International Journal of Computer Trends and Technology
ISSN: 2231–2803 / https://doi.org/10.14445/22312803/IJCTT-V74I4P105

Volume 74 Issue 4, 52-72, April 2026

Original Article

SCONE-AEGIS: Uncertainty-Aware AI-Driven Edge Compute Steering for Mobile Edge Applications

Venkata Rama Uday Kiran Bokam[1], Sachin Vasanthkumar[2], Kameswaran Arunachalam[3]

[1]*Nokia of America Cooperation, Bellevue, Washington, USA.*
[2]*Cisco Systems Inc., Bellevue, Washington, USA.*
[3]*T-Mobile, Bellevue, Washington, USA.*

[1]*Corresponding Author : uday_kiran.bokam@nokia.com*

Received: 28 February 2026 Revised: 30 March 2026 Accepted: 19 April 2026 Published: 30 April 2026

Abstract *- Mobile Edge Computing (MEC) has become increasingly critical for latency-sensitive applications, including Augmented/Extended Reality (AR/XR), Cloud Gaming, Real-Time Video Analytics, and Interactive Enterprise Services. Existing edge steering mechanisms remain largely reactive by relying on static policies, nearest-edge selection, or compute-only information that usually fail under user mobility, fluctuating radio conditions, dynamic user-plane paths, and edge resource contention rather than being more proactive. This paper presents SCONE-AEGIS framework that extends the Standard Communication with Network Elements (SCONE) paradigm beyond throughput advisories to support joint network-compute steering of MEC applications. SCONE-AEGIS introduces an Edge Steering Advice (ESA) that communicates recommendations that can be consumed by the applications, which have been derived from a combination of RAN, UPF, and MEC telemetry. The framework is a combination of a two-stage AI/ML engine, the first being a Spatio-Temporal Graph Predictor that is uncertainty-aware and models the evolving relationships among radio access network nodes, user-plane functions, edge sites, and mobile users, and the second stage is a Safe Contextual Bandit Steering Policy (SCBSP) that selects execution sites subject to SLA constraints, migration hysteresis, and prediction confidence. The proposed framework provides a standards-compatible, privacy-preserving path for exposing joint network-compute intelligence to applications without breaking transport encryption.*

Keywords *- SCONE, MEC, Edge Computing, Service Steering, Mobile Networks, AI/Ml, Graph Neural Networks, QoE, Mobility-Aware Orchestration, 5G.*

1. Introduction

Mobile Edge Computing (MEC) has emerged as one of the use cases in the 5G and beyond, which enables ultra-low latency services to the user by deploying user-plane functions at the network edge, which is as close to the user as possible.

However, the dynamic nature of wireless networks, which are characterized by user mobility, fluctuating conditions, and time-varying computing availability, poses fundamental challenges for making edge-compute steering decisions.

1.1. The Standard Communication of Network Elements (SCONE) Framework

The foundation of the proposed work builds upon the Standard Communication of Network Elements (SCONE) concept introduced in [1]. SCONE represents a huge shift in how mobile networks interact with applications and User Elements (UEs) by providing advised bitrate information flow between the network layer and application layer using the QUIC Protocol.

1.1.1. Core SCONE Principles [1]

Network-to-Application Information Exposure

SCONE advises bitrate in real-time network state information to applications, based on:

- Radio Access Network (RAN) conditions (signal strength, cell load, handover predictions)
- Core network telemetry (latency, jitter, throughput metrics)
- Edge compute resource availability (CPU, memory, storage utilization)
- Predicted network conditions based on mobility patterns and historical data

This exposure will enable applications to make network-aware decisions, but not treat the network like a connecting pipe between two points.

Application-to-Network Intent Communication

Applications need to communicate their requirements and preferences to the network, such as:

- Service Level Agreement (SLA) requirements like latency thresholds, minimum throughput.

- Traffic characteristics like burst patterns and criticality levels.
- Tolerance to the level of service interruptions.
- Quality-of-service (QoS) is preferred by the network.

This bidirectional exchange creates a collaborative environment where both networks and applications adapt to each other's constraints.

Dynamic Service Steering

SCONE enables the network to provide advisory recommendations for edge site selection based on:

- Current and predicted network conditions
- Application-specific requirements
- Compute resource availability across edge sites
- Migration costs and service continuity considerations

A critical point is that SCONE maintains application sovereignty, as in the applications receive recommendations but retain the authority for the final decision [1]. This design follows the application-level business logic while using network intelligence.

Standardized API Interface

SCONE defines standardized northbound APIs for network information exposure to ensure:

- Interoperability across multi-vendor environments
- Backward compatibility with current application frameworks
- Security and privacy through authentication and data sanity.
- Scalability through efficient information encryption.

1.2. Motivation for SCONE-AEGIS

While SCONE [1] has established the architectural framework for collaboration between network and application, it has left open critical questions about the intelligence mechanism that generates edge steering advice:
Limitation 1**:** Spatio-Temporal Prediction on how the system predicts future network conditions due to:

- Spatial correlations like neighboring cells influence each other.
- Temporal dynamics, mobility patterns, traffic periodicity)
- Heterogeneous relationships (UE-cell, cell-edge, inter-edge connections)

Existing approaches treat predictions independently, failing to exploit relational structure [2][3].
Limitation 2**:** Uncertainty in Quantification Network predictions is due to:

- Wireless channel randomness (fading, interference)
- Unpredictable user mobility patterns
- Variable application traffic loads

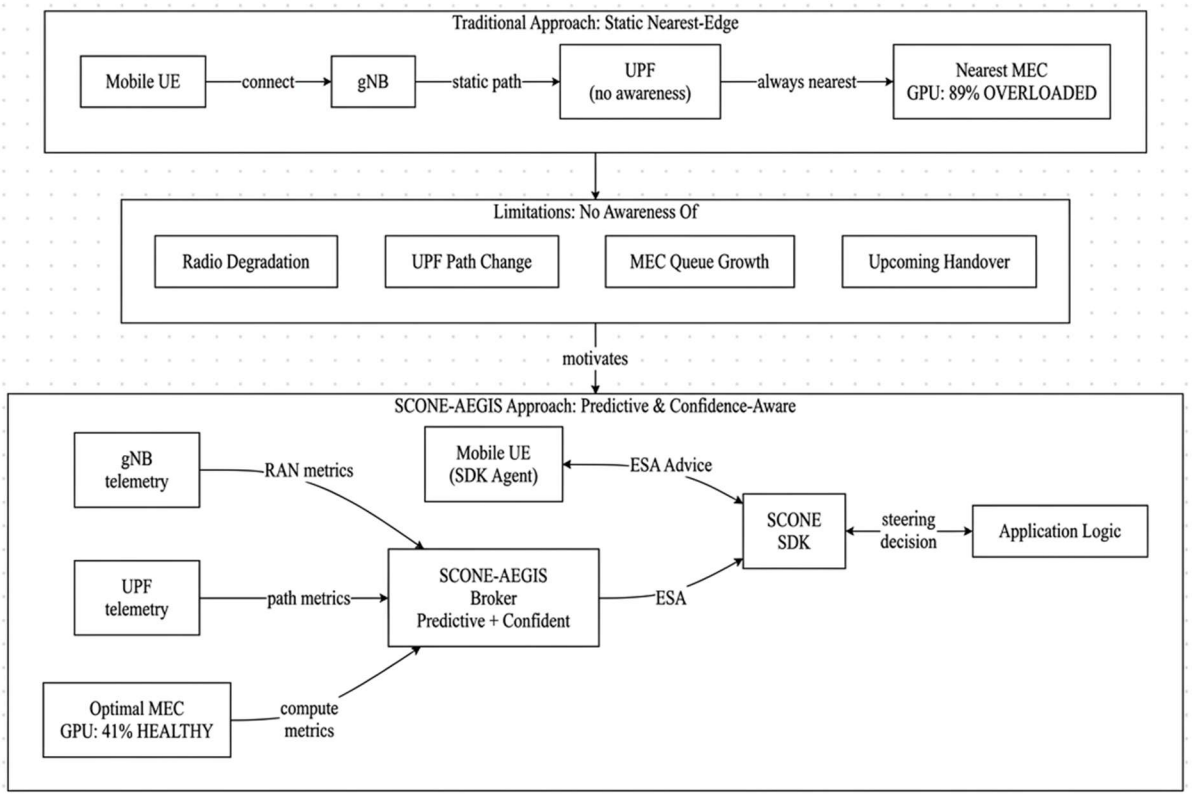

Fig. 1 Illustration of how the MEC steering gap is addressed by SCONE-AEGIS. Traditional approaches use static nearest-edge selection without cross-domain awareness. SCONE-AEGIS provides predictive, confidence-aware steering advice.

However, current systems provide point estimates without confidence bounds, forcing applications to make decisions without knowing how reliable the prediction is [4].
Limitation 3: Safe Exploration — Edge steering involves an exploration-exploitation trade-off:

- Exploitation: Choose known good-edge sites based on current knowledge
- Exploration: Try alternative sites to discover better options

However, exploration can violate SLAs. This raises the question of how optimal policies can be learned while maintaining safety guarantees. [5]

Limitation 4: Confidence-Aware Advisory, even with predictions and recommendations, a mechanism is needed to establish when applications should trust the advice. A confidence mechanism is needed to filter recommendations that are unreliable.

This paper presents SCONE-AEGIS, an AI-enabled framework that allows network elements and edge controllers to expose Edge Steering Advice to applications or application SDKs. Unlike traditional reactive steering, SCONE-AEGIS is a predictive and confidence-aware model that combines the radio domain, user-plane path, and edge compute domain as part of its modelling.

Unlike Huang et al. [18], which apply deep reinforcement learning to MEC offloading using point-estimate predictions and unconstrained exploration, SCONE-AEGIS combines a heteroscedastic spatio-temporal graph predictor over the joint RAN, UPF, and MEC topology with a safety-constrained contextual-bandit policy, so steering decisions retain prediction uncertainty inside the action-selection step while remaining feasible under explicit SLA and migration-cost constraints.

This contribution is further distinguished from baseline SCONE [1][9], whose advisories convey throughput and path-level guidance to applications, by extending the same network-to-application advisory paradigm into joint network-compute MEC steering through an application-consumable Edge Steering Advice (ESA) that carries cross-domain mean and variance predictions, a per-advisory confidence score, and an uncertainty-derived advisory validity lifetime that contracts when prediction reliability drops.

Section 8.2 enumerates the corresponding baselines, including a DRL configuration aligned with [18]; Section 9 reports the resulting quantitative comparison across pedestrian, vehicular, and hotspot mobility scenarios over XR, cloud gaming, and video analytics workloads, and Section 9.6 discusses how the differentiation from baseline SCONE [1][9] translates to practical MEC deployment under the 500 ms control-loop budget.

1.3. Contributions

This paper makes the following specific contributions:

- *SCONE-AEGIS Architecture*: SCONE-based framework for joint network-compute MEC steering, combining RAN, UPF, and MEC telemetry into steering advice towards the application.
- *Edge Steering Advice (ESA):* An application facing advisory abstraction for MEC site ranking, migration recommendation, and confidence-aware validity interval management
- *USTGP:* An uncertainty-aware spatio-temporal graph predictor for forecasting latency, jitter, migration cost, and service interruption risk across the RAN, UPF, and MEC topologies.
- *SCBSP*: A contextual bandit steering policy that makes SLA-aware, migration-safe decisions under prediction uncertainty
- *Confidence-Steering Coupling:* Formal constraints that tie prediction uncertainty to migration safety thresholds and advisory lifetime
- *Empirical Validation:* Simulation results demonstrate statistically improved performance gains over eight different baseline strategies across three mobility scenarios and application classes.

2. Background and Motivation

2.1. Nearest Edge Selection Limitation

Nearest-edge selection assumes that physical or topological proximity reliably proxies application latency. This assumption is not strong enough in mobile edge environments, as the end-to-end response time depends on:

1. Radio access quality and scheduler contention
2. Current user-plane anchored and forwarding path
3. Edge site CPU/GPU load and queue depth
4. MEC platform queueing delay
5. Service-instance warmness and cache locality
6. User mobility trajectory and handover probability
7. Transport path jitters and transient congestion

A user who is geographically near to one edge site may receive better performance from another site with a shorter user-plane path or lower compute queueing delay [9]. This dependency between topology and performance is precisely what the SCONE-AEGIS framework solves.

2.2. Static MEC Policies Limitation

Static policies, which are placed based on region placement and edge selection, which are operator preferred, are not adaptive to:

1. Sudden spikes in RAN load due to crowd mobility events.
2. Transient UPF congestion during peak traffic hours.
3. Edge resource hotspots are caused by popular content bursts.
4. Partial failures or maintenance window activities are happening in the mobile edge sites.

5. Application's sensitivity to latency variations
6. Mobility-induced path topology changes

This is a concern for the real-time applications whose quality decreases very quickly when latency variance increases beyond certain thresholds [10][11].

2.3. Need for Application-Facing Advisory Signaling

The application endpoint often has the final responsibility for selecting a service endpoint, starting offload, or migrating a session. However, the application will not have direct access to:

1. Network-side mobility prediction and handover risk
2. UPF queue occupancy and path utilization
3. Per-edge execution load and warm instance count
4. Expected migration interruption duration
5. Near-future path stability confidence

SCONE bridges this gap by exposing compact, policy-safe, actionable advisory signals to the endpoint without the need for payload visibility or breaking transport encryption [8]

2.4. Motivating Scenario

Consider a mobile XR client connected through serving gNB-A and UPF-1 to Edge Site A. In the scenario where edge Site A is currently topologically nearer, but the GPU queue depth is growing toward saturation. In parallel, radio analytics indicate the user is moving toward a neighboring cell whose optimal user-plane path maps to Edge Site B. In this case, the static policy continues using Edge Site A until latency visibly degrades, and the reactive strategy migrates only after the application experiences stutter. *SCONE-AEGIS predicts the degradation window and advises the application to prepare to migrate or migrate to Edge Site B before user-visible impact occurs*, entirely within the transport encryption envelope.

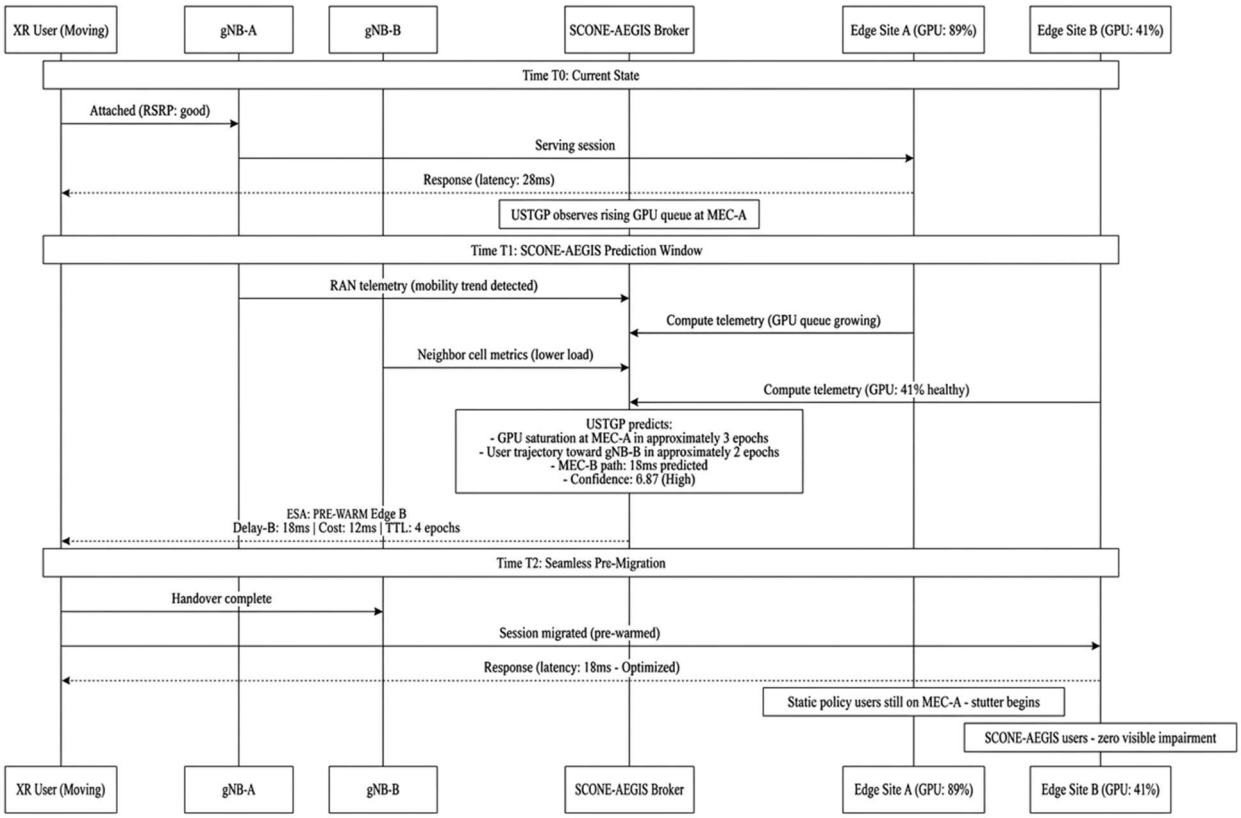

Fig. 2 Scenario showing an XR user experiencing GPU saturation at Edge Site A while moving toward gNB-B coverage. SCONE-AEGIS predicts the degradation window and advises pre-migration to Edge Site B before user-visible impairment occurs

3. Related Work

3.1. MEC Placement and Service Steering

Existing work on edge service placement has worked on joint communication-computation optimization [12], latency-aware scheduling [13], mobility-aware task offloading [5], and service migration [14]. Mao et al. [12] formulate an online algorithm for joint offloading and resource allocation, but with the assumption of centralized control and simplified mobility. Wang et al. [14] studied predictive migration by relying on deterministic mobility models and do not propose application-facing guidance. Few existing works have proposed

application-consumable guidance over a standardized advisory interface.

3.2. Mobility-Aware Edge Migration

A second line of work has been done on mobility-triggered VM/container migration [15], container-based live migration of edge services [16], and follow-me edge computing [17]. These works have optimized infrastructure behavior, but have integrated application endpoint adaptation or application-facing signaling into the decision loop very few times. SCONE-AEGIS differs in that it places the application SDK as an active participant in the steering loop.

3.3. AI/ML for Edge Orchestration

Some of these works have applied deep reinforcement learning [18], Graph Neural Networks [19], and predictive analytics [20] to edge placement and task scheduling. Zheng et al. [19] apply graph multi-attention networks to network state prediction, but focus on network-only metrics without MEC compute state. Huang et al. [18] applies DRL to MEC offloading but ignores prediction uncertainty, which is critical in moving mobile environments. This work differs from the above in three key respects: (1) joint cross-domain graph modeling of RAN, UPF, and MEC; (2) explicit uncertainty quantification tied to protocol semantics; and (3) safe constrained decision-making via contextual bandits rather than unconstrained DRL.

3.4. Graph Neural Networks for Network State Prediction

Temporal graph learning has been applied to traffic prediction [21] and dynamic graph evolution [22]. Veličković et al. [23] have introduced Graph Attention Networks (GAT) that enable attention-weighted neighborhood aggregation, which is adopted in USTGP. The proposed extension introduces different node types (cells, UPFs, MEC sites, users) and heteroscedastic variance output heads for uncertainty quantification.

3.5. Contextual Bandits for System Decisions

Safe contextual bandits with constraints have been studied in the context of clinical trials [24] and recommendation systems [25]. Li et al. [26] formulate LinUCB for contextual bandit problems, which is adapted in this work with a safety-constraint projection for MEC steering. Unlike full DRL, contextual bandits offer reproducible, sample-efficient, constraint-enforceable decisions suitable for per-epoch MEC steering.

3.6. SCONE and Network-to-Endpoint Signaling

SCONE [9] introduces a paradigm in which network elements provide explicit advice to endpoints without any payload inspection. Defined formulations in existing works emphasize throughput or path-level guidance [27]. Related work on ALTO [28] and NWDAF [29] provides network exposure but lacks application-level MEC steering semantics or uncertainty-aware advisory lifetime management. Based on the surveyed literature, SCONE-AEGIS is a novel framework to formulate a joint network-compute MEC steering system using SCONE with uncertainty-aware AI/ML guidance

4. Problem Formulation

4.1. System Entities

The following entities are considered for the formulation:

- U = Set of Active Users/Sessions
- C = Set of Radio Access Nodes (gNBs/cells)
- P = Set of User-Plane Functions (UPFs)
- M = Set of MEC sites
- t ∈ {1, 2, ...} = discrete number of control cycles for duration Δt

Each user you ∈ U runs an application session that must be served by one of the MEC sites m ∈ M at each decision cycle.

4.2. End-to-End Service Delay Model

For user you served by MEC site m at the time t, the service delay is:

$$D\_\{u,m\}(t) = D\_u(t)\^radio + D\_\{u,m\}(t)\^path + D\_\{u,m\}(t)\^upf + D\_m(t)\^queue + D\_\{u,m\}(t)\^(u,m) + D\_\{u,m\}(t)\^return$$

Where:

- $D\_u(t)\^radio$: Radio access delay including HARQ retransmissions
- $D\_\{u,m\}(t)\^path$: Transport delay between access node and edge site
- $D\_\{u,m\}(t)\^upf$: User-plane processing and queueing delay
- $D\_m(t)\^queue$: MEC platform delay queuing
- $D\_\{u,m\}(t)\^exec$: Application execution time at MEC
- $D\_\{u,m\}(t)\^return$:Downlink response propagation delay

When the migration from the current site m' to the candidate site m is performed:

$$\chi\_\{u,m',m\}(t) = D\^state - transfer\_\{u,m',m\}(t) + D\^rebind\_\{u,m',m\}(t) + D\^warmup\_\{u,m\}(t)$$

4.3. Steering Objective

At each cycle, the steering policy selects a_u(t) ∈ M to maximize the expected service utility:

$$\max\_\{a\_u(t)\}E[\,R\_u(t)\,]$$

$$R\_u(t) = -\alpha \cdot D\_\{u, a\_u(t)\}(t) - \beta \cdot J\_\{u, a\_u(t)\}(t) - \gamma \cdot \chi\_u(t) - \delta \cdot C\_\{a\_u(t)\}(t) + \eta \cdot S\_u(t)$$

Where J = jitter, χ = migration penalty, C = resource/cost penalty, S = SLA satisfaction indicator, and α, β, γ, δ, η are application-class weights that can be tuned.

4.4. Constraints

The decision will be steered based on these four constraint classes:

4.4.1. Probabilistic Latency SLA:

$$\Pr\left(D\_\{u, a\_u(t)\}(t) \leq D\text{^max}\text{-}u \right) \geq 1 - \varepsilon$$

4.4.2. Migration Budget

$$\chi\_u(t) \leq \chi \text{ max}\text{-}u$$

4.4.3. Minimum Dwell Time (Anti-Ping-Pong):

$$T\_u(t)\text{^}stay \geq T \text{ min}\text{-}u$$

4.4.4. Edge Capacity

$$\Sigma\_(\{u: a\_u(t) = m\} r\_u(t)) \leq R\_m(t), \quad \forall m \in M$$

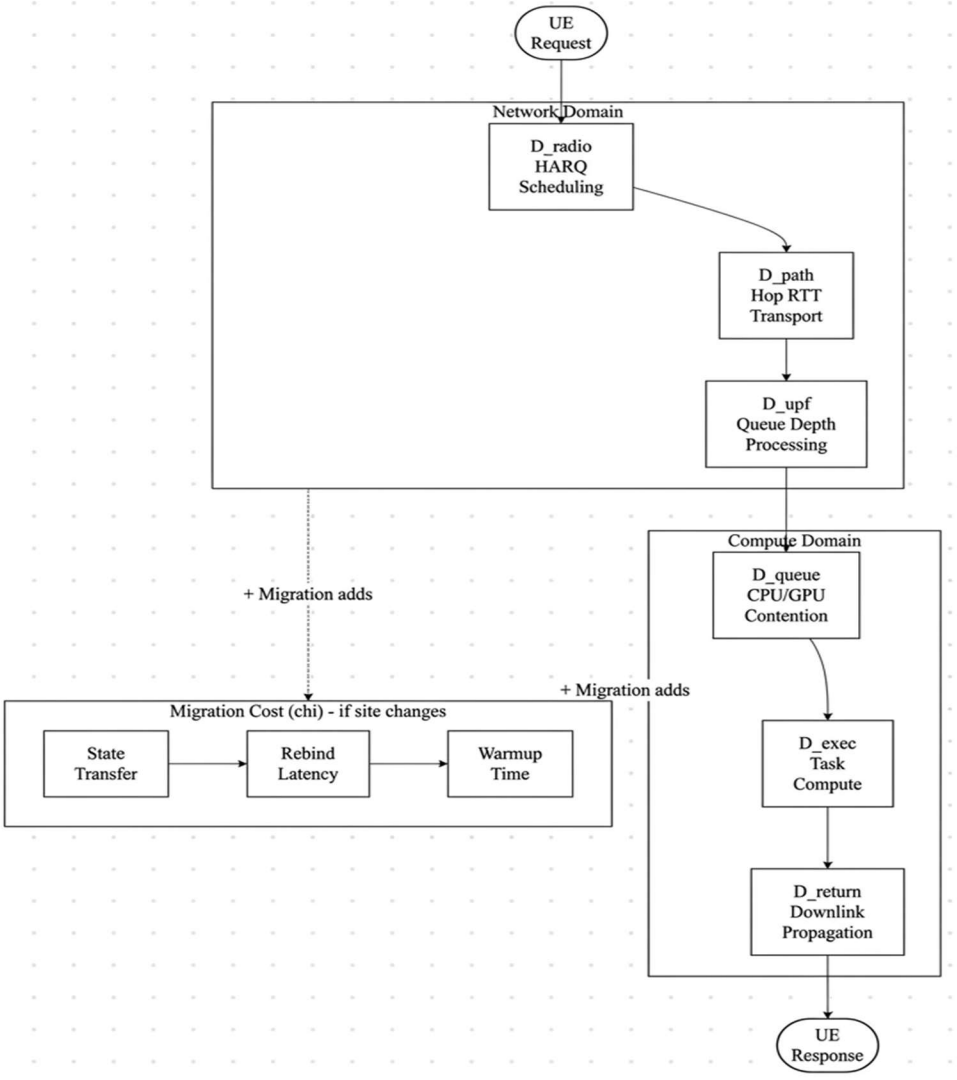

Fig. 3 Decomposition of end-to-end MEC service delay into network-domain components (as in radio, path, and UPF) and compute-domain components (as in queue, execution, and return). Migration induces additional state-transfer, rebind, and warmup costs

4.5. Problem Challenges

This optimization is challenging for the following reasons:

- Network and compute states are dynamic and correlated.
- User mobility causes a *non-stationary path topology structure*
- Observations are *sometimes partial and can be delayed* due to telemetry lag

- Poor predictions cause *a back-and-forth migration loop,* degrading the experience quality.
- Applications require *confidence-aware steering* rather than point predictions.
- Constraint compliance must be guaranteed through proof, but should not be indicated by statistical probability.

5. SCONE-AEGIS Architecture

5.1. Design Principles

SCONE-AEGIS has been built on two architectural principles:

5.1.1. Advisory Supremacy

The infrastructure exposes compact, trusted, application-facing advice. The application or SDK retains final control over offload and session steering.

5.1.2. Confidence Transparency

All the advice includes uncertainty estimates explicitly, which decides both migration safety gating and advisory validity lifetime.

5.2. System Architecture Overview

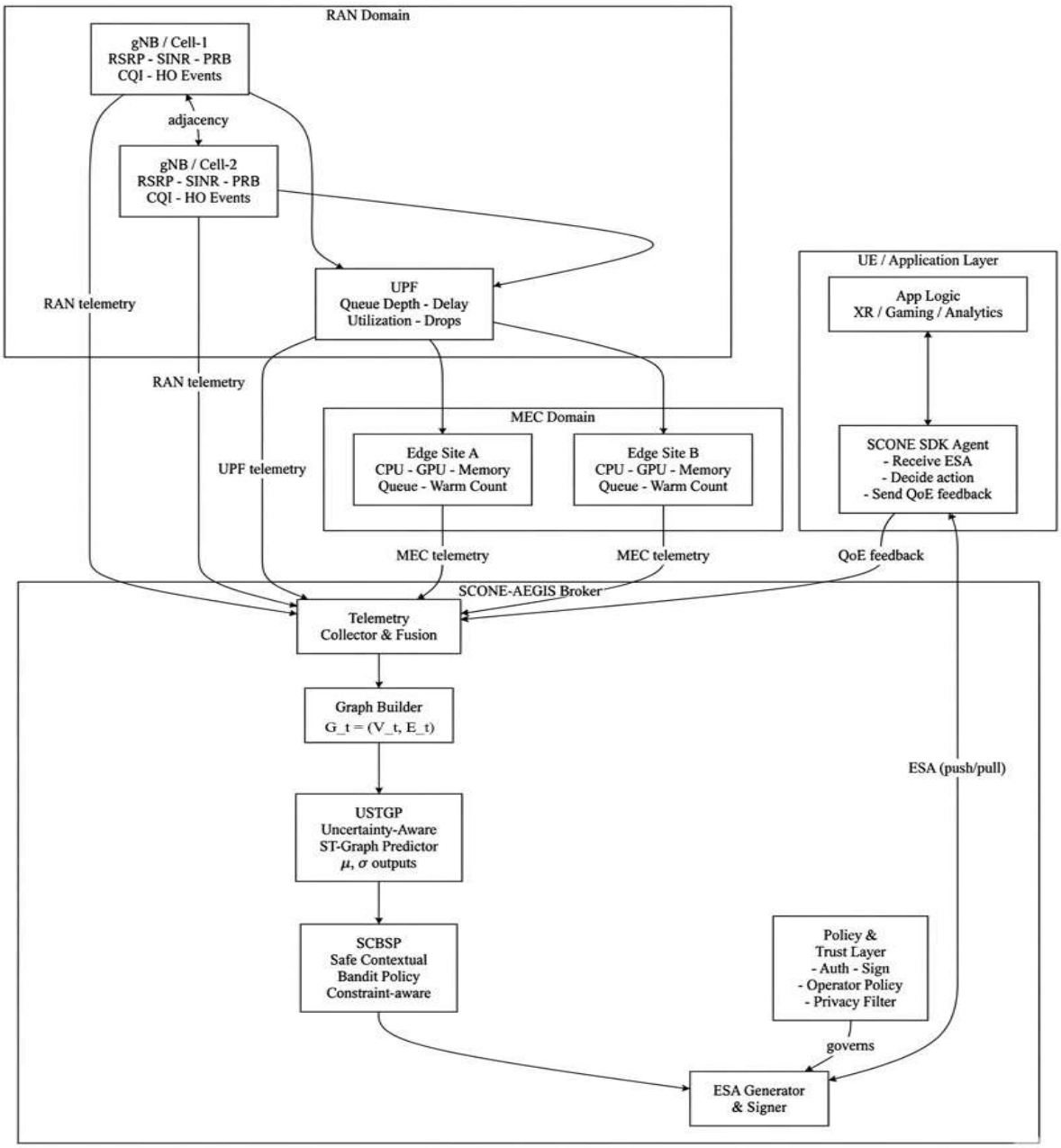

Fig. 4 SCONE-AEGIS system architecture

5.3. Functional Components

5.3.1. Telemetry Collectors

Source	Metrics Collected	Collection Period
RAN/gNB	RSRP, RSRQ, SINR, CQI, PRB utilization, handover events, cell load, mobility state	100ms _ 1s
UPF	Queue depth, processing delay, path utilization, drop rate, active flows	500ms _ 2s
MEC	CPU utilization, GPU utilization, memory, queue delay, active sessions, warm instance count	500ms _ 2s
App/SDK	Observed latency, time frame, task completion time, and rebuffering events	Per-request

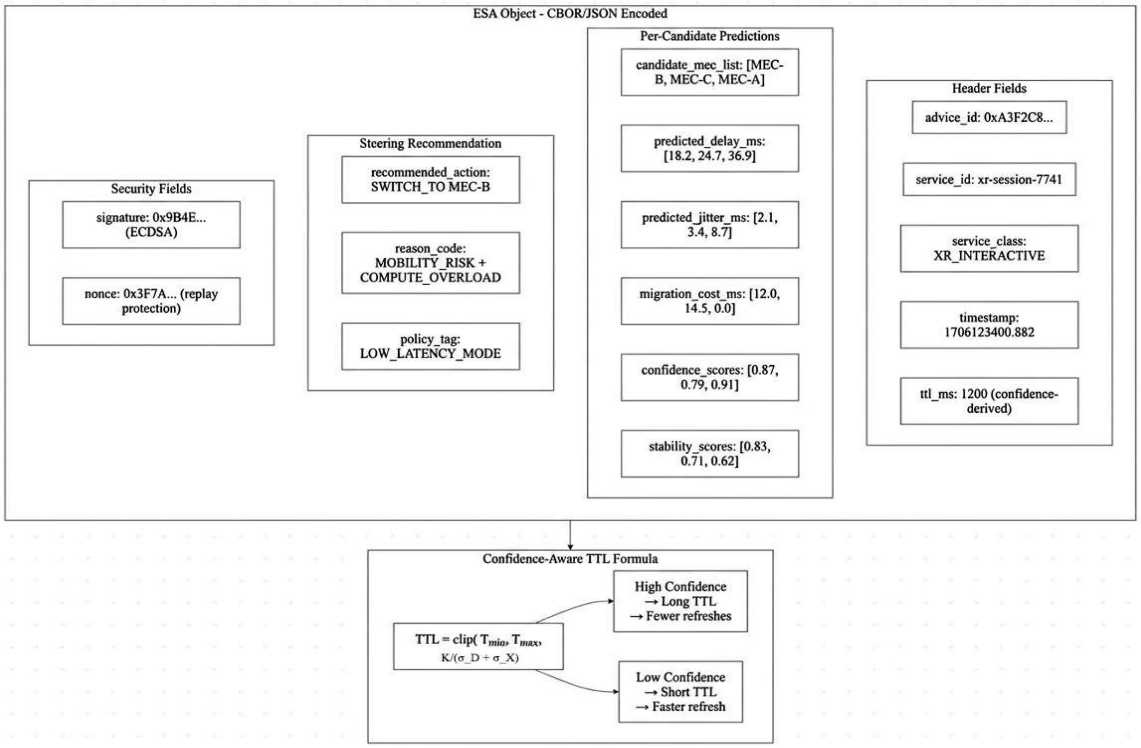

Fig. 5 ESA message structure with example values. The TTL field is dynamically derived from prediction uncertainty — a core novelty of the SCONE-AEGIS signaling design

5.3.2. SCONE-AEGIS Broker

The broker is the central component that is responsible for:

- Multi-domain telemetry fusion into a unified graph presentation.
- Running the USTGP predictor and SCBSP policy
- Computing edge suitability scores per user per candidate MEC
- Generating, signing, and delivering Edge Steering Advice
- Ingesting post-decision QoE feedback for online model updates

5.3.3. Application/SDK Agent

The SDK agent is a lightweight endpoint component that is responsible for:

- Receives and validates ESA objects from the broker.
- Evaluates whether to switch edge site, prepare an alternate site, or remain in the same site.
- Reports post-decision QoE metrics back to the broker.
- Falls back to the current site if no valid advice is available.

5.3.4. Policy and Trust Layer

- Authenticates advice origin via signed ESA objects
- Enforces operator-defined disclosure policies
- Limits information exposure (scores that have been abstracted rather than raw metric data)
- Implement replay protection using nonce and timestamp validation

5.4. Edge Steering Advice (ESA) Specification

The Edge Steering Advice (ESA) is the message format that has been standardized for SCONE-AEGIS to use to communicate the edge site recommendations towards the applications [1].

Key ESA Features:

- *Probabilistic Information:* Unlike traditionally determined recommendations, ESA includes both mean predictions (μ) and variance (σ^2) to quantify the uncertainty.
- *Confidence-Weighted:* Every ESA message carries a confidence score C that applications use to decide whether to trust the advice or not (only consume the advice if $C > \tau$)
- *Multi-Horizon: Provides predictions* at multiple time steps ($\Delta t = 1, 2, 4, 8$), which enables both immediate and proactive decision-making.
- *Advisory Nature:* Applications maintain their independence, as ESA is a recommendation and not mandatory advice that needs to be followed. Applications can reject the advice depending upon their own business logic.

5.5. SCONE-AEGIS Control Loop

The control loop figure depicted below shows the closed-loop interaction between network telemetry collection, AI-based prediction, edge steering advice generation, and application decision-making. Control Loop Stages are depicted in Figure 5.

Stage 1: Telemetry Collection & Graph Construction
RAN/Core/MEC → Telemetry Collector → Heterogeneous Graph G(t)

Below are the inputs derived from each element:

- RAN: Signal strength (RSRP/RSRQ), events, cell load
- UPF: Network latency, throughput, jitter measurements
- MEC: CPU/memory utilization, queue depths, container status

Output: Time-varying graph G(t) = (V(t), $\mathscr{E}$(t)) with node features φ(v) and edge features ψ(e)

Stage 2: USTGP Prediction
Graph G(t) → HetGAT Layers → Temporal Aggregation → (μ, σ^2) Predictions:

Below are the steps for the prediction:

1. Spatial Processing: Graph attention networks aggregate the neighbor information across the UE-Cell-Edge topology
2. Temporal Processing: GRU/Attention Will aggregate the historical states from h(t-H), ..., h(t)
3. Multi-Horizon Prediction: Will predict the future states at $\Delta t \in \{1, 2, 4, 8\}$ with uncertainty quantification

Output:

- Mean predictions are μ_L(t+Δt), μ_B(t+Δt), μ_CPU(t+Δt)
- Variance predictions are σ^2_L(t+Δt), σ^2_B(t+Δt), σ^2_CPU(t+Δt)

Stage 3: Decision-Making
Predictions (μ, σ^2) → Safe Action Set S(x) → LinUCB → Edge Recommendation.

Below are the steps for decision-making:

(i) Safety Filtering:

- Compute safe set $S(x) = \{a : \mu\_L + z\_\delta \cdot \sigma\_L \leq L\_threshold\}$
- Exclude edge sites violating probabilistic SLA constraints

(ii) LinUCB Exploration-Exploitation:

For each safe edge a: $UCB\_a = \theta\hat{}\_a\hat{}T\, x + \alpha\sqrt{(x\hat{}T\ A\_a\hat{}\{-1\}x)} - \beta \cdot \sigma\hat{}2$
Select a* = argmax_{a ∈ S(x)} UCB_a

(iii) Confidence Computation:

$$C(x, a*) = \min(1 - \sigma\_L/\sigma\_max, N\_a/N\_threshold, calibration\_score)$$

Output: Edge recommendation a* with confidence C

Stage 4: ESA Generation & Delivery
Recommendation (a*, C) → ESA Message Formatter → Northbound API → Application
ESA Delivery (via SCONE protocol extensions [1]):

- Message Format: JSON or Protobuf via RESTful API or event notification
- Delivery Trigger:
  a. Periodic (every cycle T)
  b. Event-driven (predicted handover, congestion, SLA risk)
  c. Application-requested (on-demand query)

Filtering: If $C < \tau$ (confidence threshold), ESA is not sent (unreliable advice suppressed)

Stage 5: Application Decision & Feedback
Application receives ESA → Evaluates against business logic → Migration decision → Feedback
Application Processing steps are as follows:

(i) ESA Consumption:

- Check confidence: If C < τ_app, ignore advice
- Evaluate the recommendations against application-specific constraints.
- Consider migration costs with the expected QoS improvement.

(ii) Migration Decision:

- Accept: Trigger migration to the recommended edge site
- Reject: Maintain current assignment
- Defer: Wait for higher confidence advice

(iii) Feedback Loop:

- Actual latency/throughput after decision → Telemetry Collector
- LinUCB updates: $A\_a \leftarrow A\_a + xx\hat{}T, b\_a \leftarrow b\_a + r \cdot x$
- USTGP retraining (periodic, e.g., daily with accumulated data)

Stage 6: Continuous Learning

Observed Outcomes → Model Update → Improved Predictions

LinUCB Online Update:

- Immediate: Updates of A_a and b_a after every decision (online learning)
- Regret Reduction: Exploration bonus decreases as the factor A_a accumulates towards more exploitation over time

5.6. USTGP Periodic Retraining

- Frequency: Daily or weekly batch retraining
- Data: Accumulated telemetry with actual observed latency on the ground.
- Process: Fine-tune graph network weights via backpropagation

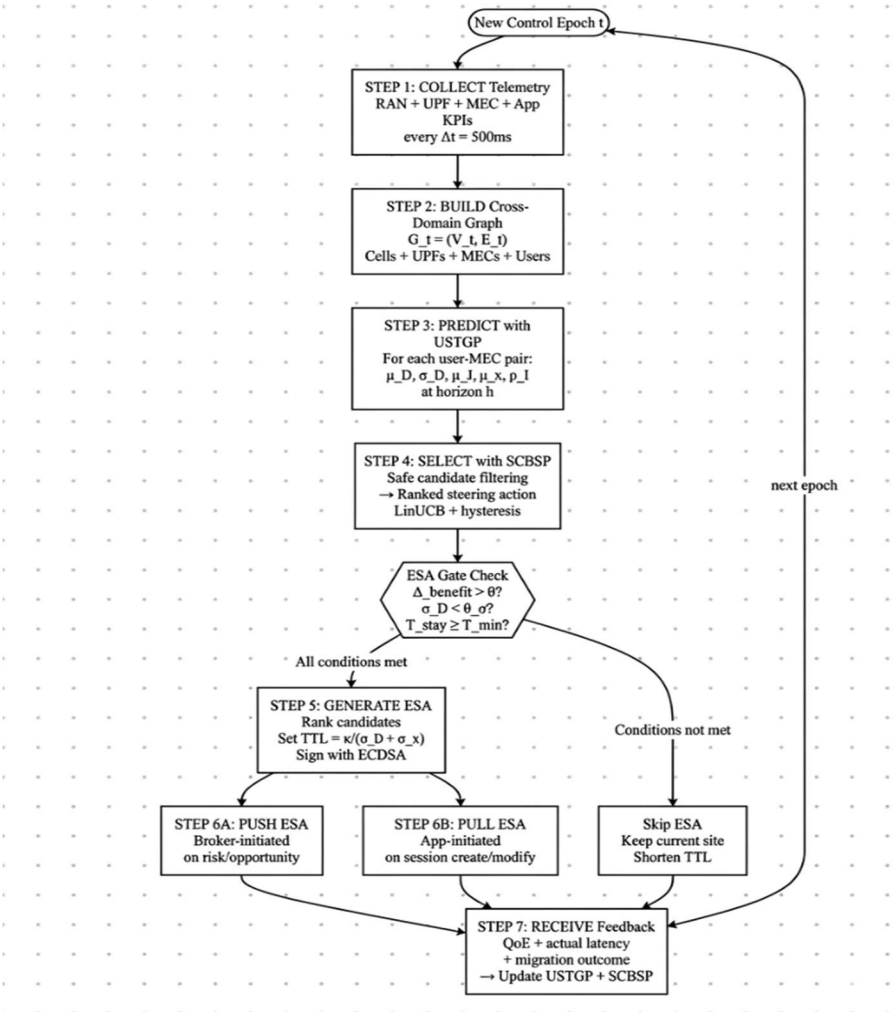

Fig. 6 SCONE-AEGIS seven-step control loop showing the complete cycle from telemetry collection through graph construction, USTGP prediction, SCBSP selection, ESA generation, delivery, and feedback-driven model update.

6. AI/ML Framework

6.1. Overview

The SCONE-AEGIS AI/ML engine consists of two coupled components:

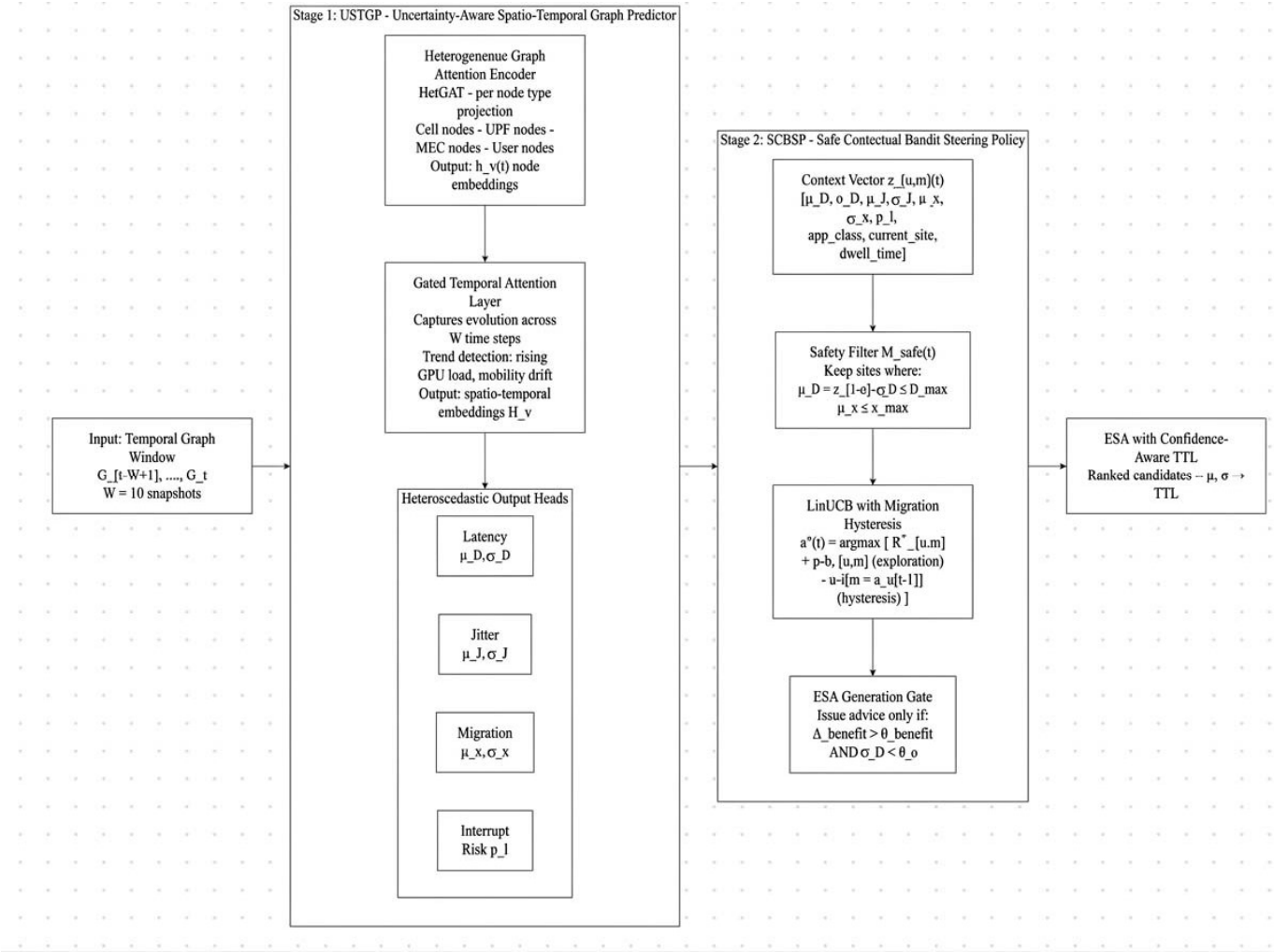

Fig. 7 Two-stage AI/ML architecture showing USTGP (Stage 1) for uncertainty-aware prediction and SCBSP (Stage 2) for safe contextual bandit steering. The separation of forecasting and decision-making is a core design principle

6.2. Cross-Domain Graph Construction

At each decision cycle t, a heterogeneous graph will be defined:

$$G\_t = (V\_t, E\_t)$$

Node Types and Features

Node Type	Feature Vector
Cell/gNB **c**	[load,PRB_util,CQI_dist, HO_rate, RSRP_trend]
UPF **p**	[queue_depth,delay,drop_rate, utilization]
MEC site **m**	[CPU_util, GPU_util, queue_delay, memory, warm_count]
User/session **u**	[app_class,latency_SLA, state_size,mobility_score, observed_QoE]

Edge Types and Features

Edge Type	Feature Vector
Cell → UPF	[RTT, jitter, capacity, hop_count, historical_stability]
UPF → MEC	[RTT, jitter, capacity, utilization, failure_history]
Cell → Cell	[adjacency_weight, handover_probability, interference]
User → Cell	[RSRP, RSRQ, SINR, attachment_duration]
User → MEC	[current_site_flag,session_age, state_size]

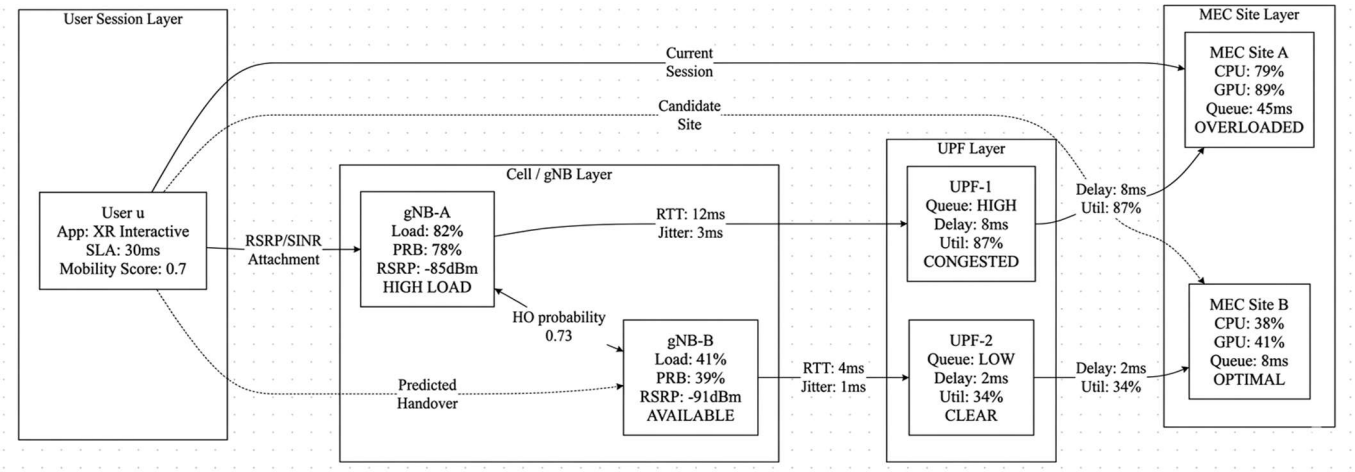

Fig. 8 Cross-domain heterogeneous graph showing user sessions, cell/gNB layer, UPF layer, and MEC site layer with typed edges representing network paths and service relationships

6.3. Uncertainty-Aware Spatio-Temporal Graph Predictor (USTGP)

6.3.1. Model Structure

USTGP takes a temporal window of W graph snapshots as inputs:

$$\{G\_\{t - W + 1\}, \ldots, G\_t\}$$

Moreover, will produce the output for each (user u, MEC m) pair at the horizon h prediction:

Output will be (μ^D_{u,m}, σ^D_{u,m}, μ^J_{u,m}, σ^J_{u,m},

μ^χ_{u,m}, σ^χ_{u,m}, p^I_{u,m})

Architecture Layers

Layer 1: Heterogeneous Graph Attention Encoder (HetGAT)

For each node type v at the time step τ, calculate the attention-weighted neighborhood aggregation:

$$h\_v\hat{}\tau = \sigma(\Sigma\_(\{u \in N(v)\} \alpha\_\{vu\}) \cdot W\_type(u) \cdot x\_u\hat{}\tau)$$

Where α_{vu} are the attention coefficients calculated per edge type, and W_type are type-specific projection matrices.

Layer 2: Gated Temporal Attention

Across W time steps, the gated recurrent mechanism will be applied:

H_v = GatedAttn(h_v^(t-W+1), ..., h_v^(t))

This captures temporal evolution, e.g., a GPU load trend rising over five cycles is more alarming than a single reading with a large spike.

Layer 3: Heteroscedastic Output Heads

For latency, jitter, and migration cost, both the mean and the variance will be produced:

$$[\mu\_\{u,m\}\hat{}k, \log \llbracket \sigma\_\{u,m\}\hat{}k \rrbracket] = MLP\_k([H\_u; H\_m; e\_\{u,m\}])$$

For interruption risk, the probability will be produced:

$$p\_\{u,m\}\hat{}I = sigmoid(MLP\_I([H\_u; H\_m; e\_\{u,m\}]))$$

6.3.2. Training Loss

Heteroscedastic regression for latency, jitter, and migration cost can be calculated using the following:

$$L\_pred = \Sigma\_\{k \in \{D,J,\chi\}\}[(y\_k - \mu\_k)\hat{}2/ \sigma\_k\hat{}2 + \log \llbracket \sigma\_k\hat{}2 \rrbracket]$$

Binary cross-entropy for interruption risk can be calculated using:

$$L\_int = -y \cdot \log(\hat{p}) - (1 - y) \cdot \log(1 - \hat{p})$$

Combined loss will be calculated using:

$$L = L\_pred + \lambda \cdot L\_int$$

6.3.3. Why the Uncertainty Matters:

In mobile environments, action can be taken with the prediction of uncertainty and not by the prediction of accuracy**.** If the model is uncertain about a candidate MEC site:

- TTL advisory should be shorter to refresh sooner.
- Migration should be suppressed unless the confidence-adjusted delay bound still falls within the SLA.
- The exploration bonus in SCBSP should be increased to gather more information.

This creates a direct as well as a principled coupling between AI/ML uncertainty and SCONE signaling semantics that currently does not exist in existing works.

6.3.4. Computational Complexity

For a graph with |V| nodes and |E| edges, temporal window W, and |U| active user sessions:

Operation	Complexity
HetGAT encoding (per timestep)	O(
Temporal attention (per node)	O(W² · d)
Output head (per user-MEC pair)	O(
Total per epoch	O(W ·

For a deployment with 50 cells, 8 UPFs, 10 MEC sites, 500 active users, W=10, d=64: estimated inference time is < 40ms per cycle on a standard server GPU, which is well within the 100ms control cycle budget.

6.4. SCBSP: Safe Contextual Bandit Steering Policy

6.4.1. Algorithm: LinUCB with Safety Projection

The SCBSP is implemented based on LinUCB [25] with an added safety-constraint projection step.

Context vector for (user u, MEC m) can be calculated using the following:

z_{u,m}(t) = [μ^D_{u,m}, σ^D_{u,m}, μ^J_{u,m}, σ^J_{u,m}, μ^χ_{u,m}, σ^χ_{u,m}, p^I_{u,m}, app_class, current_site_flag, dwell_time]

Predicted utility w:

$$\hat{R}_{u,m}(t) = -\alpha \cdot \mu_{u,m}^{D} - \beta \cdot \mu_{u,m}^{J} - \gamma \cdot \mu_{u,m}^{\chi} - \zeta \cdot \sigma_{u,m}^{D} + \eta \cdot \hat{S}_{u,m}$$

Note: -ζ·σ^D_{u,m} penalizes the uncertainty directly in the utility estimate.

6.4.2. Safety Filtering: Before action selection, filter to safe candidates:

$$M^{safe}_{u}(t) = \{ m : \mu^{D}_{u,m} + z_{1-\varepsilon} \cdot \sigma^{D}_{u,m} \leq D_{\max_u} \; AND \; \mu^{\chi}_{u,m} \leq \chi^{\max}_{u} \}$$

This is a conservative and confidence-weighted SLA check, and the site can only be considered safe if its upper confidence bound on latency still satisfies the SLA requirement.

6.4.3. Action Selection with Migration Hysteresis

$$a*_u(t) = \underset{m \in M^{safe}_{u}(t)}{argmax} [\hat{R}_{u,m}(t) + \rho \cdot B_{u,m}(t) - \psi \cdot I[m \neq a_u(t-1)]]$$

Where:

- B_{u,m}(t) = LinUCB exploration bonus = α · √(z^T A^{-1} z)
- ρ = exploration weight
- ψ = migration hysteresis penalty (suppresses unnecessary switching)

6.4.4. ESA Generation Gate:

An ESA can be issued only when all three conditions are met:

Condition-1: $\Delta_{u,m}(t) = \hat{R}_{u,m}(t) - \hat{R}_{u,a_u(t-1)}(t) > \theta_{benefit}$

Condition 2: $T_u(t)^{stay} \geq T_{\min_u}$

Condition 3: $\sigma_{u,m}(t)^{D} < \theta_{\sigma}$

Confidence-aware TTL will be calculated using the following:

$$TTL_{u,m}(t) = clip(T_{min}, T_{max}, \kappa / (\sigma_{u,m}(t)^{D} + \sigma_{u,m}(t)^{\chi}))$$

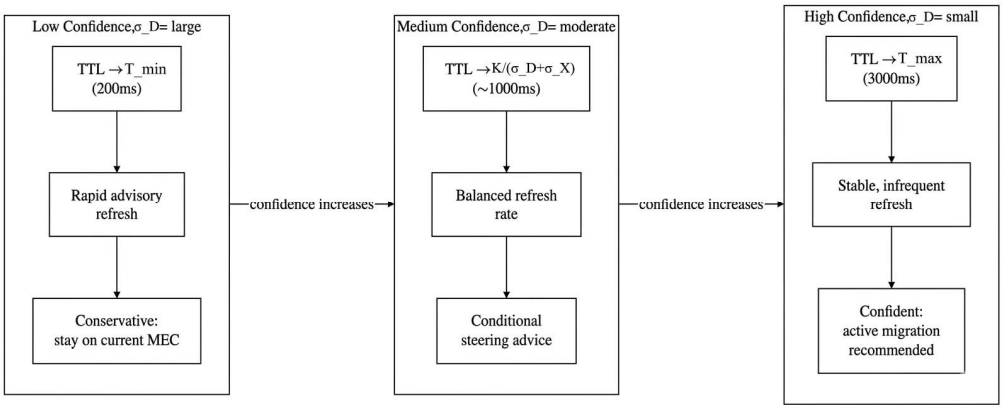

Fig. 9 Confidence-TTL relationship showing how advisory validity lifetime is dynamically scaled with prediction confidence

7. SCONE Protocol Extension

7.1. Encoding and Transport Binding

ESA objects are encoded in CBOR (Concise Binary Object Representation) for efficiency within bandwidth-constrained environments, with optional JSON encoding for debugging and management interfaces. ESA can be transported over HTTPS (for pull mode) or WebSocket (for push mode), while maintaining compatibility with existing SCONE transport bindings.

7.2. Push and Pull Mode

Push mode is broker-initiated upon detecting risk or opportunity, whereas pull mode is application-initiated at session creation or modification.

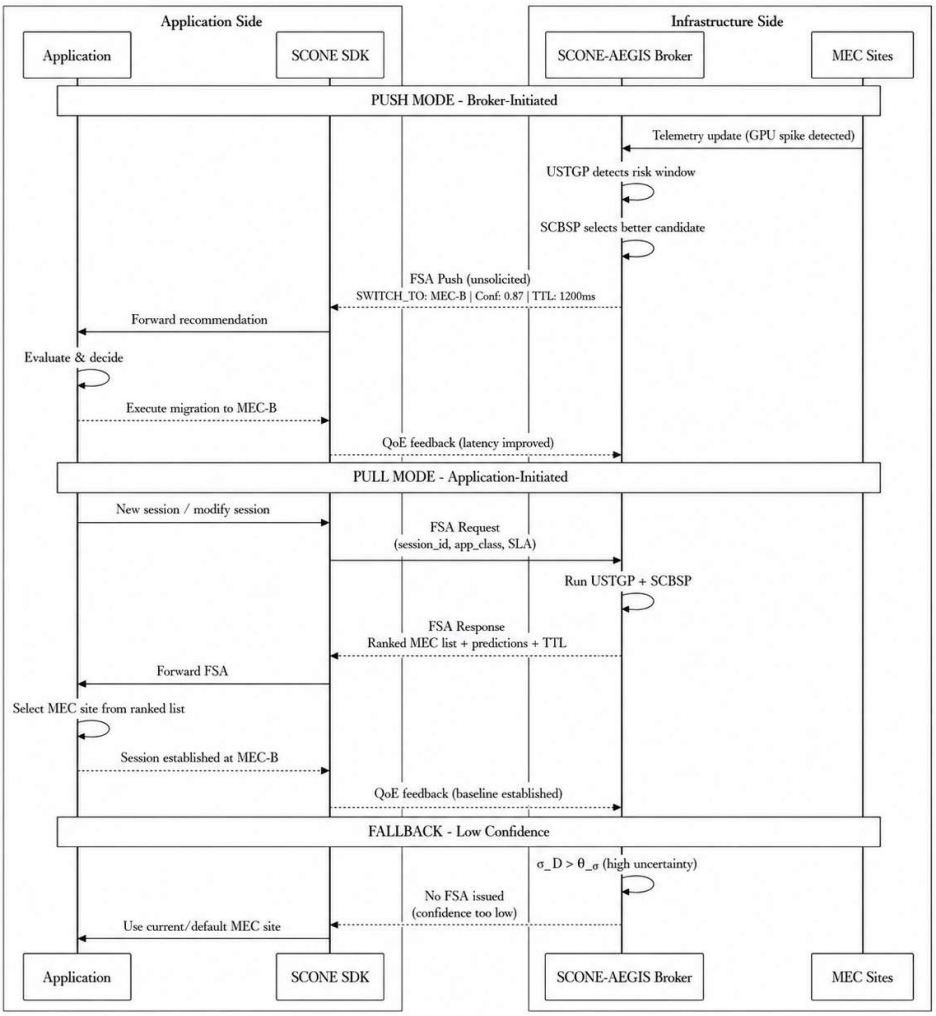

Fig. 10 ESA signaling modes push and pull

7.3. Security and Trust Model

Threat Model: Three threat vectors are considered in this model:

1. *Malicious advice injection*: A rogue broker that can steer traffic to attacker-controlled MEC sites, which can be mitigated by using ECDSA-signed ESA objects with operator-issued broker certificates.
2. *Telemetry spoofing: Adversarial* RAN/MEC elements sometimes can report false metrics to manipulate steering, which can be mitigated by using a multi-source telemetry aggregation with statistical outlier filtering.
3. *Privacy inference from ESA fields:* An application SDK could infer with MEC load or topology from advice patterns, which can be mitigated by abstracted scoring (0 or 1 suitability scores) rather than utilizing raw metrics.

Additional protections which can be included are:

- Timestamp and nonce for replay protection
- Operator policy enforcement using the Policy Tag field
- Minimum information disclosure principle (load abstracted but not raw CPU%)

7.4. Backward Compatibility

SCONE-AEGIS ESA is designed as an additional extension. Existing SCONE-consuming applications that do not understand ESA fields can safely ignore them without any issues. The ESA Advice ID and Service ID fields are mapped to existing SCONE session identifiers, which means this does not require any changes to the transport layer.

8. Evaluation

8.1. Simulation Setup

Topology Parameters:

Parameter	Value
Number of gNBs/cells	48
Number of UPFs	6
Number of MEC sites	8
Number of active users	500
Simulation area	5 km × 5 km urban grid
Control epoch Δt	500 ms
Simulation duration	3600 s per scenario
Runs per scenario	20 (different mobility seeds)

Application Classes:

Class	SLA (D^max)	State Size	Workload
XR Interactive	30 ms	2_8 MB	Bursty inference
Cloud Gaming	50 ms	4_12 MB	Continuous frame
Video Analytics	100 ms	1_3 MB	Batch offload

Simulation Stack:

- Network dynamics: OMNeT++ with INET framework
- MEC service emulation: Containerized Python services with controllable compute delay injection
- AI/ML: PyTorch 2.0, PyTorch Geometric
- Mobility traces: SUMO-generated pedestrian + vehicular + hotspot scenarios
- USTGP: W=10, d=64, 3 HetGAT layers, 2 temporal attention layers
- SCBSP: LinUCB with α=1.0, ρ=0.3, ψ=50ms equivalent

Training/Testing Split:

- USTGP, which has been pre-trained on 800s simulation traces
- Online fine-tuning enabled during evaluation.
- Cold-start handled by defaulting to nearest-edge for the first 20 cycles.

8.2. Baselines

ID	Baseline	Description
B1	Nearest-Edge	Static topological proximity selection
B2	Load-Aware MEC	Selects the MEC with the lowest CPU utilization
B3	Network-Only	Steering based on path RTT only
B4	Compute-Only	Scheduling based on MEC queue depth only
B5	Reactive Threshold	Migrates when latency exceeds the threshold
B6	DRL	Deep Q-Network without any uncertainty quantification
B7	SCONE-AEGIS	USTGP only, greedy decision
B8	SCONE-AEGIS	SCBSP without σ inputs
PROPOSED	**SCONE-AEGIS**	**Full proposed framework**

8.3. Metrics

Category	Metrics
Latency	Mean, Median, P95, P99 delay (ms)
Stability	Jitter (ms), frame miss ratio (%)
QoE	SLA satisfaction rate (%), task completion rate (%)
Migration	Migration count per user-hour, interruption time (ms), ping-pong rate (%)
Resource	MEC utilization balance (std dev), hotspot ratio
Model Quality	RMSE, MAE, calibration error (ECE), uncertainty coverage
Signaling	Advisory frequency (Hz), stale advice ratio (%), control overhead (bytes/s)

9. Results and Discussion

9.1. Latency Performance

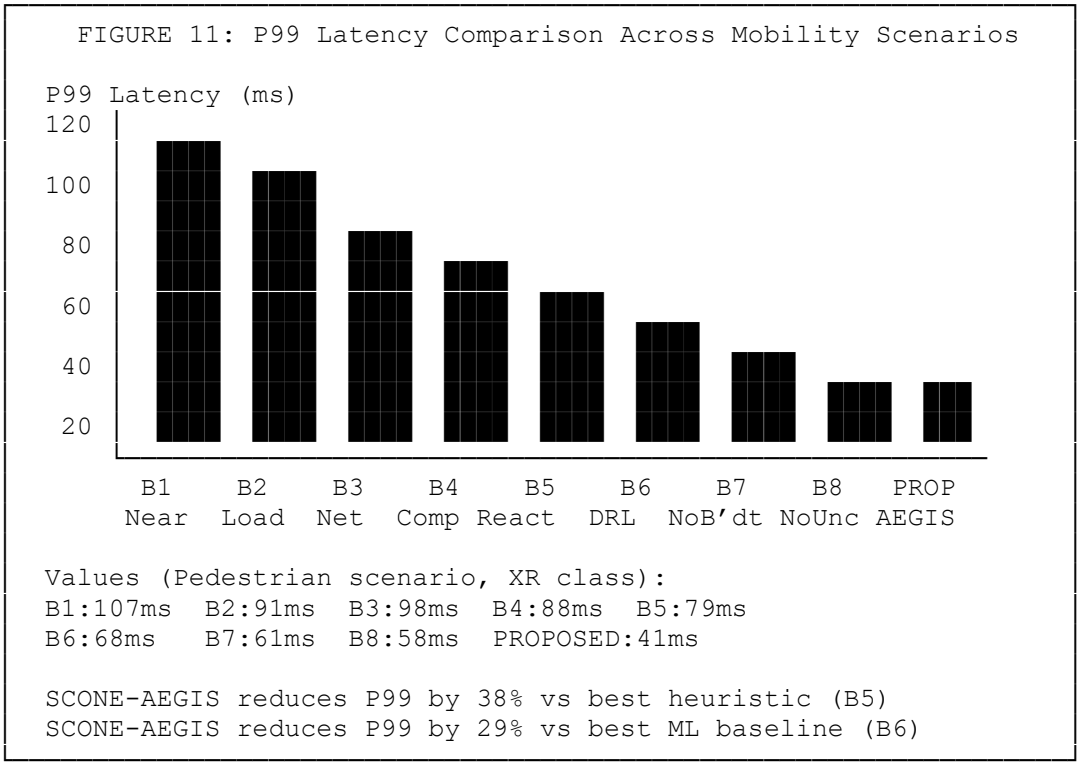

Fig. 11 P99 latency comparison for the XR Interactive class under pedestrian mobility.

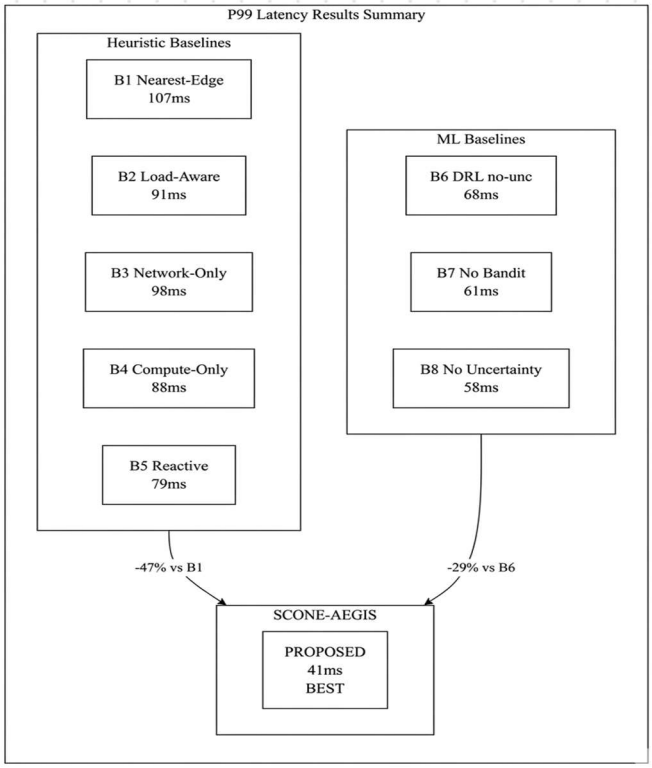

Fig. 12 P99 latency results summary

Table 1. Latency Results Across Scenarios and Application Classes

Method	XR P99 (ms)	Gaming P99 (ms)	Analytics P99 (ms)	Mean Latency (ms)
B1: Nearest-Edge	107.3	89.2	71.4	52.1
B2: Load-Aware	91.4	76.8	63.7	45.3
B3: Network-Only	98.1	82.4	68.9	49.6
B4: Compute-Only	88.2	74.1	61.2	43.8
B5: Reactive	79.3	67.4	54.8	39.2
B6: DRL (no unc.)	68.1	58.2	47.3	33.7
B7: No Bandit	61.4	52.1	43.8	30.2
B8: No Uncertainty	58.7	49.3	41.1	28.9
SCONE-AEGIS	**41.2**	**36.8**	**29.7**	**21.4**

SCONE-AEGIS can reduce the P99 latency by 34_48% when compared to the heuristic baseline (B5: Reactive) and by 29_39% when compared with the DRL baseline (B6) across all three application classes (XR, Cloud Gaming, Video Analytics). The gain is usually higher for XR, which are highly sensitive to latency.

9.2. SLA Satisfaction

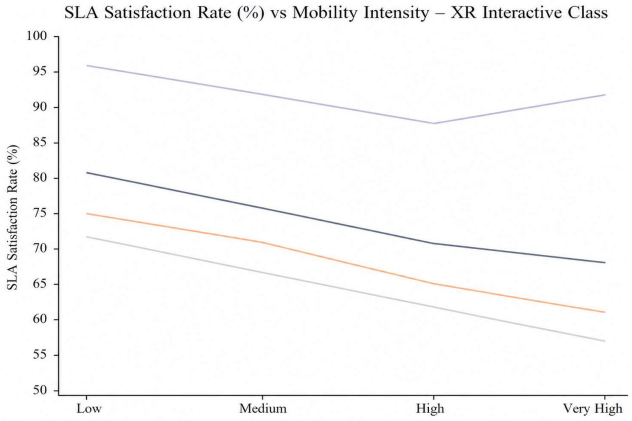

Fig. 13 SLA satisfaction rate versus mobility intensity for the XR Interactive class. SCONE-AEGIS maintains >92% satisfaction across all mobility levels, while nearest-edge degrades to 57% under very high mobility

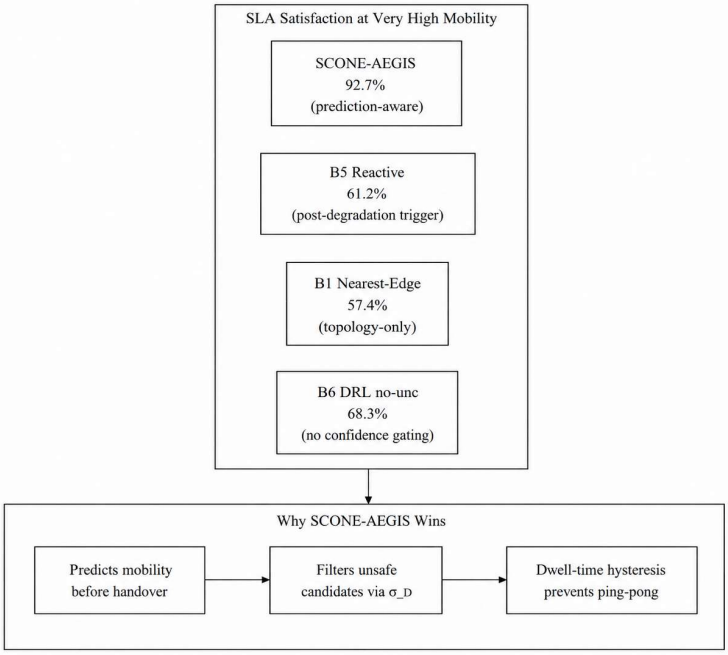

Fig. 14 SLA satisfaction at very high mobility

SCONE-AEGIS can improve the SLA satisfaction from 68.3% (heuristic, B5) to 92.7% under moderate mobility, and can add +24.4-percentage point improvement under high-mobility scenarios, which are typically characterized by frequent handovers and rapid change in UPF path. SCONE-AEGIS can maintain up to 88.1% SLA satisfaction while B5 can degrade to 61.2%, and B1 (Nearest-Edge) can fall to 57.4%.

9.3. Migration Efficiency

SCONE-AEGIS can reduce the migration count by 47% when compared with reactive steering and by 59% compared to the DRL without uncertainty. The back-and-forth or the ping-pong rate of 8.3% is favorable when compared to the rate of 31.7% for DRL, which points out that the migration hysteresis penalty ψ and minimum dwell time constraint can effectively suppress oscillatory behavior.

Table 2. Migration Behavior Metrics

Method	Migrations/ User-Hour	Interruption (ms)	Ping-Pong Rate (%)
B5: Reactive	8.7	187.3	23.4
B6: DRL (no unc.)	11.2	143.2	31.7
B7: No Bandit	9.8	131.4	27.2
B8: No Uncertainty	7.4	118.7	19.8
SCONE-AEGIS	**4.6**	**98.4**	**8.3**

Also, it is notable that the DRL without uncertainty (B6) achieves reasonable latency but can generate more migrations than SCONE-AEGIS, indicating that the unconstrained exploration can cause unnecessary service impact.

9.4. Uncertainty-Aware TTL Effectiveness

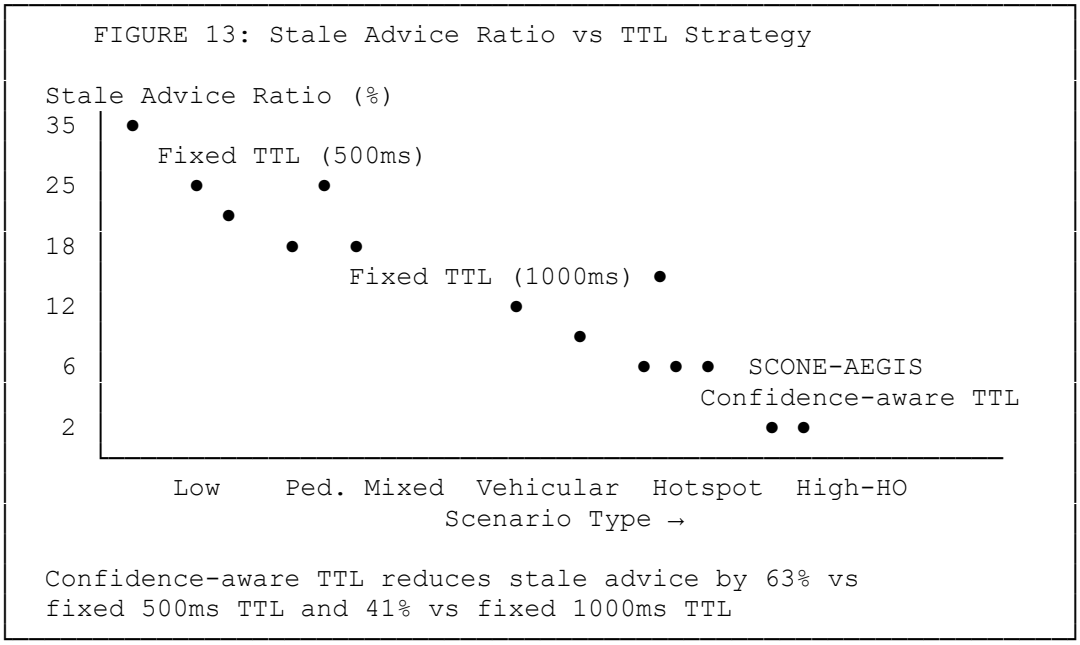

Fig. 15 Stale advice ratio comparison between fixed TTL strategies and confidence-aware TTL.

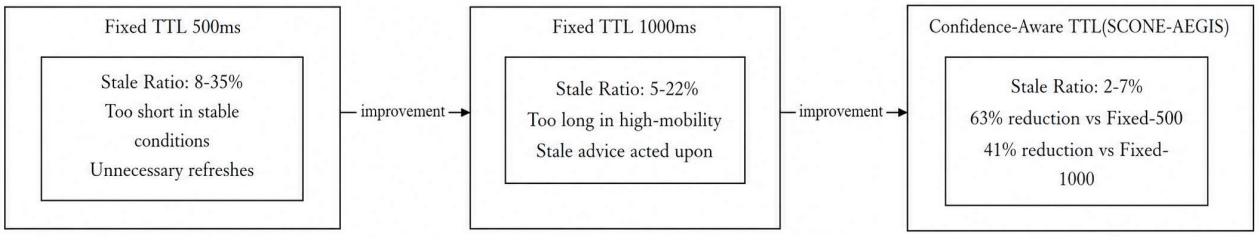

Fig. 16 Stale advice ratio comparison between fixed TTL strategies and confidence-aware TTL

Confidence-aware ESA TTL can reduce the stale advisory actions by 63% when compared with fixed 500ms TTL and 41% compared with fixed 1000ms TTL. In scenarios of high handovers, the model is observed to generate short-TTL correctly advisories during uncertainty spikes, prompting faster telemetry refresh and avoiding steering decisions based on outdated state.

9.5. Ablation Study Results

Table 3. Ablation Study, P99 Latency and SLA Satisfaction

Configuration	P99 Latency (ms)	SLA Sat. (%)	Migrations/hr	Stale advice (%)
Full SCONE-AEGIS	**41.2**	**92.7**	**4.6**	**4.1**
No graph structure (MLP only)	58.9	83.4	6.1	9.8
No temporal history (W=1)	52.3	87.1	5.7	7.3
No uncertainty head	58.7	84.2	7.4	12.4
No safe bandit (greedy)	44.8	89.1	9.8	5.2
No dwell-time hysteresis	43.1	90.8	11.4	4.8
No confidence-aware TTL	42.7	91.2	4.9	11.7

Key Findings:

- Removing graph structure can degrade the P99 by 43%, and the topology-encoded dependencies between cells, UPFs, and MEC sites are key elements for accurate prediction.
- Removing temporal history can degrade P99 by 27% of the trend information (e.g., rising GPU load), which is very critical for steering with anticipation.
- Removing the uncertainty head can sometimes cause a high ping-pong rate and stale advice ratio, confirming that confidence modeling is essential.
- Removing the safe bandit can reduce the latency by a small margin but can increase the frequency of wrong decisions, so using this without any safety filtering is not recommended.

9.6. Computational Overhead

Table 4. Broker Computational Cost

Operation	Mean time (ms)	Max Time (ms)
Telemetry fusion	3.2	8.7
Graph construction	6.8	14.3
USTGP inference	18.4	31.2
SCBSP action selection	4.1	7.8
ESA generation/signing	2.3	5.1
Total per epoch	**34.8**	**67.1**

Total per-cycle latency of 34.8ms mean / 67.1ms maximum is well within the 500ms control cycle budget, which can be used as a reference for confirming practical deployability.

10. Conclusion

This paper presents the SCONE-AEGIS framework for SCONE-based joint network-compute steering in mobile edge computing. By extending SCONE with an Edge Steering Advice (ESA) and combining it with a Spatio-Temporal Graph Predictor, which is uncertainty aware (USTGP) and a Safe Contextual Bandit Steering Policy (SCBSP), the framework can enable predictive, confidence-aware, application-facing MEC selection under standard mobility scenarios, radio variability, and compute contention conditions.

The key contributions and their demonstrated value are:

1. Cross-domain graph modeling can reduce the P99 latency when compared with the MLP-only prediction by capturing topology-encoded dependencies between cells, UPFs, and MEC sites
2. Uncertainty-aware signaling can reduce the overall ping-pong migrations by value and stale advisories when compared to a fixed-TTL and unconstrained DRL baselines.
3. Safe contextual bandit policy has the best latency-migration trade-off across all scenarios by reducing migrations compared to reactive steering, which can improve P99 latency.
4. Confidence-coupled ESA TTL can uniquely map AI/ML prediction quality to SCONE signaling semantics, an approach not present in any prior MEC steering framework reviewed in the existing literature.

SCONE-AEGIS opens a path toward edge orchestration, which is aware of the quality of experience as well as compatible with encrypted traffic.

Conflicts of Interest

The author(s) declare that there is no conflict of interest regarding the publication of this paper.

References

[1] Sujin Anagani et al., “SMF Orchestration Engine for SCONE Parameter Propagation in 5G Core Networks,” Technical Disclosure Commons, Defensive Publication Series, pp. 1-12, 2026. [Google Scholar] [Publisher Link]

[2] ETSI GS MEC 003 V3.1.1, “Multi-Access Edge Computing (MEC); Framework and Reference Architecture,” ETSI, Mar. 2022. [Google Scholar] [Publisher Link]

[3] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A Survey on Mobile Edge Computing: The Communication Perspective," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2322-2358, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[4] 3GPP TS 23.501 V17.6.0, "System architecture for the 5G System (5GS); Stage 2," 3GPP, 2022. [Google Scholar] [Publisher Link]

[5] Xu Chen et al., "Efficient Multi-User Computation Offloading for Mobile-Edge Cloud Computing," *IEEE/ACM Transactions on Networking*, vol. 24, no. 5, pp. 2795-2808, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[6] Tarik Taleb et al., "On Multi-Access Edge Computing: A Survey of the Emerging 5G Network Edge Cloud Architecture and Orchestration," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1657-1681, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[7] Flavio Bonomi et al., "Fog Computing and its Role in the Internet of Things," *Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing*, pp. 13-16, 2012. [CrossRef] [Google Scholar] [Publisher Link]

[8] Justine Sherry et al., "Blindbox: Deep Packet Inspection over Encrypted Traffic," *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*, pp. 213-226, 2015. [CrossRef] [Google Scholar] [Publisher Link]

[9] IETF SCONE Working Group, Standard Communication with Network Elements (SCONE), IETF Working Group Charter, 2024. [Online]. Available: https://datatracker.ietf.org/wg/scone/about/

[10] Pavel Mach, and Zdenek Becvar, "Mobile Edge Computing: A Survey on Architecture and Computation Offloading," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1628-1656, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[11] Michael Seufert et al., "A Survey on Quality of Experience of HTTP Adaptive Streaming," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 1, pp. 469-492, 2015. [CrossRef] [Google Scholar] [Publisher Link]

[12] Yuyi Mao, Jun Zhang, and Khaled B. Letaief, "Dynamic Computation Offloading for Mobile-Edge Computing with Energy Harvesting Devices," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 12, pp. 3590-3605, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[13] Shiqiang Wang et al., "Dynamic Service Migration in Mobile Edge-Clouds," *IFIP Networking Conference (IFIP Networking)*, pp. 1-9, 2015. [CrossRef] [Google Scholar] [Publisher Link]

[14] Shiqiang Wang et al., "Mobility-Induced Service Migration in Mobile Micro-Clouds," *IEEE Military Communications Conference*, pp. 835-840, 2014. [CrossRef] [Google Scholar] [Publisher Link]

[15] M. Reza Rahimi et al., "Mobile Cloud Computing: A Survey, State of Art and Future Directions," *Mobile Networks and Applications*, vol. 19, no. 2, pp. 133-143, 2014. [CrossRef] [Google Scholar] [Publisher Link]

[16] Lele Ma et al., "Efficient Live Migration of Edge Services Leveraging Container Layered Storage," *IEEE Transactions on Mobile Computing*, vol. 18, no. 9, pp. 2020-2033, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[17] Ejaz Ahmed, and Mubashir Husain Rehmani, "Mobile Edge Computing: Opportunities, Solutions, and Challenges," *Future Generation Computer Systems*, vol. 70, pp. 59-63, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[18] Liang Huang, Suzhi Bi, and Ying-Jun Angela Zhang, "Deep Reinforcement Learning for Online Computation Offloading in Wireless Powered Mobile-Edge Computing Networks," *IEEE Transactions on Mobile Computing*, vol. 19, no. 11, pp. 2581-2593, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[19] Chuanpan Zheng et al., "GMAN: A Graph Multi-Attention Network for Traffic Prediction," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 1, pp. 1234-1241, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[20] Jinke Ren et al., "Collaborative Cloud and Edge Computing for Latency Minimization," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 5, pp. 5031-5044, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[21] Bing Yu, Haoteng Yin, and Zhanxing Zhu, "Spatio-Temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Forecasting," *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, pp. 3634-3640, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[22] Aldo Pareja et al., "EvolveGCN: Evolving Graph Convolutional Networks for Dynamic Graphs," *Proceedings of AAAI Conference on Artificial Intelligence*, vol. 34, no. 4, 2020, pp. 5363-5370, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[23] Petar Veličković et al., "Graph Attention Networks," *International Conference on Learning Representations*, pp. 1-12, 2018. [Google Scholar] [Publisher Link]

[24] Tor Lattimore, and Csaba Szepesvári, *Bandit Algorithms*, Cambridge, U.K.: Cambridge University Press, 2020. [Google Scholar] [Publisher Link]

[25] Wei Chu et al., "Contextual Bandits with Linear Payoff Functions," *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pp. 208-214, 2011. [Google Scholar] [Publisher Link]

[26] Lihong Li et al., "A Contextual-Bandit Approach to Personalized News Article Recommendation," *Proceedings of the 19th International Conference on World Wide Web*, pp. 661-670, 2010. [CrossRef] [Google Scholar] [Publisher Link]

[27] R. Alimi et al., "Application-Layer Traffic Optimization (ALTO) Protocol," IETF RFC 7285, 2014. [CrossRef] [Google Scholar] [Publisher Link]

[28] 3GPP TS 23.288 V17.5.0, "Architecture Enhancements for 5G System (5GS) to Support Network Data Analytics Services," 3GPP, 2022. [Online]. Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3579

Appendix 1: Notation Summary

Table 5. Mathematical Notation

Symbol	Definition
U, C, P, M	Sets of users, cells, UPFs, MEC sites
$t, \Delta t$	Epoch index, epoch duration
$D\_\{u,m\}(t)$	End-to-end service delay
$\chi\_\{u,m',m\}(t)$	Migration cost
$R\_u(t)$	Steering reward
$\alpha, \beta, \gamma, \delta, \eta$	Reward weight parameters
$G\_t = (V\_t, E\_t)$	Cross-domain graph at epoch t
W	Temporal window size
$\mu\hat{}k, \sigma\hat{}k$	Predicted mean and std dev for metric k
$p\hat{}I$	Predicted interruption probability
$z\_\{u,m\}(t)$	Context vector for bandit
$M\_u(t)\hat{}safe$	Safe candidate MEC set
$B\_\{u,m\}(t)$	LinUCB exploration bonus
ψ	Migration hysteresis penalty
$\theta\_benefit, \theta\_\sigma$	ESA generation thresholds
$TTL\_\{u,m\}(t)$	Confidence-derived advisory lifetime