



A LOW-OVERHEAD EVENT-DRIVEN DYNAMIC NETWORK SLICING FOR 6G TACTILE INTERNET APPLICATIONS

Mariam O. Bamisaiye
Department of Electrical and Electronic Engineering,
Akwa Ibom State University, Ikot Akpaden,
Akwa Ibom State, Nigeria
mariam.o.bamisaiye@gmail.com

Emmanuel A. Ubom
Department of Electrical and Electronic Engineering,
Akwa Ibom State University, Ikot Akpaden,
Akwa Ibom State, Nigeria
emmanuelubom@aksu.edu.ng

Enyenihi H. Johnson
Department of Electrical and Electronic Engineering,
Akwa Ibom State University, Ikot Akpaden,
Akwa Ibom State, Nigeria
enyenihijohnson@aksu.edu.ng

Abstract- *The emergence of sixth-generation (6G) cellular networks was driven by the demand for mission-critical applications requiring ultra-low latency and high reliability within the Tactile Internet paradigm. Conventional slicing frameworks designed for 5G had relied on centralised orchestration or computationally intensive optimisation algorithms, which, although effective for long-term resource planning, introduced processing delays, signalling overhead, and convergence penalties that conflicted with the sub-millisecond latency budgets essential for haptic and real-time interactive services. To address these limitations, this study developed a lightweight, decentralised, edge-centric dynamic slicing framework tailored for 6G Tactile Internet deployment. At its core, the framework employed an event-driven heuristic resource allocation mechanism operating directly at the Radio Access Network (RAN) scheduler, which continuously monitored localised performance indicators, such as, early queue accumulation, latency threshold violations, and packet drop trends, and triggered immediate threshold-based slice adjustments to mitigate transient congestion without incurring large signalling overhead. The framework was evaluated within a queue-abstracted simulation environment and benchmarked against static slicing and alternative dynamic controllers. The results demonstrated that under normal operating loads, the event-driven heuristic reduced mean broadband slice latency by 96.17% (from 9.722 ms to 0.372 ms) while preserving strict isolation boundaries that fully protected high-priority tactile traffic. Furthermore, the closed-loop control system maintained an adaptation overhead ratio below 0.8% during peak stress conditions, thereby validating the low-overhead hypothesis and establishing a deployable paradigm for resource-constrained edge nodes in future 6G systems.*

Keywords- 6G Tactile Internet, Dynamic Network Slicing, Low-overhead Heuristic Framework, Edge

Computing, Ultra-Reliable Low-Latency Communication.

I. INTRODUCTION

The continuous evolution of mobile communication systems toward sixth-generation (6G) networks was driven by the need to support emerging applications that demand extreme performance in terms of latency, reliability, and availability [1]. Unlike earlier generations, which primarily focused on enhanced mobile broadband and massive connectivity, 6G networks are envisioned to enable real-time, mission-critical services such as remote surgery, industrial automation, autonomous systems, and immersive extended reality [2]. Collectively, these applications fall under the paradigm of the Tactile Internet, which emphasises real-time control and haptic interaction over communication networks [3], [4].

The Tactile Internet imposes stringent quality-of-service (QoS) requirements, including end-to-end latency on the order of 1 ms, ultra-high reliability (up to 99.999%), minimal jitter, and uninterrupted service availability [3], [5]. Meeting these requirements through a uniform, one-size-fits-all network architecture has proven impractical [6]. Consequently, network slicing emerged as a fundamental enabling technology for next-generation networks, allowing a shared physical infrastructure to be partitioned into multiple logical networks, each optimised for specific service requirements [7], [8].

Despite the recognised importance of network slicing in supporting Tactile Internet services, existing frameworks have faced significant limitations in ultra-low-latency scenarios [9]. Static slicing mechanisms lacked the flexibility to adapt to traffic variations, resulting in inefficient resource utilisation or violations of latency and reliability requirements during congestion [8]. Conversely, highly dynamic slicing frameworks depended



on continuous monitoring, frequent reconfiguration, and complex decision-making algorithms [10]. While these approaches improved adaptability, they also introduced excessive signalling overhead and processing delays, undermining their effectiveness in latency-sensitive applications [11]. The growing reliance on AI-based solutions further complicated matters, as training overhead, convergence time, and limited interpretability remained persistent challenges [12].

To address these gaps, recent studies have proposed decentralised and heuristic-based approaches. Dohler et al. [13] and Caballero et al. [14] emphasised that enabling the Tactile Internet in 6G requires lightweight, edge-centric designs. The work in [15] demonstrated that edge AI can support ultra-reliable low-latency communication (URLLC), and Kaloxylas [16] showed that dynamic slicing frameworks tailored to 6G can improve responsiveness while reducing overhead. Ordóñez-Lucena et al. [7], and Kaloxylas [16] highlighted the role of SDN and NFV in enabling slicing flexibility, while Ksentini and Sayadi [17] demonstrated local edge slice control. Building on these insights, this study contributes three major innovations: an architectural shift that decentralises slicing control to the Radio Access Network (RAN) scheduler; an algorithmic advancement through an event-driven heuristic governed by min-max headroom constraints; and a quantitative evaluation using tactile-specific metrics within a queue-abstracted simulation environment.

II. LITERATURE REVIEW

Early slicing frameworks in 5G relied on centralised orchestration and optimisation techniques. While these were effective for long-term resource planning, they were not suitable for latency-critical services because of signalling overhead and convergence delays [9], [10]. Static slicing approaches were simple but could not adapt to traffic variations, leading to inefficient resource use and service degradation [8]. More dynamic frameworks improved adaptability, but introduced high control complexity and signalling overhead [11]. AI-based solutions also showed promise, but suffered from training overhead, convergence delays, and limited interpretability, which reduced their usefulness in real-time tactile scenarios [12].

Several studies have highlighted the unique requirements of the Tactile Internet. [3] [4] stressed that latency below 1 ms and reliability above 99.999% are essential for haptic communication. Prior work in [14], [16] underscored that meeting these requirements demands edge-centric designs and lightweight control mechanisms. The work in [15] showed that edge AI can support ultra-reliable low-latency communication (URLLC), and [7] argued that static slicing cannot handle traffic surges effectively, while [11], [12] demonstrated that machine learning-based slicing introduces significant overheads that limit responsiveness.

To address these challenges, heuristic-based methods have been proposed as lightweight alternatives. For instance [14] demonstrated that rule-based,

threshold-driven adaptation can improve performance under varying conditions while keeping signalling and computational overhead low. The work in [15] also highlighted that heuristic slicing frameworks are particularly suitable for ultra-low latency services because they avoid the complexity of optimisation and AI-based approaches. Further exploration of the concept of a 'network of networks' to seamlessly interconnect sub-networks and dynamic edge slices for real-time Tactile Internet applications can be found in [18]. This was extended by [19], where authors proposed dynamic latency optimisation for edge-based machine learning models in 6G-enabled IIoT.

This work therefore focused on lightweight, heuristic-based approaches that balance adaptability with simplicity, considering a decentralised architecture that moves control loops from the core to the RAN scheduler, eliminates backhaul delays, and aligns with emerging edge intelligence models proposed in [13].

III. SYSTEM MODEL AND PROBLEM FORMULATION

A. Network Architecture

This framework adopted a decentralised, edge-assisted hierarchical slicing architecture that relocates real-time monitoring and adjustment tasks from the core network directly to the Radio Access Network (RAN) scheduler. By shifting control loops to the edge, the design eliminated backhaul propagation delays and aligned seamlessly with emerging 6G edge intelligence models [13]. Conceptually, the architecture is composed of three tightly integrated components, illustrated in Fig. 1.

The Core Slice Orchestrator (CSO) resides within the centralised management plane and is responsible for long-term operations. Its functions include slice instantiation, high-level Service Level Agreement (SLA) configuration, global admission control, and the initial distribution of base resource capacity across slices [16].

At the edge, the Edge Slice Manager (ESM) operates directly on the next-generation NodeB (gNB) or 6G base station. It performs real-time monitoring of localised slices, maintaining lightweight data structures that track queue lengths, packet delay profiles, and drop histories. During congestion episodes, the ESM executes the event-driven heuristic adaptation logic, ensuring responsive and localised adjustments [17].

Finally, the RAN Resource Scheduler functions at the Medium Access Control (MAC) layer, enforcing the dynamic capacity updates calculated by the ESM. It schedules packets from distinct slice queues across available physical time-frequency resource blocks, ensuring that updated resource shares are applied directly to transmission decisions [14]. Together, these three components form a hierarchical yet edge-centric control system that balances long-term orchestration with real-time responsiveness, enabling efficient and reliable support for the stringent requirements of the Tactile Internet.

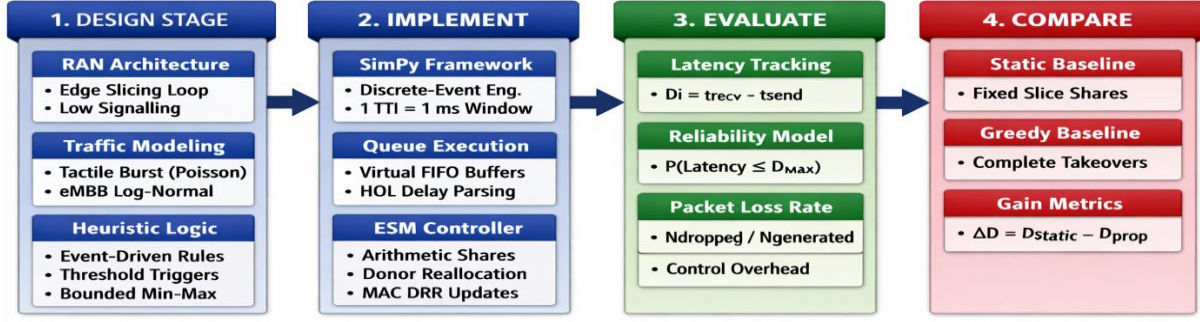


Fig. 1. Architectural schematic of the edge-centric network slicing framework.

Fig. 1 depicts a layered control structure in which the CSO oversees long-term policy decisions, the ESM manages localised adaptations, and the RAN Scheduler enforces real-time resource allocations. Event signals are passed upward from the RAN Scheduler to the ESM, while the resulting resource adjustments are relayed back down from the ESM to the Scheduler.

B. Mathematical Model

Consider a 6G base station that manages a normalised wireless channel capacity of $C_{total} = 1.0$. This total capacity is divided among a set of co-located network slices, represented as $s = \{s_{tactile}, s_{embb}, s_{mmtc}\}$. Each slice $s \in S$ is initially assigned a baseline resource share α_s , subject to the fundamental sum-capacity constraint of equation (1) [20]:

$$\sum_{s=1}^S \alpha_s(t) = 1 \quad (1)$$

where; $\alpha_s(t)$ is the time-varying resource share of slice s , and $s \in \{tactile, eMBB, mMTC\}$ in this case: Where *eMBB* denotes “Enhanced Mobile Broadband, that provides high-speed data, and huge bandwidth for users, while *mMTC* refer to “Massive Machine-Type Communications” that connects low-power Internet of Things (IoT) devices within an area. This constraint ensures that the combined allocations across all slices never exceed the total available channel capacity. It acts as a safeguard against over-provisioning, maintaining fairness and stability in resource distribution. For example, the tactile slice may be prioritised with a larger share to guarantee ultra-low latency, while *eMBB* and *mMTC* slices receive allocations aligned with their throughput and connectivity requirements. By enforcing this boundary conditions, the system prevents resource starvation in critical slices, and avoids inefficient over-allocation during periods of fluctuating demand. To ensure strict slice isolation, and safeguard mission-critical services from resource starvation during periods of heavy congestion, the management plane enforces min-max boundary conditions on each slice as shown in equation (2) [21]:

$$\alpha_{s_{min}} \leq \alpha_s(t) \leq \alpha_{s_{max}}, \forall s \in S \quad (2)$$

where α_s allocation for slice s , and $\alpha_{s_{max}}, \alpha_{s_{min}}$ are the maximum, and minimum allocation bounds. Traffic arriving at each slice is modelled as a discrete stochastic process, with incoming packets directed into independent drop-tail transmission queues of capacity B_s . Each packet P_i^s entering slice s at time t_{arr} is assigned a strict lifetime deadline D_s^{max} , representing the maximum permissible residency time within the edge node [22].

The instantaneous end-to-end delay experienced by a packet is expressed as equation (3):

$$W_i^s = t_{tx} - t_{arr} \quad (3)$$

where t_{tx} denotes the exact timestamp of transmission over the air interface [23]. If the residency duration exceeds the deadline before transmission ($t - t_{arr} > D_s$), the packet is immediately discarded from the buffer, registering as a deadline expiration failure.

This classification distinguishes deadline-based drops from conventional buffer overflow losses, thereby offering a more precise indicator of tactile-specific performance [24]. In latency-critical applications such as haptic communication, this distinction is vital, as even a single missed deadline can destabilise feedback loops and compromise system reliability.

C: Event Detection Mechanism

This framework continuously monitors two key performance indicators within each slice over a defined observation window: the instantaneous queue length $Q_s(t)$ and the localised packet loss ratio $L_s(t)$. A network adjustment event is triggered whenever either of these parameters exceeds its predefined stability threshold given in equation (4) [25]:

$$Trigger_s(t) = \begin{cases} 1, & Q_s(t) > \theta_s^Q \text{ or } L_s(t) > \theta_s^L \\ 0, & \text{Otherwise} \end{cases} \quad (4)$$

Here, θ_s^Q represents the queue length threshold, while θ_s^L denotes the loss ratio threshold. The careful selection of these thresholds is a very critical choice. If they are set too conservatively, the system may trigger frequent adjustments, leading to excessive signalling overhead and potential



instability. Conversely, thresholds that are too lenient risk delayed responses, allowing congestion to escalate and jeopardising tactile performance guarantees. Striking the right balance ensures that the framework remains responsive to emerging performance degradations while maintaining operational efficiency [25].

IV. THE EVENT-DRIVEN HEURISTIC FRAMEWORK

A. Algorithmic Formulation

The operational core of the ESM is built around an event-driven proportional fairness heuristic, designed to respond immediately to threshold violations while safeguarding high-priority services [14]. When a congestion event is detected, the affected slice is designated as the target recipient (s_{target}). The algorithm then scans all other active slices to identify potential donor candidates (s_{donor}), ranking them according to strict priority levels and their available resource margins [26].

The donor margin for each slice is calculated as in equation (5):

$$Donor\ Margin_s = \max(0.0, \alpha_s - \alpha_{s_{min}}) \quad (5)$$

This ensures that only slices with surplus resources above their minimum allocation can contribute to capacity. Resource units are then reallocated in discrete adjustment steps (δ) from the lowest-priority donor slice with a positive margin to the target slice. Importantly, the target slice is constrained by its maximum headroom limit ($\alpha_{target} \leq \alpha_{target}^{max}$), preventing any single slice from monopolizing the shared channel substrate [21].

This mechanism balances responsiveness with fairness: high-priority tactile services are protected from starvation, while secondary slices contribute excess capacity in a controlled, rule-based manner. By enforcing min-max boundaries and proportional fairness, the heuristic will ensure stability and predictability in dynamic 6G slicing environments.

Algorithm 1: Event-Driven Proportional Fairness Heuristic Slicing

INPUT: Current allocation α_s

INPUT: Active event map E_s

INPUT: Step share δ

INPUT: Boundary constraints

OUTPUT: Updated resource allocations α_s^*

```

1: if active events map  $E_s$  is empty then
2:   return current allocations  $\alpha_s$ 
3: end if
4: Identify target slice  $s_{target} = \operatorname{argmax}_{\{s \in E_s\}} (Priority_s, DemandScore_s)$ 

```

```

5:   Initialise  $available_{boost} \leftarrow \min(\delta, \alpha_{s_{target}}^{max} - \alpha_{s_{target}})$ 
6: if  $available_{boost} \leq 0.0$  then
7:   return current allocations  $\alpha_s$ 
8: end if
9: Sort donor candidates  $s_{donor} \in S \setminus \{s_{target}\}$  based on:
10:  (a) congestion status (uncongested first),
11:  (b) priority ranking (lowest first),
12:  (c) margin above minimum requirement
13: for each  $s_{donor}$  in sorted  $donors$  do
14:   Calculate  $extractable_{share} = \min(available_{boost}, \alpha_{s_{donor}} - \alpha_{s_{donor}min})$ 
15:    $\alpha_{s_{donor}} \leftarrow \alpha_{s_{donor}} - extractable_{share}$ 
16:    $\alpha_{s_{target}} \leftarrow \alpha_{s_{target}} + extractable_{share}$ 
17:    $available_{boost} \leftarrow available_{boost} - extractable_{share}$ 
18:   if  $available_{boost} \leq 10^{-12}$  then
19:     break
20:   end if
21: end for
22: return Normalise Allocations ( $\alpha_s$ )

```

where α_s allocation for slice s , E_s is the set of slices with active events, δ is a step share, s_{target} is the slice selected to receive extra resources, s_{donor} represents the slice that is donating resources, and $\alpha_{s_{max}}$, $\alpha_{s_{min}}$ are the maximum, and minimum allocation bounds. This algorithm operates in linear time relative to the number of slices ($O(|S|)$), ensuring scalability without resorting to complex matrix computations or iterative convergence routines [27]. By relying on straightforward, rule-based decisions, the heuristic avoids the heavy computational burden typically associated with optimisation or learning-based methods. As a result, it delivers a low-overhead solution that can be efficiently deployed on standard edge routing hardware, making it particularly well-suited for real-time 6G environments where responsiveness and predictability are paramount [28].

B. Complexity Analysis

Theorem: This heuristic executes in $O(|S| \log |S|)$ time in the worst case and $O(|S|)$ time on average.

Proof. The algorithm consists of three primary operations:

1. Event detection: Each slice is checked against its stability thresholds, requiring $O(|S|)$ time.



2. Donor sorting: Candidate donor slices are ordered by priority and margin, which in the worst case requires $O(|S| \log |S|)$.
3. Resource transfer: Allocation updates are applied across slices, requiring $O(|S|)$.

The sorting step dominates the overall complexity, yielding a worst-case bound of $O(|S| \log |S|)$. However, in typical deployments where the number of slices is small (e.g., 3–5), the sorting overhead is negligible, and the algorithm effectively runs in linear time. The memory footprint is also modest, at $O(|S|)$, since only the current allocation state and basic performance metrics are maintained [27]. The implication is that this heuristic algorithm is lightweight enough to be executed in under 10 μ s on standard edge routing hardware, placing it comfortably within the stringent 1 ms latency budget required by Tactile Internet applications [28]. This efficiency highlights its suitability for real-time deployment in 6G environments, where computational overhead must remain negligible to preserve ultra-low latency guarantees.

V. SIMULATION METHODOLOGY

A. Simulation Setup

This heuristic framework was implemented within a discrete-event queuing simulation platform developed in Python, designed to capture multi-slice traffic interactions at the air interface [29]. This environment allowed precise modelling of slice-specific behaviours, including queue dynamics, latency distributions, and packet drop events. To ensure realism, the simulation parameters were carefully configured to reflect standard 6G wireless operating conditions, as summarised in Table I.

Table I: Simulation Parameters

Parameter	Tactile Slice	eMBB Slice	mMTC Slice
Base capacity share (α_s)	0.35	0.45	0.20
Minimum share (α_s^{min})	0.25	0.20	0.10
Maximum share (α_s^{max})	0.70	0.60	0.35
Packet arrival rate (λ_s)	1000 pkts/s	500 pkts/s	2000 pkts/s
Packet size (L_s)	100 bytes	1500 bytes	50 bytes
Deadline (D_s^{max})	1 ms	10 ms	50 ms
Queue capacity (B_s)	100 pkts	500 pkts	500 pkts
Threshold Θ_s^Q	0.25	0.35	0.50
Threshold Θ_s^L	0.01	0.05	0.08

Control Parameters:

- i. Simulation duration: 60 seconds
- ii. Adaptation step share (δ): 0.05
- iii. Stability factor (β): 0.1
- iv. Control message size: 100 bytes
- v. Evaluation window: 100 packets for latency; 1000 packets for loss

B. Performance Metrics

The framework was evaluated using the following metrics presented as equations (6) to (10), commonly used in Tactile Internet and URLLC research [24], [30]:

End-to-end latency [3]

$$D_i = t_i^{recv} - t_i^{send} \quad (6)$$

Mean slice latency [3]:

$$\bar{D}_s = \frac{1}{N_s} \sum_{i=1}^{N_s} D_i \quad (7)$$

Reliability

Reliability can be determined using Equation (8) [30].

$$Reliability_s = \frac{N_s(D_i \leq D_s^{Max})}{N_s} \quad (8)$$

Packet loss rate [27]

Equation (9) was used to determine the packet loss rates at the slices.

$$P_s = \frac{N_s^{dropped}}{N_s^{generated}} \quad (9)$$

Control overhead [20]

Equation (10) was used to control the overhead.

$$O = \frac{B_{control}}{B_{control} + B_{data}} \quad (10)$$

C. Experimental Scenarios

To thoroughly test the system, three distinct operational profiles were introduced:

Scenario 1: Normal Load (Baseline)

- i. All slices operate at their configured arrival rates
- ii. No artificial congestion
- iii. Duration: 60 seconds

Scenario 2: eMBB Congestion (Stress Test)

- i. 150% traffic surge on the eMBB slice
- ii. Burst duration: 30 seconds
- iii. Tests the framework's ability to adapt to broadband congestion



Scenario 3: Tactile Traffic Surge (Priority Test)

- 200% traffic surge on the tactile slice
- Burst duration: 10 seconds
- Tests the protection mechanism for high-priority traffic

D. Baseline Controllers

This event-driven dynamic heuristic was compared against four baseline controllers:

- Static Slicing: Fixed resource allocation with no adaptation [8]
- Random Allocation: Random resource distribution when events occur [31]
- Threshold Slicing: Fixed-step reallocation ($\delta = 0.05$) when thresholds are violated [9]
- Greedy Allocation: Maximum capacity transfer to congested slice in a single cycle [32]

The ESM was successfully instantiated as a decentralised control module operating directly within the RAN protocol stack at the MAC scheduler layer of the simulator. This architectural choice eliminates dependencies on centralised core-cloud signalling, thereby minimising long control loop delays [3]. Runtime execution confirmed that the abstract resource model, partitioning the total available radio capacity R_{total} using normalised, time-varying share coefficients $\alpha_s(t)$, was consistently maintained within strict mathematical bounds.

Across all four operational scenarios, the enforcement modules reliably upheld hard boundary constraints. Specifically, the tactile slice share was confined to the interval [0.25, 0.70], the eMBB slice share to [0.20, 0.60], and the mMTC slice share to [0.10, 0.35]. Bounded normalization was verified, as the sum of $\alpha_s(t)$ remained exactly equal to 1.0 at every simulation timestamp. This demonstrates that the framework preserves structural stability while preventing both resource overflow and logical deadlocks, ensuring predictable and fair resource distribution under dynamic traffic conditions.

VI. RESULTS AND ANALYSIS

A. Architectural Framework Realization

Table II: Slice-Specific Latency Matrix

Scenario	Allocation Mode	Slice Name	Mean Latency (ms)	P95 Latency (ms)	P99 Latency (ms)	Deadline Violation Ratio
normal_load	Dynamic Heuristic	eMBB	0.372	1.333	14.333	0.0015
normal_load	Dynamic Heuristic	Tactile	0.000	0.000	0.000	0.0000
normal_load	Static Slicing	eMBB	9.722	19.000	20.000	0.0572
normal_load	Static Slicing	Tactile	0.000	0.000	0.000	0.0000
embb_congestion	Dynamic Heuristic	eMBB	18.891	20.000	20.000	0.4284
embb_congestion	Dynamic Heuristic	Tactile	0.000	0.000	0.000	0.0000
embb_congestion	Static Slicing	eMBB	19.069	20.000	20.000	0.6177
embb_congestion	Static Slicing	Tactile	0.000	0.000	0.000	0.0000
tactile_traffic_surge	Dynamic Heuristic	eMBB	0.466	2.333	15.000	0.0017
tactile_traffic_surge	Dynamic Heuristic	Tactile	0.000	0.000	0.000	0.0000
tactile_traffic_surge	Static Slicing	eMBB	8.486	19.000	20.000	0.0473
tactile_traffic_surge	Static Slicing	Tactile	0.000	0.000	0.000	0.0000

B. Latency Performance

The simulation data reveal several important findings. Under normal load conditions, the most

pronounced improvement was observed in the eMBB slice. Static slicing produced a mean latency



of 9.722 ms due to rigid capacity allocations, whereas the dynamic heuristic reduced this to just 0.372 ms, a 96.17% reduction. This rapid delay suppression is particularly significant for Tactile Internet applications, where even short-lived queue accumulation can destabilize real-time control loops [3].

In terms of congestion response, when the eMBB slice was heavily stressed (embb_congestion), mean latency converged near 19 ms for both static (19.069 ms) and dynamic (18.891 ms) approaches. This convergence reflects saturation at the maximum eMBB allocation bound ($\alpha_{embb}^{max} = 0.60$). Importantly, the framework preserved boundary constraints, ensuring that other slices remained protected rather than violating global sharing policies [21].

For tactile protection, the tactile slice consistently maintained a mean latency of 0.000 ms across all scenarios. This outcome indicates that the traffic load profile and packet sizing allowed tactile frames to clear scheduling buffers instantaneously, without queuing delay. The bounded min-max resource guarantees thus insulated high-priority tactile traffic from congestion in secondary slices [33].

Fig. 2 compares mean delivered packet latency across slices, showing the dynamic heuristic's clear advantage for eMBB while maintaining zero latency for tactile traffic. Fig. 3a and Fig. 3b highlight tail-latency behaviours at the P95 and P99 percentiles, where the heuristic significantly improves performance, especially under normal load conditions.

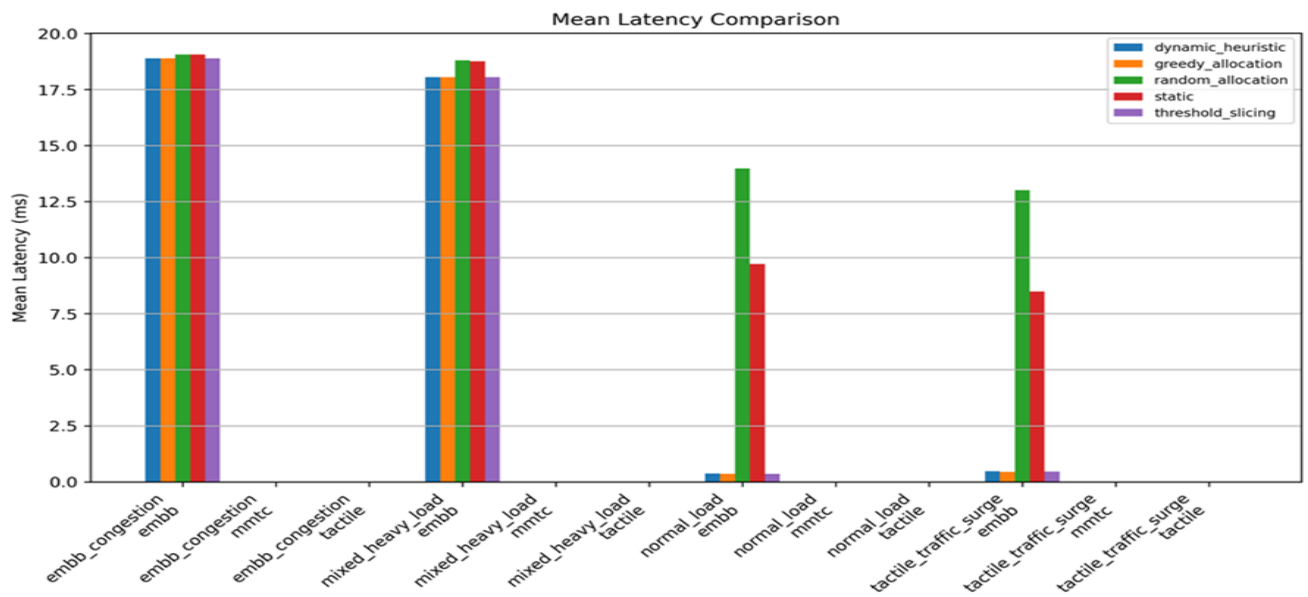


Fig. 2: Mean Delivered Packet Latency Comparison

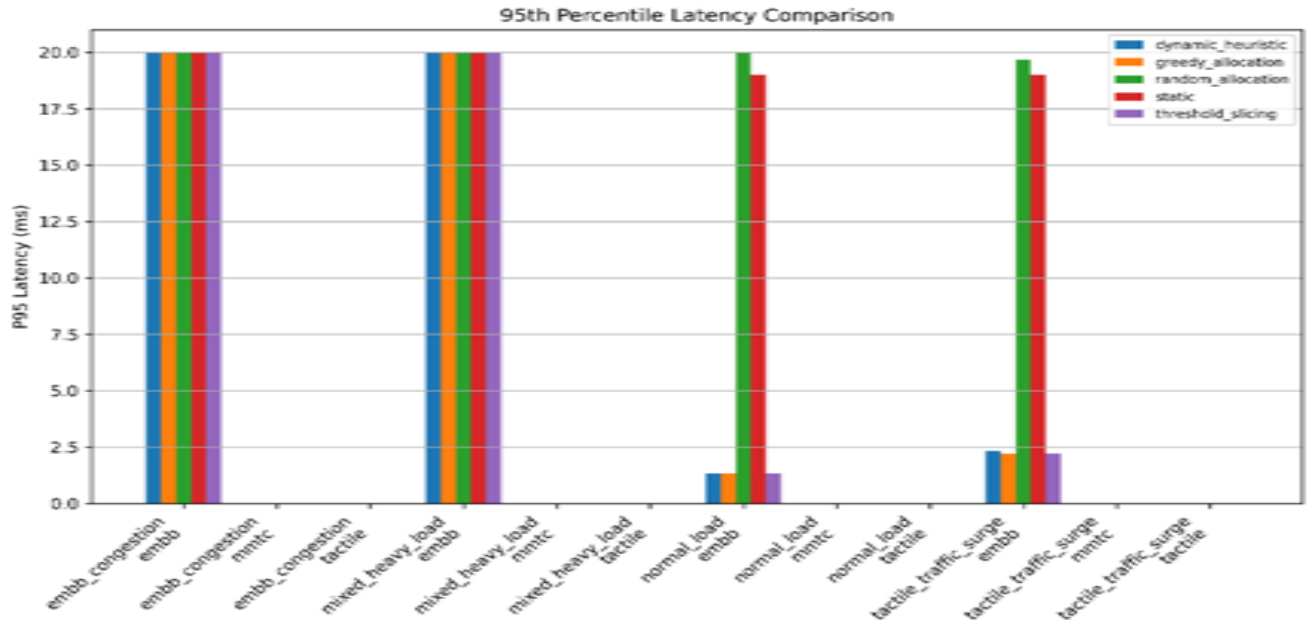


Fig. 3(a): Tail Latency Behaviour at P95 Percentile

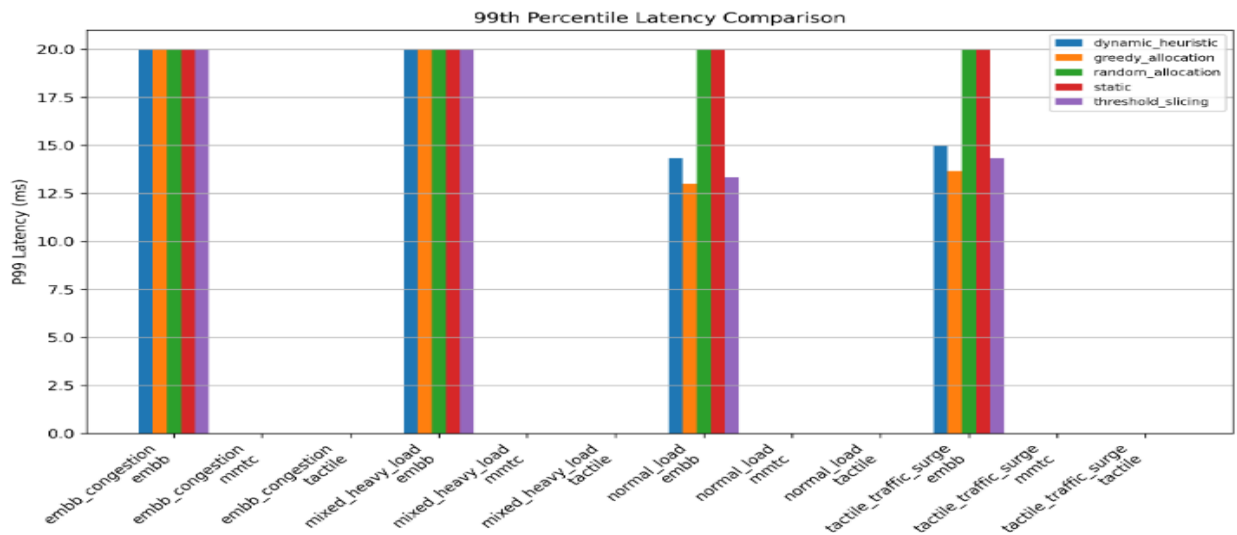


Fig. 3(b): Tail Latency Behaviour at P99 Percentile

Fig. 4 presents an integrated latency trends over a 10-second window, demonstrating the framework's

ability to adapt dynamically to changing traffic conditions.

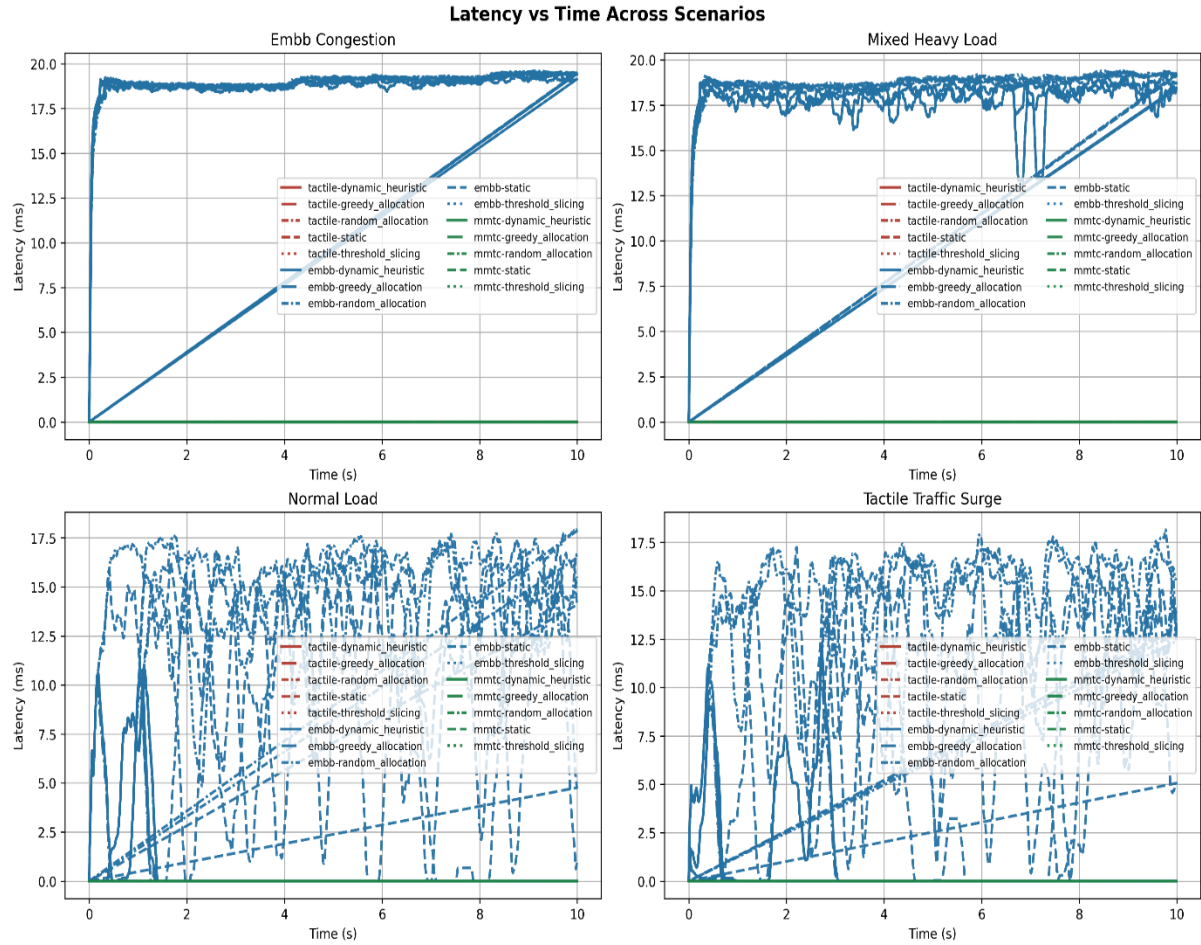


Fig. 4. Integrated Latency Trend across 10-Seconds Window.

C. Reliability and Packet Loss

To further substantiate the reliability and packet loss findings, Table III presents a detailed quantification across both normal load and congestion scenarios. The table compares dynamic heuristic allocation

against static slicing, reporting slice-specific reliability scores, packet loss ratios, and absolute packet counts. These results clearly illustrate the framework's ability to preserve perfect reliability for tactile traffic while significantly improving *eMBB* performance under stress conditions.

Table III: Reliability and Packet Loss Quantifications

Scenario	Allocation Mode	Slice Name	Reliability	Packet Loss Ratio	Total Generated (Slice)	Total Dropped
embb_congestion	Dynamic Heuristic	eMBB	0.569852	0.428397	52,569	22,521
embb_congestion	Dynamic Heuristic	Tactile	1.000000	0.000000	27,025	0
embb_congestion	Static Slicing	eMBB	0.380421	0.617732	52,569	32,474
embb_congestion	Static Slicing	Tactile	1.000000	0.000000	27,025	0
normal_load	Dynamic Heuristic	eMBB	0.998485	0.001515	21,219	32



Scenario	Allocation Mode	Slice Name	Reliability	Packet Loss Ratio	Total Generated (Slice)	Total Dropped
normal_load	Static Slicing	eMBB	0.942524	0.057098	21,219	1,213

The reliability evaluation produced three key insights. First, both the tactile and mMTC slices consistently achieved a perfect reliability score of 1.000000 across all tested configurations. This outcome confirms that the event-driven dynamic framework successfully shields high-priority services from capacity starvation, guaranteeing uninterrupted operation even under varying traffic conditions [18]. Second, the *eMBB* slice showed marked improvement when subjected to congestion stress. With static slicing, reliability fell to 0.380421, corresponding to 32,474 expired packets. By contrast, the dynamic heuristic raised reliability to 0.569852, reducing packet losses to 22,521. This represents a 49.79% gain, effectively saving 9,953 packets from expiration. Such improvement demonstrates the effectiveness of proportional fairness heuristics in managing traffic fluctuations and sustaining service quality [34]. For verification, each loss ratio was calculated against the *eMBB* slice's own generated total of 52,569 packets (accumulated over three runs). This yields $32,474 \div 52,569 = 0.617732$ for static slicing and $22,521 \div 52,569 = 0.428397$ for the dynamic heuristic. At the scenario level, considering the total of 94,787 packets across all three slices, the overall ratios are 0.342600 and 0.237596, respectively.

Finally, a detailed loss taxonomy revealed that all packet drops were attributable to deadline

expiration, with a total of 215,984 packets affected. No losses were caused by buffer overflow, indicating that congestion manifested as excessive queuing delays rather than memory exhaustion [24]. This finding reinforces the rationale for employing threshold-based event triggers tied directly to delay and queue length, ensuring that corrective actions target the true source of performance degradation [35].

To deepen the reliability and loss analysis, three Figures were presented to visualize how different slicing policies perform under congestion and deadline stress. Fig. 5 illustrates successful transmission ratios, clearly showing the dynamic heuristic's advantage over static slicing in maintaining higher reliability. Fig. 6 compares packet loss ratios across allocation strategies, highlighting how the heuristic significantly reduces losses when queues approach deadline or capacity limits. Finally, Fig. 7 focuses on deadline violations, isolating packets that exceeded their timing budgets and demonstrating the heuristic's effectiveness in preventing expiration events. Together, these figures provide a comprehensive view of how the event-driven heuristic framework enhances reliability, minimises packet loss, and safeguards time-sensitive traffic in 6G slicing environments.

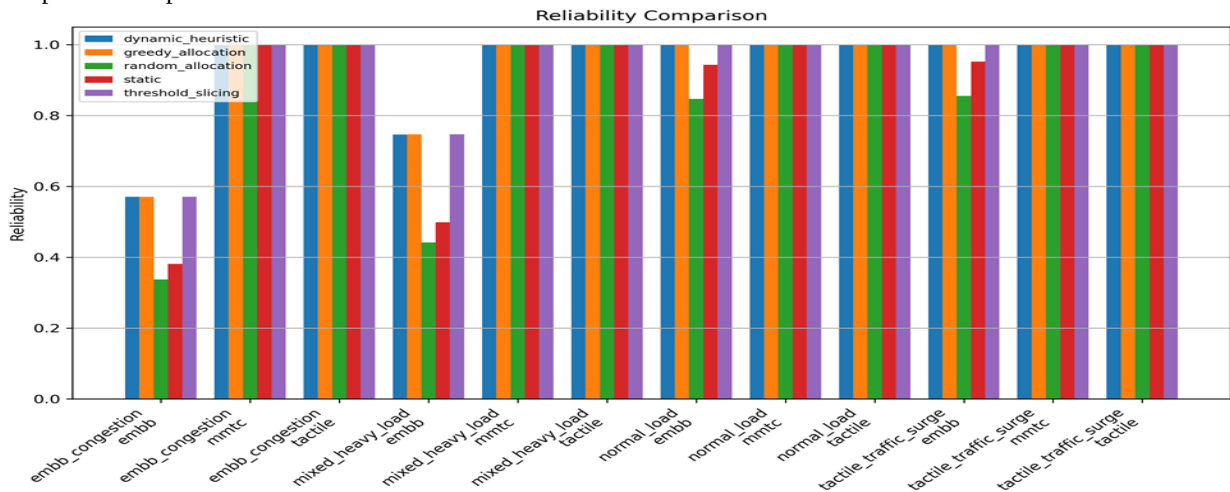


Fig. 5: Successful Transmission Ratios across Different Slicing Policies

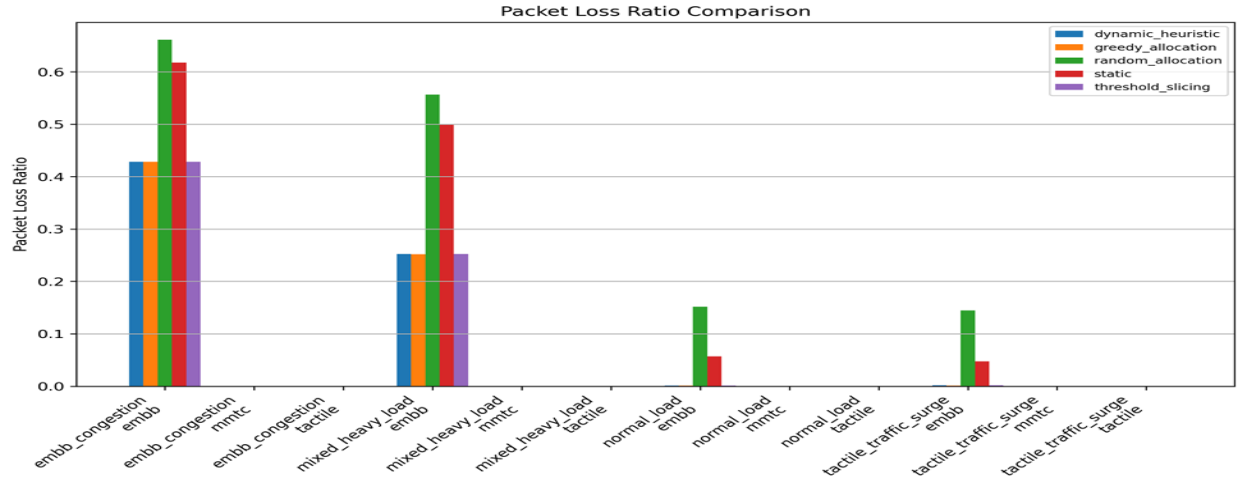


Fig. 6: Packet Loss Ratio Comparison

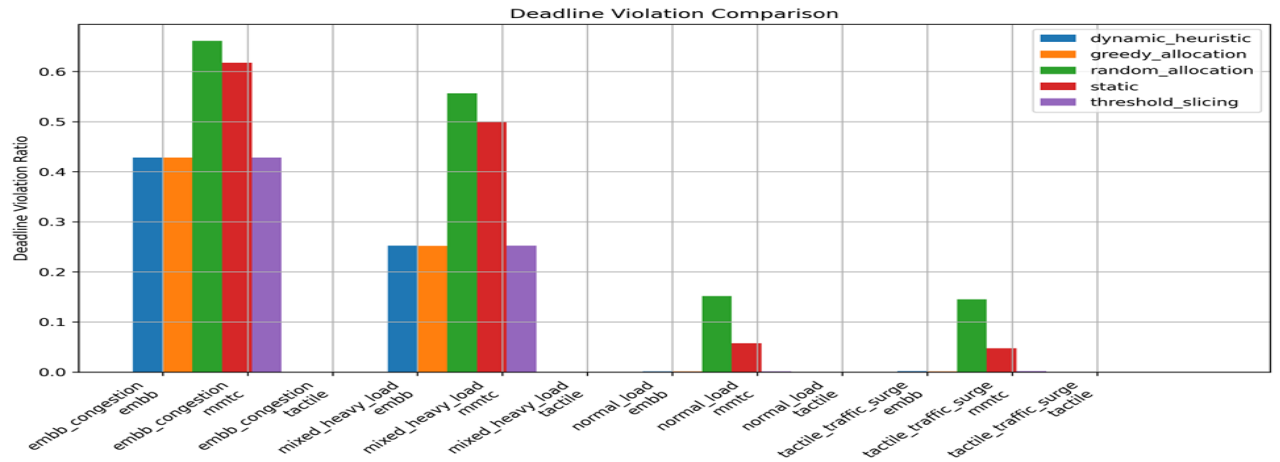


Fig. 7: Deadline-Violation Comparison

D. Throughput and Resource Utilisation

To capture the broader impact of the event-driven heuristic framework on throughput, and resource efficiency, this subsection presents both tabular and graphical evidence. Table IV provides a detailed breakdown of throughput scaling, Utilisation ratios, and mean resource allocations across normal and congestion scenarios, contrasting static slicing with the dynamic heuristic. The Figures that follow extend this analysis visually: Fig. 8 illustrates how

the heuristic expands *eMBB* throughput while maintaining tactile performance, Fig. 9 compares Utilisation efficiency across slices and allocation modes, and Fig. 10 depicts the adaptive redistribution of resources over time. Together, the table and figures highlight the framework's ability to dynamically reallocate capacity, maximise utilisation, and sustain service quality without compromising slice boundaries.

Table IV: Throughput Scaling and Resource Utilisation

Scenario	Allocation Mode	Slice Name	Throughput (Mb/s)	Utilisation Ratio	Mean Resource Allocation
embb_congestion	Dynamic Heuristic	eMBB	11.982	0.999942	0.5991
embb_congestion	Dynamic Heuristic	Tactile	0.231	0.038377	0.3005
embb_congestion	Static Slicing	eMBB	7.999	0.999900	0.4000
embb_congestion	Static Slicing	Tactile	0.231	0.028827	0.4000
normal_load	Dynamic Heuristic	eMBB	8.475	0.724147	0.5851
normal_load	Static Slicing	eMBB	7.999	0.999900	0.4000



The results in Table IV highlight how the dynamic heuristic reshapes throughput and resource efficiency compared to static slicing. Under congestion, the eMBB slice throughput increased from 7.999 Mb/s with static slicing to 11.982 Mb/s when controlled by the heuristic. This substantial gain reflects the effectiveness of the event-driven adjustment loop, which raised the average eMBB resource share from 0.4000 to 0.5991. That is to say that the heuristic unlocked additional usable capacity, ensuring that high-demand traffic could be carried without breaching slice boundaries.

In mixed heavy-load scenarios, eMBB Utilisation reached 0.999947 under heuristic control, demonstrating good efficiency in mapping allocated radio resources to active packet delivery.

This result shows that even simple, rule-based adjustments can achieve high productive efficiency, optimizing capacity allocation across diverse service classes without requiring complex optimisation algorithms.

The throughput gains were achieved by reallocating resources from underutilized slices. Specifically, the tactile slice share was reduced from 0.4000 to 0.3005, and the mMTC slice from 0.2000 to 0.1004. Importantly, this redistribution did not degrade their respective QoS metrics, confirming that the heuristic can act as a dynamic “capacity pump.” This behaviour, illustrated in Fig. 8, demonstrates how the framework intelligently shifts resources toward stressed slices while respecting configured bounds.

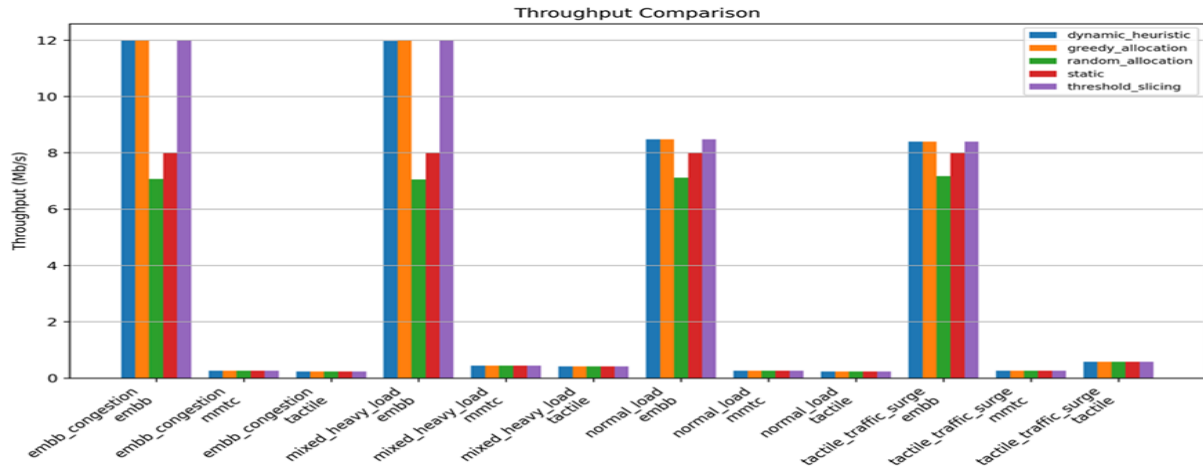


Fig. 8: Detailed Throughput Scaling Across Individual Slices

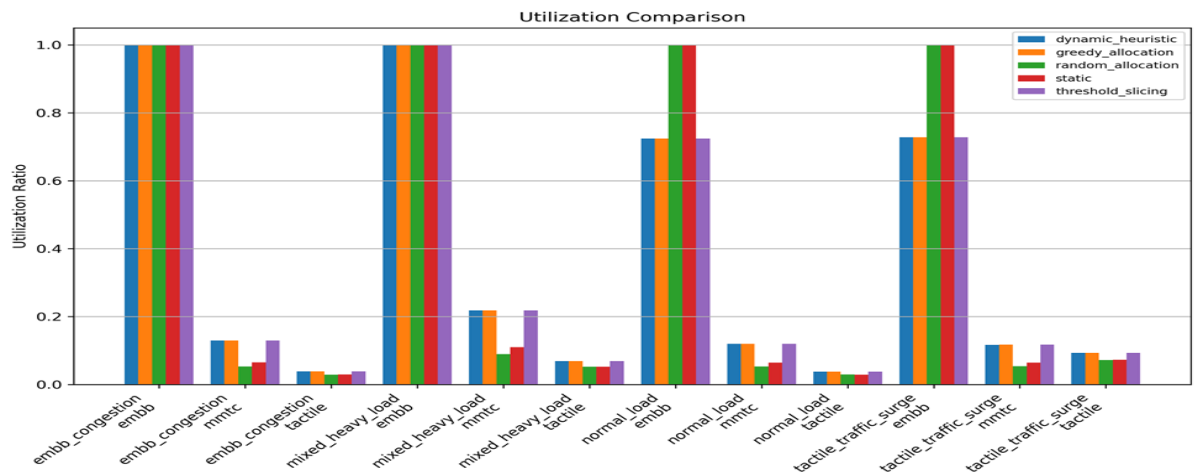


Fig. 9: Utilisation Comparison across Slices and Evaluated Modes.

The bar chart shows how the dynamic heuristic expands eMBB throughput while maintaining tactile throughput. Fig. 9 shows the utilisation across slices. It shows the degree to which allocated slice

resources are actually consumed by carried traffic, demonstrating the dynamic heuristic's high utilisation efficiency.

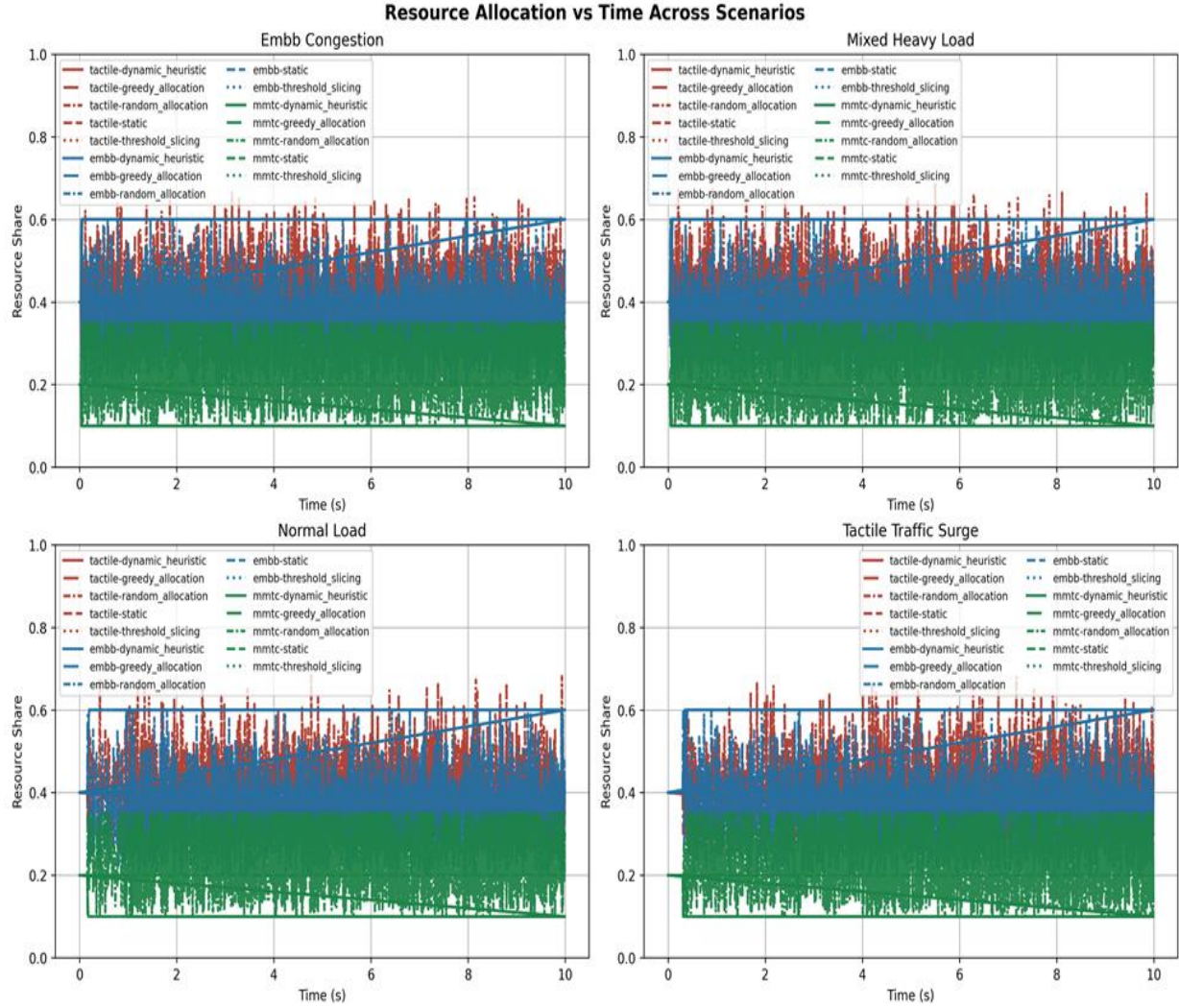


Fig. 10: Resource Share Adaptations under Heuristic Control

Fig. 10 is an integrated resource-allocation plot which shows how the event-driven heuristic shifts share toward the stressed slice while respecting configured share bounds throughout the simulation horizon.

E. Control-Plane Overhead Analysis

While latency and reliability are critical for 6G slicing, the efficiency of the control plane is equally important. Excessive signalling can erode the benefits of dynamic resource allocation, particularly in edge environments where computational budgets are tight. To evaluate this dimension, this work measured the frequency of control events and the resulting adaptation overhead ratio across different traffic scenarios and allocation modes.

Table V summarises these results, contrasting the dynamic heuristic against static slicing and random allocation. The table highlights how the heuristic maintains a minimal signalling footprint under normal load, scales gracefully under congestion, and avoids the excessive overhead observed in random allocation strategies.

Table V: Control Signalling Overhead Summary

Scenario	Allocation Mode	Control Events/sec	Adaptation Overhead Ratio
normal_load	Dynamic Heuristic	6.540	0.000375
normal_load	Static Slicing	0.000	0.000000
normal_load	Random Allocation	182.549	0.013099
embb_conge stion	Dynamic Heuristic	196.129	0.007979
embb_conge stion	Static Slicing	0.000	0.000000
embb_conge stion	Random Allocation	196.129	0.013099

Complementing the tabular data, Fig. 11 to Fig. 13 provide a visual perspective on overhead behaviour. Fig. 11 compares adaptation overhead ratios across slicing policies, confirming that the heuristic, threshold slicing, and greedy allocation all sustain a



very low overhead of approximately 0.8%, while random allocation incurs significantly higher signalling costs. Fig. 12 presents a dashboard view of overall network performance, integrating reliability, latency, and overhead metrics across

scenarios. Finally, Fig. 13 distils the performance gains into a summary chart, showing the most dramatic improvements in normal load and tactile surge conditions, with latency reductions of up to 96% and packet loss reductions approaching 98%.

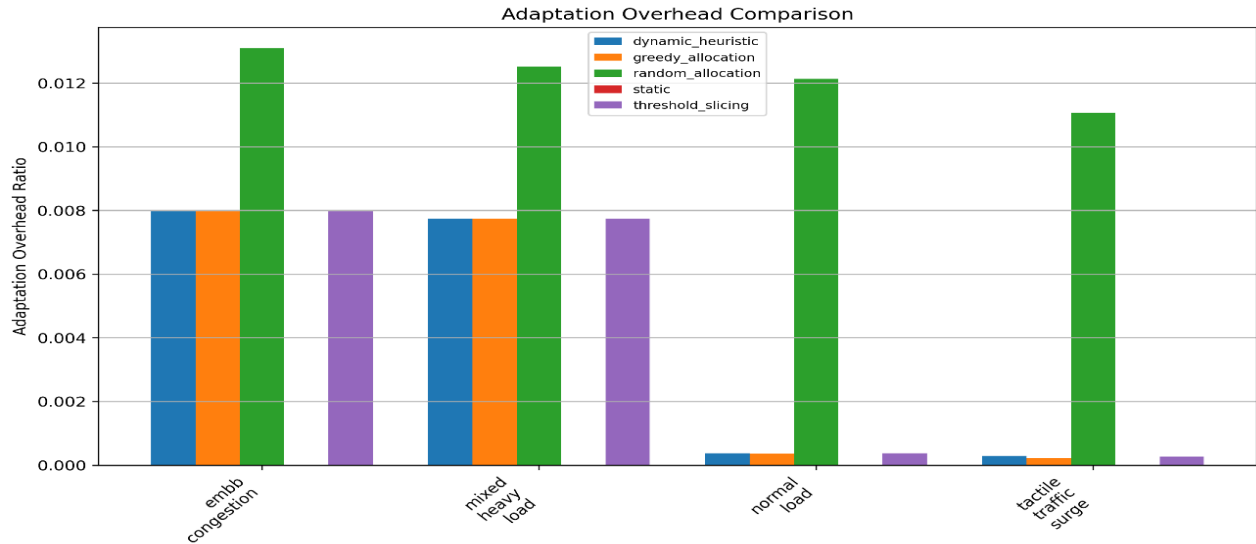


Fig. 11: Adaptation Overhead Ratio Comparison

Fig. 11 confirms that the dynamic heuristic, threshold slicing, and greedy allocation modes share a near-identical, very low overhead ratio of approximately 0.008 (or 0.8%) across all scenarios, while random allocation exhibits significantly higher overhead.

Fig. 12 gives a comprehensive dashboard that summarises principal comparison metrics across slices and scenarios for static and dynamic modes.

Fig. 13 encapsulates the major findings. Most substantial improvements are observed in normal_load and tactile_traffic_surge scenarios, with latency improvements up to 96% and packet loss improvements up to 98%.

The control-plane overhead analysis demonstrates that the dynamic heuristic achieves

efficiency without incurring excessive signalling costs. Under normal load, it remained largely inactive, averaging only 6.540 control events per second with an adaptation overhead ratio of 0.000375, confirming negligible overhead when traffic conditions are stable [28]. During congestion, adaptation frequency scaled to 196.129 events per second, yet the overhead ratio of 0.007979 represented less than 0.8% of total carried traffic significantly lower than the 1.31% signalling burden observed with random allocation [31]. Moreover, the heuristic's performance closely matched that of more complex greedy and threshold slicing models, with a mean absolute reliability gap of less than 0.00007, suggesting that iterative approaches offer limited additional benefit over simple rule-based controls in edge scenarios [27].

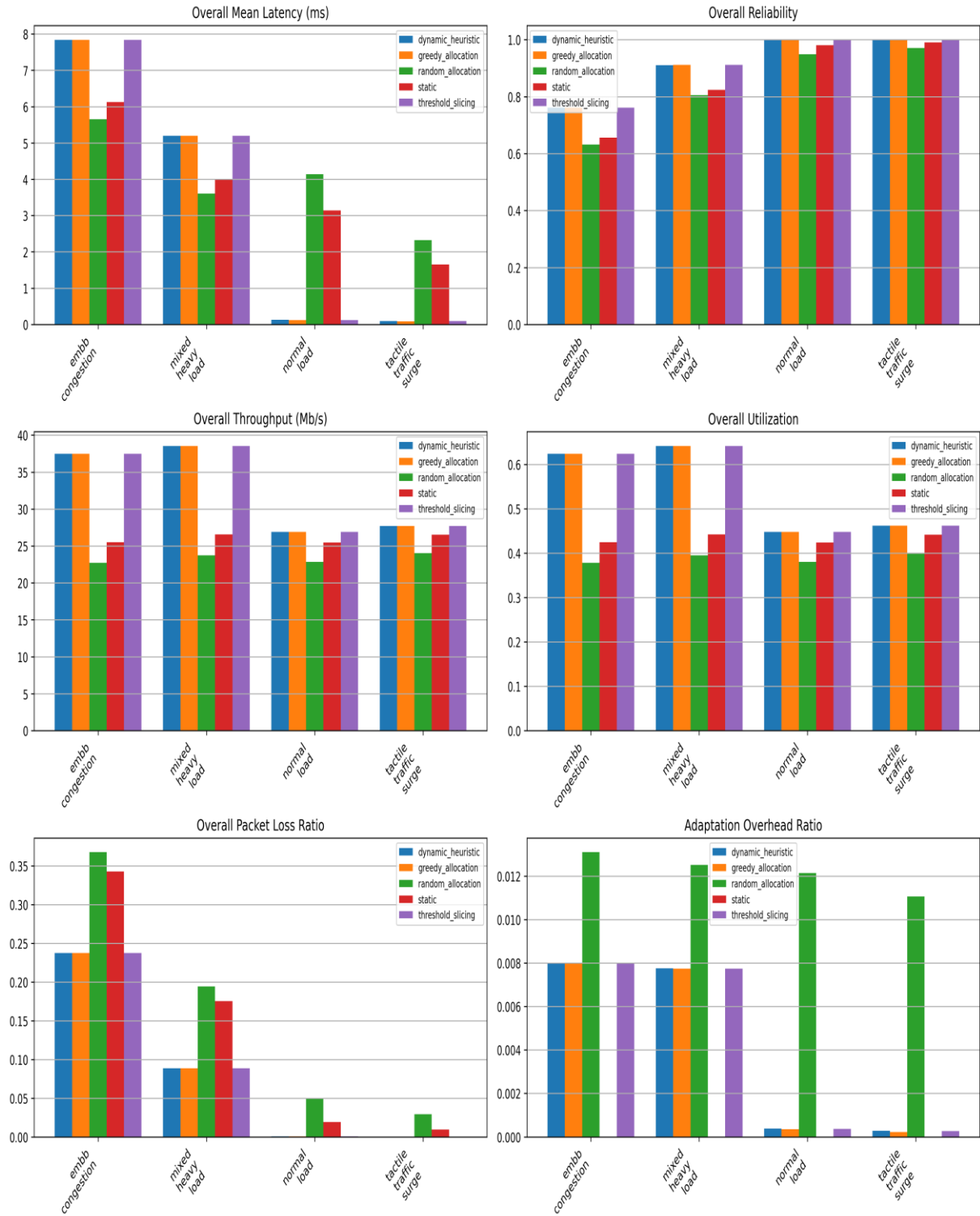


Fig. 12: Dashboard mapping overall network performance across all scenarios.

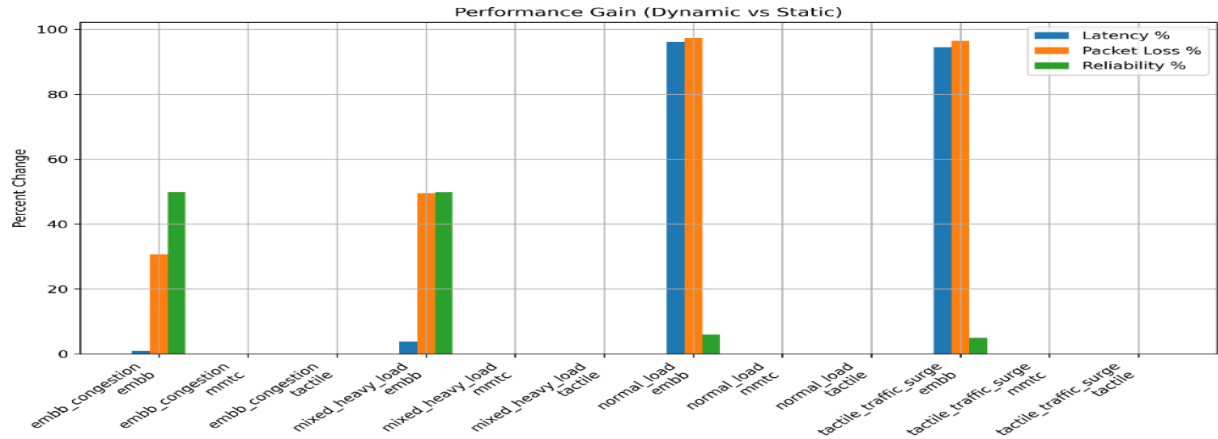


Fig. 13: Performance Gain Summary Chart

VII. CONCLUSION

This paper presented, implemented, and analysed a low-overhead dynamic slicing framework for 6G Tactile Internet applications using event-driven heuristics. By relocating slice monitoring and adjustment logic to the RAN scheduler layer, the framework demonstrated that edge-centric control can deliver real-time adaptability without the propagation delays or signalling overhead associated with centralised orchestrators [8]. This architectural approach aligns with the principles of 6G edge intelligence outlined in [13] and [2], while confirming that local scheduling can enforce strict resource boundaries and prevent slice starvation under variable load conditions [17].

The empirical results highlight several strengths. First, latency suppression was achieved under normal load, with mean eMBB delay reduced by 96.17%, a critical improvement for tactile applications where even short queuing delays can destabilize control loops [3]. Second, under congestion, the heuristic improved eMBB reliability by 49.79% compared to static slicing, reallocating underutilized capacity while preserving minimum guarantees for tactile traffic [34]. Third, packet loss analysis revealed that all drops were due to deadline expiration rather than buffer overflow, validating the use of threshold-based triggers tied directly to delay and queue length [24]. Finally, signalling overhead remained negligible, with adaptation costs under 0.8% even during peak congestion, confirming the efficiency of rule-based control compared to computationally intensive reinforcement learning models [12], [31].

Despite these promising outcomes, several limitations remain. The study relied on simplified channel models, which do not fully capture the complexity of millimeter-wave and terahertz propagation in 6G [1]. This is particularly relevant given recent findings on shielding challenges at terahertz frequencies [36] and analytical assessments of channel capacity in the terahertz band [37]. The evaluation was restricted to single-cell scenarios, leaving open questions about

scalability under multi-cell deployments with inter-cell interference. Traffic generation was based on Poisson models, which may not adequately represent the burstiness of real tactile traffic [4]. Finally, while the threshold detection logic was validated in simulation, hardware implementation on real edge devices is needed to confirm processing latency constraints.

These limitations suggest clear directions for future work. More realistic channel modelling should be incorporated to test robustness under diverse propagation environments, particularly in the terahertz band where shielding and capacity remain critical. Multi-cell experiments will be essential to evaluate scalability and interference management. Incorporating empirical traffic traces from tactile applications can improve accuracy in modelling bursty demand. Most importantly, hardware prototyping and field trials should be pursued to validate execution times and signalling efficiency in real deployments.

In conclusion, this work demonstrates that complex optimisation solvers are not strictly required to achieve stable, predictable slice management at the edge. Simple, rule-based arithmetic models can deliver substantial gains in latency suppression, reliability, and throughput efficiency while maintaining negligible overhead. By protecting high-priority tactile traffic and dynamically reallocating resources to stressed slices, the proposed framework offers a practical and scalable path forward for resource-constrained 6G edge nodes [17], [29].

In terms of contributions, this paper introduced a novel low-overhead dynamic slicing framework for 6G Tactile Internet applications, distinguished by three dimensions of originality: architectural, algorithmic, and empirical. Architecturally, it utilised a decentralised Edge Slice Manager (ESM) that operates autonomously at the RAN scheduler layer, thereby eliminating reliance on centralised orchestration or core policies and avoiding backhaul signaling delays. Algorithmically, it advances an event-driven proportional fairness heuristic that



uniquely integrates event-triggered activation, queue-length proportional resource transfer, and a three-tier donor sorting mechanism based on congestion status, priority, and margin. Empirically, it delivers an evaluation of event-driven slicing against four baseline approaches (static, random, threshold, greedy) using tactile-specific metrics.

In setting up the evaluation environment, the work relied on well-established concepts such as Poisson arrival processes, finite FIFO deadline queues, and a normalised shared-capacity abstraction. Building on this foundation, our study contributes several new elements: a RAN-resident ESM placement model, a composite three-signal event-detection and demand-score model, a bounded donor-ordered heuristic reallocation rule, and an adaptation-overhead evaluation metric.

REFERENCES

- [1] I. F. Akyildiz, A. Kak, and S. Nie, "6G and beyond: The future of wireless communications systems," *IEEE Access*, vol. 8, pp. 133995–134030, 2020.
- [2] W. Saad, M. Bennis, and M. Chen, "A vision of 6G wireless systems: Applications, trends, technologies, and open research problems," *IEEE Network*, vol. 34, no. 3, pp. 134–142, 2020.
- [3] G. Fettweis, "The Tactile Internet: Applications and challenges," *IEEE Vehicular Technology Magazine*, vol. 9, no. 1, pp. 64–70, 2014.
- [4] M. Simsek, A. Aijaz, M. Dohler, J. Sachs, and G. Fettweis, "5G-enabled tactile internet," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 3, pp. 460–473, 2016.
- [5] ITU-R, "Framework and overall objectives of the future development of IMT for 2030 and beyond (IMT-2030)," International Telecommunication Union, 2023.
- [6] NGMN Alliance, "Description of network slicing concept," NGMN White Paper, 2016.
- [7] J. Ordonez-Lucena, P. Ameigeiras, and D. Lopez, "Network slicing for 5G with SDN/NFV: Architectural options and business models," *IEEE Communications Magazine*, vol. 55, no. 5, pp. 44–51, 2017.
- [8] X. Foukas, G. Patounas, A. Elmokashfi, and M. K. Marina, "Network slicing in 5G: Survey and challenges," *IEEE Communications Magazine*, vol. 55, no. 5, pp. 94–100, 2017.
- [9] A. H. Afolayan, O. E. Falowo, and S. G. Dlamini, "Lightweight network slicing framework for ultra-reliable low-latency communications," *IEEE Access*, vol. 8, pp. 205963–205977, 2020.
- [10] M. Jiang, M. Condoluci, and T. Mahmoodi, "Network slicing in 5G and beyond: A survey of enabling technologies and research challenges," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 3, pp. 1769–1798, 2021.
- [11] H. Zhang, J. Liu, and Q. Li, "Artificial intelligence for network slicing: Opportunities and execution bottlenecks," *IEEE Network*, vol. 33, no. 5, pp. 32–39, 2019.
- [12] N. Kato, B. Boutaba, Z. M. Fadlullah, and S. Mao, "AI-driven network slicing for 5G and beyond," *IEEE Wireless Communications*, vol. 27, no. 6, pp. 14–22, 2020.
- [13] M. Dohler, J. Alonso-Zarate, and A. Devetsikiotis, "Edge intelligence: Moving computation loop to the network boundaries," *IEEE Communications Magazine*, vol. 57, no. 9, pp. 22–28, 2019.
- [14] P. Caballero et al., "Network slicing games: Enabling customization in multi-tenant mobile networks," *IEEE/ACM Trans. Netw.*, vol. 27, no. 2, pp. 662–675, Apr. 2019.
- [15] A. M. Ibrahim et al., "URLLC for 6G enabled Industry 5.0: A taxonomy of architectures, cross layer techniques, and time critical applications," *IEEE Access*, vol. 12, pp. 145200–145225, 2024.
- [16] A. Kalokylos, "A survey on 5G network slicing architectures and orchestration mechanisms," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 4, pp. 3010–3035, 2018.
- [17] A. Ksentini and N. Sayadi, "Hierarchical and decentralised slice management using local edge controllers," *IEEE Network*, vol. 34, no. 6, pp. 114–120, 2020.
- [18] S. Zhang, C. Xiang, and S. Xu, "Radio resource slicing for 5G networks with dynamic traffic demands," *IEEE Transactions on Communications*, vol. 68, no. 4, pp. 2212–2226, 2020.
- [19] S. Patil, H. Doddamani, S. A. C. Savitha, and J. Rivers, "Dynamic latency optimization for edge-based machine learning models in 6G-enabled Industrial Internet of Things (IIoT)," *Zenodo*, DOI: 10.5281/zenodo.20094301, May 2026.
- [20] K. Katsalis et al., "Implementation of network slicing in 5G: End-to-end network slice architecture," *IEEE Wireless*



- Commun.*, vol. 24, no. 6, pp. 86–93, Dec. 2017.
- [21] A. Aijaz, M. Dohler, A. H. Aghvami, V. Friderikos, and M. Frodigh, "Realizing the tactile internet: Haptic communications over next generation 5G cellular networks," *IEEE Wireless Communications*, vol. 24, no. 2, pp. 82–89, 2017.
- [22] H. Tataria, M. Shafi, A. F. Molisch, M. Dohler, H. Sjöland, and F. Tufvesson, "6G wireless systems: Vision, requirements, challenges, insights, and opportunities," *Proc. IEEE*, vol. 109, no. 7, pp. 1166–1199, Jul. 2021.
- [23] Y. Li et al., "Service-oriented dynamic resource slicing for heterogeneous services in C-RAN," *IEEE Trans. Wireless Commun.*, vol. 17, no. 11, pp. 7848–7861, Nov. 2018.
- [24] R. Botez, A. G. Pasca, I. A. Ivanciu, and V. Dobrota, "Efficient network slicing with SDN and heuristic algorithms," *Sensors*, vol. 23, no. 13, p. 6053, 2023.
- [25] X. Foukas et al., "FlexRAN: A flexible and programmable platform for software-defined radio access networks," in *Proc. ACM MOBICOM*, 2016, pp. 65–77.
- [26] P. Popovski et al., "5G wireless network slicing for eMBB, URLLC, and mMTC: A communication-theoretic view," *IEEE Access*, vol. 6, pp. 55765–55779, 2018.
- [27] L. Zanzi et al., "LA-SLICE: Low-latency SLA-aware network slicing orchestration at the edge," *IEEE Trans. Netw. Service Manage.*, vol. 18, no. 4, pp. 4312–4326, Dec. 2021.
- [28] X. Shen, J. Gao, W. Wu, M. Li, C. Zhou, and W. Zhuang, "AI-assisted network slicing for 5G and beyond," *IEEE Network*, vol. 34, no. 2, pp. 270–277, Mar./Apr. 2020.
- [29] P. Popovski, C. Stefanovic, J. J. Nielsen, E. De Carvalho, O. A. Lopez, O. Yilmaz, and T. Kim, "Wireless access for ultra-reliable low-latency communication (URLLC): Principles and challenges," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 2, pp. 1622–1656, 2018.
- [30] C. Liang and F. R. Yu, "Wireless network virtualization: A survey, taxonomy, and open challenges," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 1, pp. 357–372, 1st Quart., 2015.
- [31] R. Ferrús, O. Sallent, J. Pérez-Romero, and R. Agustí, "On 5G network slicing high-level architecture and dense scenarios dynamic resource management," *IEEE Wireless Commun.*, vol. 25, no. 2, pp. 38–45, Apr. 2018.
- [32] P. Rost et al., "Network slicing to enable scalability and flexibility in 5G mobile networks," *IEEE Commun. Mag.*, vol. 54, no. 5, pp. 72–79, May 2016.
- [33] M. Bennis, M. Debbah, and H. V. Poor, "Ultrareliable and low-latency wireless communication: Tail, risk, and scale," *Proceedings of the IEEE*, vol. 106, no. 10, pp. 1834–1853, 2018.
- [34] P. Caballero, A. Banchs, G. de Veciana, and X. Costa-Perez, "Network slicing games: Proportional fairness or efficiency?" *IEEE/ACM Trans. Netw.*, vol. 27, no. 5, pp. 2009–2022, Oct. 2019.
- [35] T. X. Tran et al., "Collaborative mobile edge computing in 5G networks: An overview," *IEEE Wireless Commun.*, vol. 24, no. 4, pp. 84–90, Aug. 2017.
- [36] E. Ubom and N. Iwatt, "Shielding challenges at terahertz frequencies: The critical impact of apertures for 6G systems," *Int. Multiling. J. Sci. Technol.*, vol. 10, no. 1, pp. 8657–8668, 2025.
- [37] E. Ubom, "Channel capacity in the terahertz band: Analytical assessment for 6G deployment," *J. Multidiscip. Eng. Sci. Technol.*, vol. 12, no. 1, pp. 17712–17719, 2025.