(s, y = 1) = R(s, 1) + \sum\nolimits_{s' \in S} \beta p(s'|s, 1) V^\nu(s') \quad (9)$$

The term $p(s'|s, y)$ denotes the probability that slice admission i changes from state s to next state s' when decision y is taken and $V^\nu(s')$ is the total discounted future reward.

Definition 1: The Whittle index $\nu_i(s)$ of an arm i in state s is the infimum subsidy ν that makes the two decisions (activating arm i or not) equally rewarding [9]:

$$\nu_i(s) = \inf_{\nu} \{\nu : V(s, y = 0) \geq V(s, y = 1)\} \quad (10)$$

Definition 2: An arm is indexable if the passive set $Z(\nu) = \{\nu : V(s, y = 0) \geq V(s, y = 1)\}$ of the single-armed bandit process with subsidy ν monotonically increases as ν increases from $-\infty$ to $+\infty$. An RMAB is indexable if every arm is indexable [9].

We study all possible states to establish the indexability and derive the closed-form expression of the Whittle index:
1) When $\tau_r = 1$: *a)* If $q_i = 0$ and $y = 0$, then $V((1, 0), 0) = \nu + \beta V(\tau_{i,max}, 1)$. On the other hand, if $y = 1$, we have $V((1, 0), 1) = \beta V(\tau_{i,max}, 1)$. Therefore, the Whittle index is $\nu(1, 0) = 0$; *b)* If $q_i = 1$ and $y = 0$, we have $V((1, 1), 0) = \varphi + \nu + \beta V(\tau_{i,max}, 1)$, whereas if $y = 1$, we have $V((1, 1), 1) = \omega(r - \psi E) - (1 - \omega)\delta + \beta V(\tau_{i,max}, 1)$. Then, $\nu(1, 1) = \omega(r - \psi E) - (1 - \omega)\delta - \varphi$.
2) When $\tau_r > 1$: *a)* If $q_i = 0$ and $y = 0$, then $V((\tau_r, 0), 0) = \nu + \beta V(\tau_r - 1, 0)$. On the other hand, if $y = 1$, we have $V((\tau_r, 0), 1) = \beta V(\tau_r - 1, 0)$. Therefore, the Whittle index is $\nu(1, 0) = 0$. *b)* If $q_i = 1$ and $y = 0$, we have $V((\tau_r, 1), 0) = \nu + \beta V(\tau_r - 1, 1)$, whereas if $y = 1$, we have $V((\tau_r, 1), 1) = \omega(r - \psi E) + \beta\omega V(\tau_r - 1, 0) + \beta(1 - \omega)V(\tau_r - 1, 1)$. To obtain $\nu(\tau_r, 1)$, we define

$$\begin{aligned} g(\tau_r, 1) &= V((\tau_r, 1), 0) - V((\tau_r, 1), 1)) \\ &= \nu - \omega(\mathrm{r} - \psi \mathrm{E}) + \beta\omega(\mathrm{V}(\tau_\mathrm{r} - 1, 1) - \mathrm{V}(\tau_\mathrm{r} - 1, 0)) \\ &= \nu - \omega(r - \psi E) + \beta\omega M(\tau_r - 1). \end{aligned} \quad (11)$$

Next, we calculate $\frac{\partial g(\tau_r, 1)}{\partial \nu} = 1 + \beta\omega \partial M(\tau_r - 1)/\partial \nu$. If we assume that $\partial M(\tau_r - 1)/\partial \nu \geq -\frac{1}{\beta\omega}$, then we can see that $\partial g(\tau_r, 1)/\partial \nu \geq 0$. By definition 2, this implies indexability under state $(\tau_r, 1)$. Let us prove that $\partial M(\tau_r)/\partial \nu \geq -\frac{1}{\beta\omega}$ is true by induction. The expression of $M(\tau_r)$ is calculated as $M(\tau_r) = V(\tau_r, 1) - V(\tau_r, 0) = max\{V((\tau_r, 1), 0), V((\tau_r, 1), 1)\} - max\{V((\tau_r, 0), 0), V((\tau_r, 0), 1)\}$. Then,

$$M(\tau_r) = \begin{cases} \omega(r - \psi E) - \beta\omega M(\tau_r - 1) \\ +\beta M(\tau_r - 1), & v < 0 \\ \omega(r - \psi E) - v + \\ \beta(1 - \omega) M(\tau_r - 1), & 0 \leq v < v(\tau_r, 1) \\ \beta M(\tau_r - 1), & v \geq v(\tau_r, 1) \end{cases} \quad (12)$$

where $0 < \beta \leq 1$ and $0 \leq \beta(1 - \omega) < 1$. We can observe that $\partial M(\tau_r - 1)/\partial \nu \geq -1/\beta\omega$ for all three cases, which demonstrates the indexability of the configuration activation problem.

Theorem 1: The closed-form Whittle index $\nu(s)$ of an arm in state $s = (\tau_r, q)$ is

$$\nu(\mathrm{s}) = \begin{cases} 0, & \text{if } q = 0 \\ \omega(r - \psi E) - (1 - \omega)\delta - \varphi, & \text{if } q = 1 \text{ and } \tau_\mathrm{r} = 1 \\ \omega(r - \psi E) + \\ \frac{\beta^{t-1}\delta\omega(1-\omega)^{t-1}}{1 - \beta\omega - \cdots - \beta^{t-1}\omega(1-\omega)^{t-2}}, & \text{if } q = 1 \text{ and } \tau_\mathrm{r} > 1 \end{cases} \quad (13)$$

Proof: By definition 1, for a given state s, we can obtain the Whittle index by solving (10). Since previously we have obtained the closed-formed expressions of $V(s, y = 0)$ and $V(s, y = 1)$ for all possible states, we have solved (10) and the Whittle index (13).

F. Joint Slice Configuration, SLA Decomposition, Routing, and Resource Allocation

We design an online iterative algorithm, as described in Algorithm 1, to solve the joint slice configuration, SLA decomposition, routing, and resource allocation ($CSLAR^2$). At each time slot, the orchestrator selects I^k slices with the highest indexes which are calculated based on the expected r_i^k and E_i^k shared by the configuration coordinator (from line 4 to 10 in Algorithm 1). Each domain controller shares with the coordinator the resource availability so the coordinator can optimize the SLA decomposition per domain. The routing and resource allocation for each slice is solved as described in Algorithm 1 (from line 12 to 24). The complexity of calculating all the indexes and sorting for N slice requests and K configurations is $O(NKlog(NK))$. Assuming that there are $|\Xi_{jk}|$ possible values of SLA decomposition per domain j and configuration k, each domain has T_{E2E} paths, and all users Q^i use the shortest path, the complexity is $O(NK|\Xi_1| * .. * |\Xi_j| * .. * |\Xi_{Jk}| * Q^i * T_{E2E} * log(NK))\}$.

IV. Numerical Results

We conduct extensive simulations to show the performance of our schemes. Five service classes are considered with the SLA requirements described in Table I [5]. We compare the performance of our schemes with three algorithms: agnostic, shortest deadline first (SDF), and random. The agnostic algorithm assumes no prior knowledge on the resource availability

Algorithm 1 Joint slice configuration, SLA decomposition, routing and resource allocation (CSLAR^2)

Input: $\beta, q_i, \tau_{max,i}, I^k$, δ, φ, λ, domain topology with limited resources, $(T^i_{min}, d^i_{max}, \rho^i)$: SLA of the slice i, Q^i: Number of users in slice i, $\boldsymbol{Dis}^k_i$ matrix contains the distance between the users in the slice i and the e2e paths ($\mathrm{T_{E2E}}$).
Initialization: $\tau_r = \tau_{max,i}$, $q_i = 1$, $\boldsymbol{Routing}^k_i = 0$.
Output: $I^k slices$, $\{\mathrm{a}^{\mathrm{i,k}}_1, .., \mathrm{a}^{\mathrm{i,k}}_{J_k}\}$, $\{\mathrm{P}^{\mathrm{i,k}}_1, \mathrm{P}^{\mathrm{i,k}}_2, \ldots, \mathrm{P}^{i,\mathrm{k}}_{Q^i}\}$: set of best routing paths ($\mathbf{Routing}^k_i$), the amount of resources allocated to the slice i ($\boldsymbol{RA}^k_i$).
Online slice configuration.
For $e = 1 : N$
  While $\tau_{r,e} > 0$ and $q_e = 1$
    Orchestrator receives r^k_e and E^k_e from the configuration coordinator.
    Orchestrator calculates Whittle index as in (13).
  end
end
Orchestrator activates I^k slices with the highest indexes and calculates (7).
$\tau_{r,i} = \tau_{r,i} - 1$
Jointly SLA decomposition, routing, and resource allocation.
For every $\mathrm{a}^{\mathrm{i,k}}_1, .., \mathrm{a}^{\mathrm{i,k}}_{J_k}$
  For l = 1: Q^i and e2e path = 1 : T_{E2E}
    Find the minimum distance in $\boldsymbol{Dis}^k_i$ (e.g, $(l_1, e2e_1)$).
    If $\mathbf{Routing}^k_i(l_1, :) == 0$ & $\boldsymbol{Dis}^k_i(l_1, e2e_1) \neq \infty$ & enough resources
      Allocate resources to user l_1.
      $\mathbf{Routing}^k_i(l_1, e2e_1) \leftarrow 1$, $\boldsymbol{Dis}^k_i(l_1, e2e_1) \leftarrow \infty$
      $Tl_1 = min_j\{T^k_j\}$, $d_{l_1} = \sum_j d^k_j$, calculate R_{l_1} as in (6).
    else
      $\boldsymbol{Dis}^k_i(l_1, e2e_1) \leftarrow \infty$.
    end
  end
  Total $R(\mathrm{a}^{\mathrm{i,k}}_1, .., \mathrm{a}^{\mathrm{i,k}}_{J_k}) = \sum_{l=1}^{Q^i} R_l$.
end
Get the optimal $\mathrm{a}^{\mathrm{i,k}}_1, .., \mathrm{a}^{\mathrm{i,k}}_{J_k}$, $\boldsymbol{RA}^k_i$ and $\boldsymbol{Routing}^k_i$.

Table I The SLA parameters for different classes.

Class	ϑ	ε	T_{min} (MbPs)	d_{max} (ms)	ρ (%)	packet size (kbit)	I_c	λ_c	μ_c
Emergency communications	0.3	0.7	40	150	99.99	1	9	12	4
In-space cellular backhaul remote connectivity	0.3	0.7	100	600	99.9	250	6	8	4
Remote industrial automation	0.2	0.8	120	300	99	20	7	10	4
Monitoring and reconnaissance	0.6	0.4	100	700	99.9	250	4	6	4
Traffic efficiency in vehicle communication	0.5	0.5	50	1200	99.999	20	5	14	4

and success probability of the configuration. The SDF serves first the slices with less remaining admission time. Finally, the random algorithm serves the slices by choosing a random configuration. Different amount of resources and priorities per class are considered to analyze the performance. We conducted Monte Carlo simulations over 1000 realizations in Matlab using the parameters in Table II, which are described below.

To describe the physical mode at each segment of the satellite-terrestrial network, we define the channel capacity from transmitter (Tr) to receiver (Re) as $W_{Tr,Re}\log_2\left(1+\frac{P_{Tr}\ h_{Tr,Re}}{N_0 W_{Tr,Re}}\right)$, where P_{Tr} is a transmitter power, $h_{Tr,Re}$ is the channel gain between Tr and Re, and $W_{Tr,Re}$ is the allocated bandwidth. The channel gain between user equipment (UE) and Sat is $h_{UE,Sat} = \frac{G_{UE}G_{Sat}(10)^{-(\frac{A}{10})}}{(dis_{UE,Sat})^{\alpha}}$, where G_{UE} and G_{Sat} are the transmitter (UE) antenna gain and the receiver (Sat) antenna gain, respectively; $dis_{UE,Sat}$ is the distance between UE and Sat, and A [dB] denotes

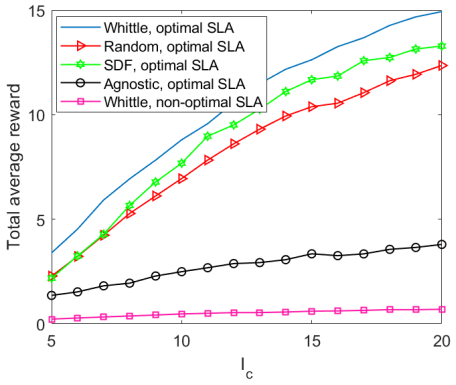

Fig. 3: Total average reward versus I_c.

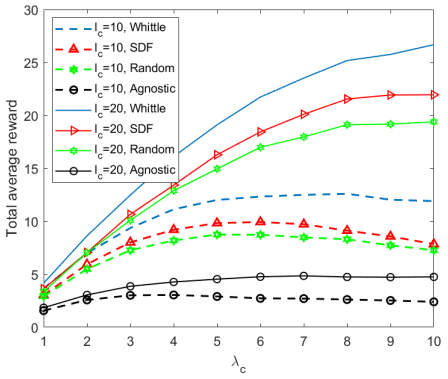

Fig. 4: Total average reward versus λ.

the rain attenuation. The channel gain between UAVs is $h_{UAV_1,UAV_2} = \frac{G_{UAV_1}G_{UAV_2}}{(dis_{UAV_1UAV_2})^{\alpha}}$, where α is the path loss exponent. The channel gain between LEO satellites is $h_{Sat_1,Sat_2} = \frac{G_{Sat_1}G_{Sat_2}v^2}{k_B\left(4\pi.S_{Sat_1Sat_2}.f_c\right)^2}$, where v is the speed of the light, f_c is the carrier frequency, k_B is the Boltzmann's constant, and $S_{s_1s_2}$ is the slant range (in km). The channel gain between UAV and LEO satellite is $h_{UAV,Sat} = \frac{G_{UAV}G_{Sat}v}{4\pi \mathrm{f_c} dis_{UAV,Sat}K_B}$. The channel gain between Sat and GS is $h_{Sat,GS} = \frac{G_{Sat}G_{GS}v}{4\pi \mathrm{f_c} dis_{Sat,GS}K_B}$, and the channel gain between UE and UAV is $h_{UE,UAV} = 10^{-(L_{UE,UAV}/10)}$, where $L_{UE,UAV}$=$\mathrm{P}_{LoS}(t) \times \mathrm{PL}_{LoS}(t)$+$P_{NLoS}(t) \times \mathrm{PL}_{NLoS}(t)$, where $P_{LoS}(t) = \left(\frac{1}{1+ae^{(-b(-12+\sin^{-1}((100-\mathrm{a})/dis_{UE,UAV}))}}\right)$ is the probability of LoS, $PL_{LoS}(t) = \left(20\log(f_c) + 20\log\left(\frac{4\pi}{L}\right) + 20\log(dis_{UE,UAV}(t)) + \eta_{LoS}\right)$ is the LoS pathloss, $PL_{NLoS}(t) = \left(20\log f_c + 20\log\left(\frac{4\pi}{L}\right) + 20\log(dis_{UE,UAV}(t)) + \eta_{NLoS}\right)$ is the NLoS pathloss, $P_{NLoS}(t) = 1 - P_{LoS}(t)$ K_B is the Boltzmann's constant and a, b are constants, which depend on the environment. ϑ is the required CPU cycles per bit of computation, ∂ is the writing speed of the device (UAV/sat/GS), and σ_{dev} is the effective capacitance coefficients of the device (UAV/sat/GS).

In Fig. 3 we show the average reward for different values of $I^k_c = I_c$ for all schemes when λ_c = 5 and μ_c = 7. We can see that our scheme outperforms existing schemes and achieves

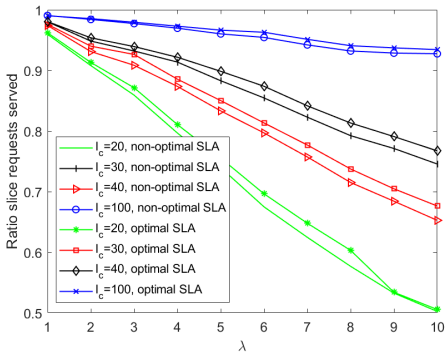

Fig. 5: Slice requests served per class versus I_c.

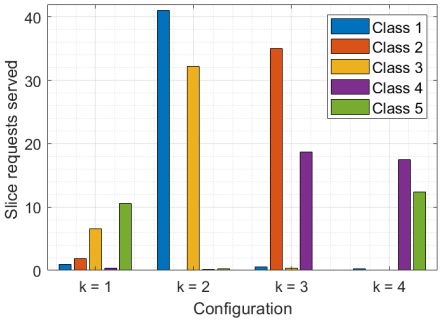

Fig. 6: Slice requests served per configuration and class for priority $\boldsymbol{O}$ = [0.1 0.6 0.1 0.1; 0.3 0.05 0.4 0.1; 0.2 0.15 0.2 0.1; 0.1 0.1 0.2 0.4; 0.3 0.1 0.1 0.3].

a reward four times higher than the agnostic scheme. When the number of available slices I_c increases our algorithm can efficiently serve the slice requests to maximize the reward. In fact, our algorithm considers the future performance in the allocation decision. We can see also that the reward decreases drastically when using a given non-optimal SLA decomposition as compared to jointly optimizing the SLA and configuration. In Fig. 4, the average reward is presented for different values of λ. We can see that the improvement achieved with our scheme is even more significant as the network congestion increases, and resources become scarce. However, the reward for SDF and random algorithms decreases faster with congestion. Besides, our scheme achieves three to six times higher reward than the agnostic scheme when there are twice the available resources. In Fig. 5 we show the ratio of slice requests served using our algorithm for different λ and I_c and with optimal and non-optimal SLA decomposition. Although the reward significantly decreases when the SLA decomposition is not optimal, the number of slices served slightly decreases. This demonstrates that our algorithm can estimate the belief even when there is outage.

The slice requests served per class and configuration are shown in Fig. 6 for $I_c^k = I_c = 20$. We assume slices are allocated following the priority matrix O described in the legend. Slice requests from classes 1 and 3 which require the smallest delay are served mainly by configuration 2. On the other hand, classes 4 and 5 tolerate the highest delay and thus, they are served by configurations 3 and 4, respectively. Finally, class 2 is mainly served by configurations 3 and 1. Since configuration 1 has the highest delay and energy cost and lowest reliability, fewer requests are served using this configuration. Similarly, since the reliability in configuration 3 is higher than in 4, and the power consumption is higher in the latter more requests are served by configuration 3.

Table II Main parameter settings for simulations

Parameter	Value	Parameter	Value
k_B	1.38×10^{-23}J/K	$dis_{UE,UAV}, dis_{UAV,UAV}$	2000,800 m
η_{LoS}, η_{NLoS}	1, 20	a, b	12, 0.135
ϑ	$10^3 cycles/bit$	$dis_{GS,GS}, dis_{Sat,Sat}$	1000, 73000 m
$\partial_{\text{UAV}}, \partial_{\text{Sat}}, \partial_{\text{GS}}$	80, 200, 550 Mbps	$dis_{UE,Sat}, dis_{UAV,Sat}, dis_{GS,Sat}$	160 km
$\sigma_{\text{UAV}}, \sigma_{\text{Sat}}, \sigma_{\text{GS}}$	$10^{-27}, 10^{-28}, 5*10^{-26}$	f_c, N_0	$2.4GHz, -174dBm/Hz$

V. Conclusion

In this paper we have developed a multi-domain network slicing scheme for STECNs which operates with multiple slice configurations. A framework to jointly optimize the configuration selection, SLA decomposition, routing, and resource allocation is presented. Since the cross-domain orchestrator has no knowledge of the available resources per domain, we design an algorithm based on RMABs that builds a belief on resource availability and SLA achievement. Extensive simulations have been conducted using five typical application scenarios in STECNs to show the performance of our approach. We have shown that our algorithm achieves up to six times higher reward than existing approaches and selects the optimal configuration based on the expected outage and network condition.

References

[1] X. Zhu and C. Jiang, "Integrated satellite-terrestrial networks toward 6g: Architectures, applications, and challenges," *IEEE Internet of Things Journal*, vol. 9, no. 1, pp. 437–461, 2022.

[2] M. N. Sweeting, "Modern small satellites-changing the economics of space," *Proceedings of the IEEE*, vol. 106, no. 3, pp. 343–361, 2018.

[3] M. Chahbar, G. Diaz, A. Dandoush, C. Cérin, and K. Ghoumid, "A comprehensive survey on the e2e 5g network slicing model," *IEEE Transactions on Network and Service Management*, vol. 18, no. 1, pp. 49–62, 2021.

[4] T. Kim, J. Kwak, and J. P. Choi, "Satellite edge computing architecture and network slice scheduling for iot support," *IEEE Internet of Things Journal*, vol. 9, no. 16, pp. 14938–14951, 2022.

[5] A. Kak and I. F. Akyildiz, "Towards automatic network slicing for the internet of space things," *IEEE Transactions on Network and Service Management*, vol. 19, no. 1, pp. 392–412, 2022.

[6] Y. Drif, E. Chaput, E. Lavinal, P. Berthou, B. Tiomela Jou, O. Gremillet, and F. Arnal, "An extensible network slicing framework for satellite integration into 5g," *International Journal of Satellite Communications and Networking*, vol. 39, no. 4, pp. 339–357, 2021.

[7] F. Lyu, P. Yang, H. Wu, C. Zhou, J. Ren, Y. Zhang, and X. Shen, "Service-oriented dynamic resource slicing and optimization for space-air-ground integrated vehicular networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 7469–7483, 2022.

[8] H. Wu, J. Chen, C. Zhou, J. Li, and X. Shen, "Learning-based joint resource slicing and scheduling in space-terrestrial integrated vehicular networks," *Journal of Communications and Information Networks*, vol. 6, no. 3, pp. 208–223, 2021.

[9] B. Lorenzo, F. J. González-Castaño, L. Guo, F. Gil-Castiñeira, and Y. Fang, "Autonomous robustness control for fog reinforcement in dynamic wireless networks," *IEEE/ACM Transactions on Networking*, vol. 29, no. 6, pp. 2522–2535, 2021.

[10] K. Liu and Q. Zhao, "Indexability of restless bandit problems and optimality of whittle index for dynamic multichannel access," *IEEE Transactions on Information Theory*, vol. 56, no. 11, pp. 5547–5567, 2010.