Proactive Performance Assurance for 6G Network Service Using Multi-Agent Systems

H. Shakespear-Miles[1], S. Barzegar[1], M. Ruiz[2], and L. Velasco[2*]

[1] *Barcelona Super Computing Center (BSC), Barcelona, Spain*
[2] *Optical Communications Group (GCO), Universitat Politecnica de Catalunya (UPC), Barcelona, Spain*
[*] *luis.velasco@upc.edu*

Abstract: Ensuring consistent Network Service (NS) performance in high-mobility environments is difficult due to the unpredictable nature of traffic and topology changes. This paper proposes a distributed control system that secures end-to-end performance by proactively reconfiguring resources based on mobility forecasting. To address the "cold-start" problem during service handovers, we introduce a knowledge generation framework that utilizes knowledge sharing between pre-trained models. Operating within a Multi-Agent System (MAS), autonomous agents use shared traffic insights to efficiently probe and adapt to unknown network conditions. Our simulations demonstrate that this approach achieves high prediction accuracy, streamlines probing in 3-path networks, and successfully maintains low latency during mobility-driven handovers.

Keywords: Multi-Agent Systems, Deep Reinforcement Learning, Knowledge transfer, Near-real-time control

1. Introduction

Ensuring the performance of Network Services (NS), for example with respect to end-to-end (e2e) delay guarantees, remains a challenging problem, particularly in highly dynamic and variable network conditions. In such contexts, traditional control mechanisms are often insufficient to meet stringent service requirements. To address these limitations, Near-Real-Time (nRT) control of NS has been introduced as a promising approach to maintain and enforce e2e performance levels for demanding services [1]. One viable strategy relies on the adoption of an intelligent, distributed control architecture capable of offloading nRT decision-making from the Software-Defined Networking (SDN) controller. This distributed paradigm can be realized through Multi-Agent Systems (MAS) [2], in which multiple agents, deployed close to network elements, autonomously perform control actions aimed at service assurance [3]. As an illustrative example, autonomous flow routing mechanisms based on Deep Reinforcement Learning (DRL) have been explored in prior work [4], [5]. Within this framework, agent coordination is achieved through a centralized Machine Learning Function Orchestrator (MLFO) [6], which operates as a delegated entity of the Service Orchestrator (SO) for managing the nRT control of NSs.

The focus of this paper is on mobility-aware scenarios involving e2e NSs that span both Radio Access Network (RAN) and transport network domains. In such scenarios, user mobility and fluctuating traffic patterns can significantly affect service performance, making proactive NS reconfiguration essential. While the prediction of when a handover will occur is handled externally (for example, by an xApp running in the nRT RAN Intelligent Controller (RIC)) [7] this paper addresses the reconfiguration procedure that must be executed upon receiving the resulting handover notification some 10s in advance. Without such proactive intervention, the absence of a prepared control configuration could result in service degradation once the handover occurs. To support this objective, we introduce a dedicated procedure designed to ensure that the MAS processing pipeline can be activated and made operational within the nRT control loop constraints. This procedure leverages DRL-based traffic prediction models, which are transferred to other agents and used to probe and evaluate the resources provisioned during the NS reconfiguration phase. The resulting delay measurements used to identify and select the most suitable pretrained DRL model for the operational phase following reconfiguration.

2. NS Reconfiguration Scenario

Figure 1 illustrates the targeted mobility scenario, where distributed MAS-based agents proactively reconfigure the NS and its associated MAS pipeline to ensure service continuity. Figure 1a shows the resources allocated to the NS connecting a drone, currently served via RAN-1, to an application hosted at an edge/metro data center (DC). Specifically, Agent 1, co-located with ingress packet switch S1, distributes traffic across an SDN-provisioned route set (routes P1 and P2 in the Figure 1 example). E2e delay is then measured by Agent 2, co-located with egress packet switch S3, which interfaces with the edge/metro DC. Through dynamic traffic balancing across available routes, the MAS framework regulates the e2e delay ensuring it stays below some maximum d_{max} requirement [4].

As the drone follows a trajectory away from RAN-1 and toward RAN-2, an unprepared handover would reduce connectivity to best-effort service, risking application performance degradation. Figure 1b shows the proactive response: two additional routes (P3 and P4) are allocated, and Agent 3 is instantiated to govern routing between S3 and the edge/metro DC ahead of the handover event. Given advanced notice this process can be completed before the handover event occurs, maintaining service delay requirements smoothly.

979-8-3195-4420-9/26/$31.00 ©2026 IEEE

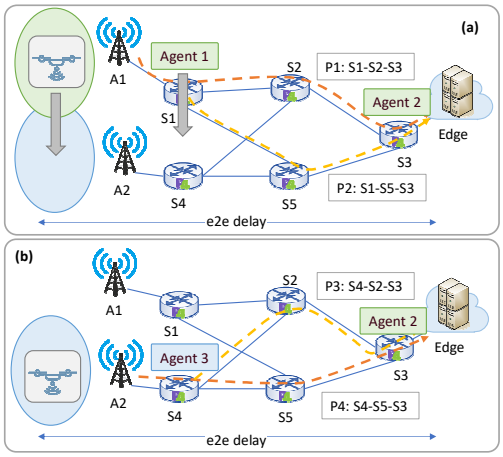

Figure 1 Example handover scenario

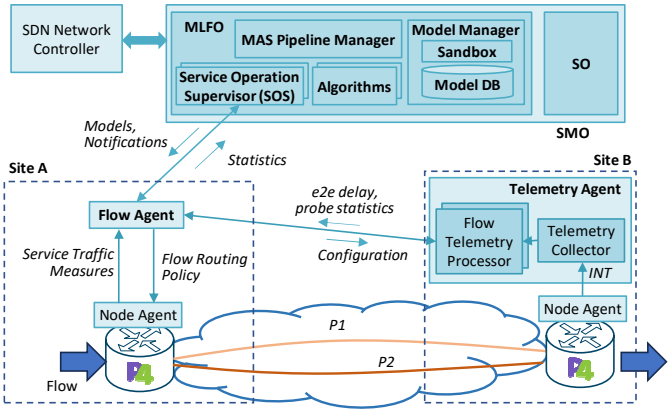

Figure 2 MAS control architecture for NS management

3. MAS Architecture

Figure 2 illustrates the proposed MAS control architecture for nRT NS management. The architecture comprises two principal layers: a Service Management and Orchestration (SMO) layer and a data plane layer.

The SMO is comprised of the SO, which interfaces with the SDN controller and supports resource deployment, and the MLFO, which is responsible for managing the MAS pipeline. The MLFO contains four functional components: the MAS Pipeline Manager, which handles the deployment and configuration of agent pipelines; the Model Manager, which maintains the model repository (Model DB) and pre-trains models in preparation for service deployment; the Service Operation Supervisor (SOS), which supervises NS operation, ensures required performance levels, and is responsible for operational model selection which uses some utility algorithms. Importantly, the MLFO and the MAS control plane do not replace the SDN Network Controller but rather operate alongside it. The SDN controller retains responsibility for resource provisioning and path configuration, while the MAS framework offloads nRT service-level decision-making that would otherwise burden the controller, allowing each component to operate within its appropriate timescale and scope of responsibility.

At the data plane layer, each switch (ingress and egress) is associated with a Node Agent responsible for local control actions. A Flow Agent governs traffic distribution across the available paths, receiving telemetry feedback from a dedicated Telemetry Agent. The Telemetry Agent comprises a Telemetry Collector, which gathers in-band network telemetry (INT) measurements from the data plane, and a Flow Telemetry Processor, which processes and forwards these measurements to the Flow Agent to inform routing decisions.

The MLFO maintains bidirectional communication with both the Flow Agent and the SO, enabling coordinated control across orchestration and data plane layers, closing the nRT control loop while preserving the SDN controller's role in underlying network management.

4. Methodology

To enable seamless NS reconfiguration, inter-agent knowledge transfer is performed between an existing agent and a newly deployed agent assuming NS control. Without this transfer, a new agent would operate without prior knowledge, discarding accumulated experience. To address this, the existing agent shares a traffic model of the NS, trained alongside its routing operations. This allows the incoming agent to characterize expected post-handover traffic conditions before assuming control. As shown in Figure 1, this is exemplified by the transfer from Agent 1 to Agent 3 during NS handover.

Both the traffic prediction and routing models are based on the Twin Delayed Deep Deterministic Policy Gradient (TD3) framework [1]. The traffic prediction model forecasts NS traffic over a future window, with state, action, and reward vectors computed periodically. The state encodes recent traffic up to time t, while the action represents predicted minimum and maximum traffic over the next interval. The reward penalizes prediction error as the negative maximum discrepancy between predicted and observed values, as defined in (1) with notation in Table 1.

$$reward = -n * \max(E_{min}, E_{max}) \quad (1)$$

$$c_m = \sum_{i \in I} c_i * (P_{i,min} + P_{i,max})/i \quad (2)$$

The newly instantiated agent uses the shared traffic model to generate predictions of upcoming NS traffic. These predicted traffic values are used to synthesize probe traffic, which is injected into the previously uncharacterized network environment to obtain e2e delay measurements along the newly provisioned paths. This probing procedure enables the agent to characterize path behavior prior to handover. The resulting delay measurements

are subsequently used as input to the model selection procedure, ensuring that the most suitable pretrained model is identified before the agent assumes operational control of the NS.

For the TD3 routing models, no single model is guaranteed to generalize to the post-handover environment. So, a number of models are pretrained on a variety of conditions in the Sandbox and stored in the model DB. Then, when a new model is required, for example in the case of NS handover, the SOS agent executes Algorithm 1 to identify the most suitable candidate from the available model set. The algorithm takes as input the model set M, a collection of probe scenarios S, and the corresponding delay measurements D received from the probing process. A probe scenario, s, involves sending a specific amount of traffic along each of the paths and observing the resulting e2e delay, d_s.

Each model in M has been trained on specific network topologies with unknown background traffic and subsequently validated on entirely distinct topologies, allowing each model to be characterized by attributes such as delay behavior and routing policies. Following initialization (line 1), the algorithm evaluates the delay recorded under each probe scenario against d_{max}. Where the maximum delay requirement is satisfied, models capable of operating under that scenario are identified as candidates and ranked by their distance to the measured delay (lines 2–5). Models in the data base are additionally, characterized by the operating cost, shown in Equation 2. Which is a weighted sum of the cost of each path (given by the SDN controller) and the amount of traffic being sent through that path. If candidate models exist, the one with the lowest operating cost is selected (lines 6–8); if none qualify, the algorithm returns no selection (line 9), triggering a notification to the SO via the MLFO to provision additional NS resources.

Table 1 General Notation

M	Set of models, each characterized by $<d_{max}$, P^i_{max}, P^i_{min} for every path considered by the model}, where P^i_{max}, P^i_{min} are the percentages of traffic sent through P^i.
S	Set of probe scenarios, index s. Each scenario is characterized by $\{P^i_{(\cdot)}\}$
D	Set of delay measurements, $D=\{d_s, \forall s \in S\}$.
n	maximum traffic of the flow.
$Tr_{(\cdot)}$	Actual min and max traffic for the window.
$Pr_{(\cdot)}$	Predicted min and max traffic for the window.
$E_{(\cdot)}$	Absolute prediction error; abs($Tr_{(\cdot)}$ - $Pr_{(\cdot)}$)

Algorithm 1 Agent model selection process

INPUT: M, S, D OUTPUT: m

1: $C \leftarrow \{\}$
2: **for** each $s \in S$ **do**
3: **if** $d_s < d_{max}$ **then**
4: $M_s \leftarrow \{<m, abs(d_s – d_m)>$ for each $m \in M(s)\}$
5: $C \leftarrow C \cup M_s$
6: **if** $|C| > 0$ **then**
7: $m \leftarrow$ select_lowest_cost(C)
8: **return** m
9: **return** none

5. Illustrative Results

A Python-based ad-hoc simulator was used throughout all stages of model development and system performance evaluation, covering training, validation, probe testing, and handover assessment. The topology shown in Figure 1a was used for training to generate the model set M by varying conditions to train distinct models that satisfy a d_{max} requirement of 0.5ms. Validation was conducted on a distinct and more complex topology, in which paths P3 and P4 each comprise three hops. Independent background traffic flows, unknown to the models, were introduced to load the links, simulating realistic network conditions. This separation between training and evaluation topologies is crucial, as it assesses the generalization capability of the pretrained models to previously unseen network conditions.

Figure 3 presents the behavioral ranges of the pretrained models in terms of the percentage of NS traffic routed through P1. Here models are trained to operate over two routes, and the set of eight models collectively covers the full range of possible routing distributions, ensuring that any operating point can be served by at least one candidate model. Figure 4 shows the performance of the traffic prediction model over a 24-hour evaluation period, with estimates generated every 10 minutes for a forward window of 10 minutes. The predicted minimum and maximum traffic bounds closely track the actual traffic throughout the evaluation period, with a consistently narrow margin between predicted and measured values. This level of accuracy is sufficient to reliably characterize the expected NS traffic conditions during the probing procedure.

To evaluate the model selection process, the NS traffic model was used to generate traffic predictions, which were then applied to probe new paths across two distinct evaluation topologies. These topologies differ from those used during training and validation, while remaining structurally similar to Figure 1b, featuring two routing paths. The normalized costs associated with the paths in the testing scenarios are [0.3, 0.7] for P1 and P2, respectively. From this topology two testing scenarios could be further defined to assess the model selection process under varying network conditions. In Scenario A, the available capacities for P1 and P2 are [99 Gb/s, 15 Gb/s], respectively, whereas in Scenario B, the capacities are [25 Gb/s, 99 Gb/s]. Crucially, these capacities are not known to the agent performing the selection. Instead, the agent must infer the most suitable model based solely on the observed outcomes of the probing process. For a topology with 2 paths, a total of 5 distinct probe scenarios are

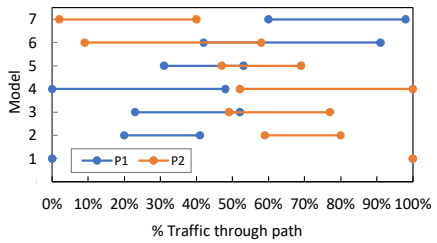

Figure 3 Model behavior ranges for % of Traffic sent to P1.

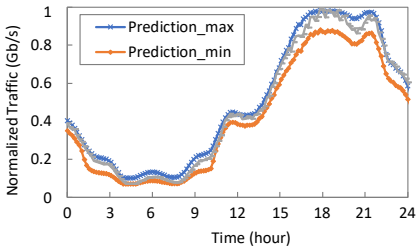

Figure 4 Predicted and actual traffic

Table 2 Two path probe scenarios

Probe	P1 %	P2%
1	100	0
2	0	100
3	50	50
4	70	30
5	30	70

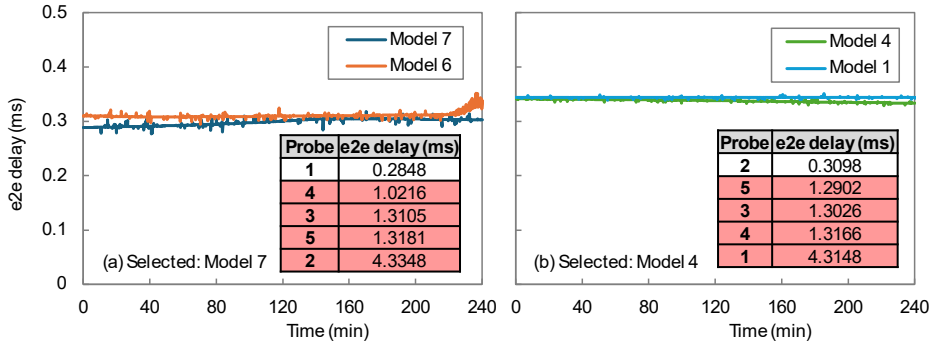

Figure 5 e2e delay produced by Scenario A for selected (Model 7) and candidate (Model 6) (a), and delay produced by Scenario B for selected (Model 4) and candidate (Model 1) (b)

employed to gather information about the network characteristics, as outlined in Table 2. With a traffic prediction of 95 Gb/s, both unknown topologies were probed, producing the performance statistics presented in the inset table of Figure 5a and Figure 5b for Scenarios A and B respectively. Across both scenarios, it can be observed that only one probe scenario can satisfy the d_{max} requirement. This is consistent with the available path capacities, as Scenario A strongly favours routing traffic through P1, while Scenario B favours routing through P2.

Based on the probe information, Algorithm 1 selects Model 7 in Scenario A, which routes most traffic through P1. The resulting delays for Model 7 and the next-best candidate, Model 6, are shown in Figure 5a. Both models satisfy the d_{max} constraint. However, Model 7 consistently achieves lower e2e delay. In contrast, Model 6 exhibits a late stage increase in delay. These results confirm that Algorithm 1 selects the most effective model in this scenario. In Scenario B, Algorithm 1 selects Model 4, which routes most traffic through P2. Model 1 is also identified as a candidate due to a similar routing preference. However, Model 4 achieves a lower operating cost and is therefore selected. As shown in Figure 5b, Model 4 also yields slightly lower e2e delay compared to Model 1, further validating the selection algorithm.

6. Conclusions

This paper presented a MAS-based framework for near-real-time NS reconfiguration in mobility-aware scenarios, targeting e2e delay assurance under dynamic network conditions. By distributing routing control across a set of agents, supervised by an MLFO, scalable and timely adaptation to NS changes can be made. A key process being knowledge transfer between agents, which allows newly instantiated agents to leverage pretrained TD3-based traffic prediction models to characterize post-handover conditions prior to control activation. This enables proactive probing of newly provisioned resources and supports informed routing model selection using observed delay behavior. The results demonstrate that the proposed model selection procedure can reliably identify suitable pretrained routing models under varying network conditions. Across both evaluated scenarios, the selected models consistently satisfy delay constraints while achieving improved performance relative to competing candidates, confirming the effectiveness of the proposed probing and selection approach for seamless NS reconfiguration.

Acknowledgements

The research leading to these results has received funding from the European Union's Horizon Europe research and innovation programme under G.A. No. 101092766 (ALLEGRO), the project PID2024-157824OB-I00 funded by MICIU/ AEI/10.13039/501100011033 /FEDER, UE and from ICREA.

References

[1] S. Barzegar *et al*. "Packet Flow Capacity Autonomous Operation Based on Reinforcement Learning," Sensors, 2021.
[2] A. Dorri *et al*., "Multi-agent systems: A survey," IEEE Access, 2018.
[3] L. Velasco *et al*., "Autonomous and energy efficient lightpath operation based on digital subcarrier multiplexing," IEEE JSAC, 2021.
[4] S. Barzegar *et al*., "Autonomous Flow Routing for Near Real-Time Quality of Service Assurance," IEEE TNSM, 2024.
[5] H. Shakespear-Miles *et al*. "Intelligent Reconfiguration of Distributed Control of 6G Network Services under Mobility Scenarios" OECC/PSC, 2025
[6] P. Gonzalez *et al*., "Deployment of Secure ML Pipelines for Near-Real-Time Control of 6G Network Services," in Proc. OFC, 2024.
[7] T. Rodrigues *et al*., "Smart Handover with Predicted User Behavior using Convolutional NN for WiGig Systems," IEEE Network, 2024.