Distributed and Autonomous Flow Routing Based on Deep Reinforcement Learning

Sima Barzegar, Marc Ruiz, and Luis Velasco*
Optical Communications Group (GCO), Universitat Politècnica de Catalunya (UPC), Barcelona, Spain
e-mail: luis.velasco@upc.edu

***Abstract*:** *A DRL approach with a specific reward function is proposed for autonomous flow routing operating on multilayer scenarios. Illustrative results reveal that the DRL achieves optimal flow routing in terms of cost and service quality.*
Keywords: Distributed Control, Autonomous systems, Deep Reinforcement Learning

I. INTRODUCTION

Transport networks are becoming more complex, mainly to support the large traffic dynamicity and stringent performance needed to support beyond 5G and 6G services. In fact, such support requires increased levels of flexibility and automation, together with higher priority given to network optimization, security, energy consumption, and cost efficiency. As a result, such infrastructures need to apply Artificial Intelligence (AI) / Machine Learning (ML) techniques [1] to create an automated management to implement data-driven closed control loops.

To achieve autonomic networking, Software defined Networking (SDN) control is being augmented with instantaneous data-driven decision-making [2]. This approach is beneficial for many applications that do not required making decisions near real time, like failure management. However, in the case of dynamic traffic conditions, centralized decision making leads to poor resource utilization and high energy consumption. In addition, ML algorithms might be executed as close as possible to the data sources (contrarily to the centralized architecture of SDN) looking at minimizing the amount of data to be conveyed, as well as minimizing the response time. Examples, include the use of Deep Reinforcement Learning (DRL) [3] for the management of the capacity at packet [4] or optical connections (*lightpath*) [5] in real time, where the capacity of the connections is managed near real-time to adapt to input traffic. Note that packet and optical layers are closely related in multi-layer networks, where links connecting packet nodes are supported by lightpaths.

In this paper, we propose a distributed decision-making following the concept of Multi-Agent Systems (MAS) [6]. MAS is a subfield of AI and it can be defined as a set of individual agents that share knowledge and communicate with each other in order to solve a problem that is beyond the scope of a single agent. In the scope of networking, we proposed that agent nodes make autonomous decisions real-time based on guidelines received from the SDN controller. Such decision-making will be performed based on its own observed data, as well as on the data and models received from other nodes. To illustrate this concept, in this paper we focus on flow routing. We assume packet flows carrying traffic with unknown characteristics entering in a packet node. Such flows can be routed to the destination though several output interfaces, while ensuring some given Quality of Service (QoS), e.g., in term of end-to-end delay, and optimizing resource utilization, since the different interfaces represent routes with different performance to the final destination.

II. DISTRIBUTED AUTONOMOUS FLOW ROUTING

Fig. 1 sketches the proposed distributed intelligence architecture, where a single node and the centralized SDN controller are represented. Note that in such architecture, we are moving the intelligence from the centralized control plane to the nodes thus resulting in a hybrid centralized SDN control with distributed network intelligence. Agent nodes are able to communicate with each other to exchange data and models for the sake of coordination. Decision-making is performed by every individual agent near real-time (sub-second to few second granularity) based on its own observed data, as well as on the data and models received from other agents. The control plane is responsible for the general coordination of the network, being in charge of generating the needed guidelines for the agents that leave the desired degree of freedom for their autonomous operation.

To illustrate this concept, let us focus on the packet layer; packets in a flow follow the route that has been decided from the SDN controller. Route computation is based on the network topology and the network conditions at the computation time and it does not usually change until some event occurs, even though network conditions change. In our approach, the SDN controller gives degrees of freedom to the packet layer by computing a set of routes (including a single one) for every flow that are given to the node agents as guidelines (together with some other parameters), so the actual route is decided by the node agents based on AI/ML models and observed data (e.g., end-to-end delay) and it might be changed near real time. In addition, some other specific responsibilities that are hard to implement in a distributed manner, like multilayer issues, failure management, etc., remain in the control plane. This will relieve the control plane from near real time operation and now focus on long-term activities.

Fig. 2 represents an example of flow routing operation in a node in the targeted multilayer scenario. The packet node agent has received from the SDN controller three possible routes for a given traffic flow and it has to decide which one or combination of several are the allow to reach some QoS performance (in this case, the end-to-end delay), while

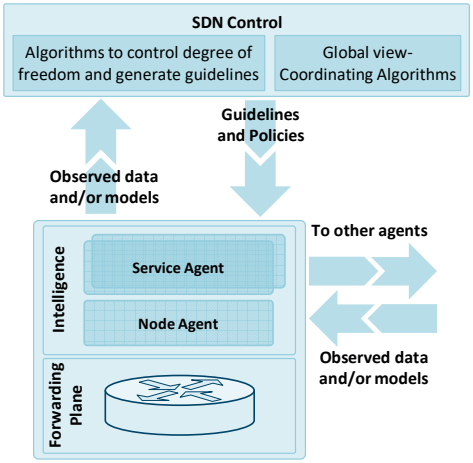

Fig. 1 Proposed distributed intelligence.

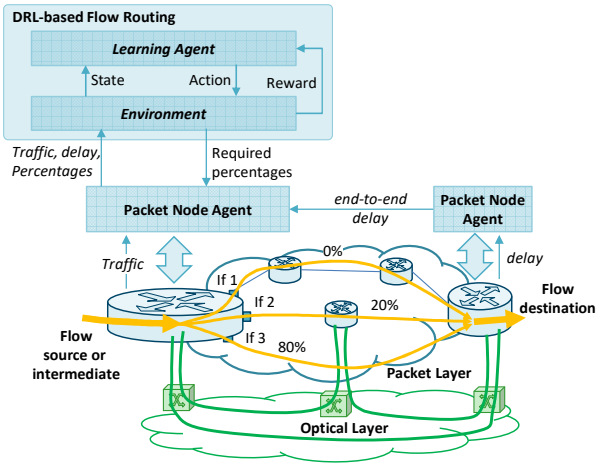

Fig. 2 Example of distributed flow routing based on DRL.

minimizing some cost function. In this case, we use DRL to implement a service agent for the traffic flow, which is represented in Fig. 2 with two inter-related blocks, named *learning agent* and *environment*; the former is in charge of learning the best actions to be taken based on the current state and the received reward, whereas the latter is in charge of computing the state based on the observed traffic and current combination of routes (percentages), as well the obtained reward based on the measured end-to-end delay for the selected routes. In such figure, the traffic of the flow is routed through two different interfaces (in this case both supported by the optical layer) among the three available ones for the flow. Here, we assume that the traffic flow actually consists of multiple sub-flows, which are routed independently so the packets belonging to each sub-flow follow the same route. In the destination, the end-to-end delay is measured (e.g., using in-band telemetry) and some statistics are computed (e.g., maximum and weighted average) and sent to the node agents participating in the routing of the flow.

Finally, just to highlight that in this example, the intelligence of the node agent is very limited as it just collects monitoring data from the forwarding plane and from the destination of the flow. However, it might be in charge of imposing some constraints among the different flows and other node agents in the network to reach some global optimal that require considering some additional constraints.

III. DRL FOR AUTONOMOUS FLOW ROUTING

In this section, we propose a new reward function specifically designed for autonomous flow routing based on DRL. We started from the lessons learned from our previous work in [4], where we applied Twin Delayed Deep Deterministic Policy Gradients (TD3) [3]. TD3 is an off-policy DRL algorithm that uses a pair of *Critic* networks and an *Actor* network that is updated with some delay.

Let n_s be the number of continuous states in the DRL algorithm. We have designed our DRL algorithm to deal with several flows, so every state represents one of the flows. We define the state as the ratio traffic over the capacity of the interfaces. In addition, let n_a be the number of continuous actions, where each action is related to one flow and output interfaces and represents the percentage of flow to be sent through those interfaces. As an example, for one single flow that can be routed through 3 different interfaces, action [50, 20, 30] entails 50% of traffic flow through the first interface, 20% through the second one, and 30% through the third one. In our approach, the computation of states and actions is performed periodically (e.g., every second).

A generic reward function has been defined with the objective of penalizing the actions causing that some target delay is exceeded and/or increasing the cost of network. In consequence, two reward components have been considered, to account for the obtained delay (r_{delay}) and for the cost (r_{cost}), where the final reward is defined as follows; parameters α_{delay} and α_{cost} represent the proportion of each of them.

$$r(t) = \alpha_{delay} \cdot r_{delay}(t) + \alpha_{cost} \cdot r_{cost}(t) \quad (1)$$

Assuming a given maximum delay to be ensured for the flow (denoted D_{max}), the reward related to the obtained delay can be defined as follows, where β is a fixed penalty for violating the maximum delay.

$$r_{delay} = \{-\beta - d/D_{max} \; if \; d > D_{max} \; 0 \; otherwise \quad (2)$$

Finally, the reward related to the cost of using the output interfaces is related to the percentage of traffic sent through each of them, as well as to the ratio cost capacity of the interface.

$$r_{cost} = -traffic \cdot \sum_i \; percentage_i \cdot \frac{cost(i)}{capacity(i)} \quad (3)$$

We assume that the capacity of the interfaces, as well as the cost of each interface (which is related to the route to the destination for the flow), D_{max} and β for each flow have been received from the SDN controller.

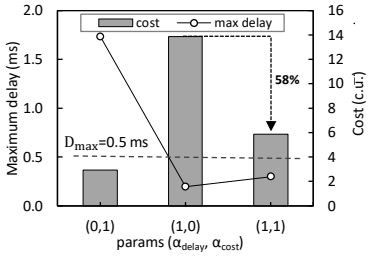

Fig. 3 Overall DRL performance

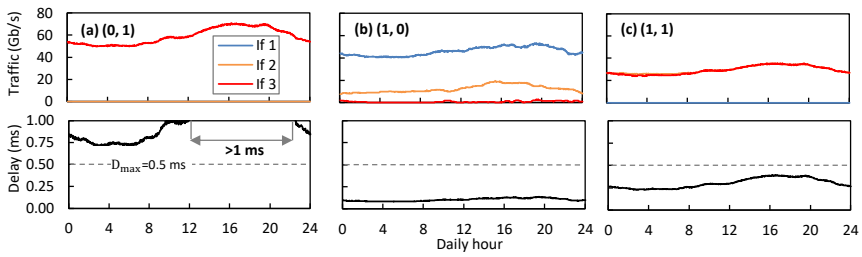

Fig. 4 Detailed performance for configuration (0,1) (a), (1,0) (b), and (1,1) (c)

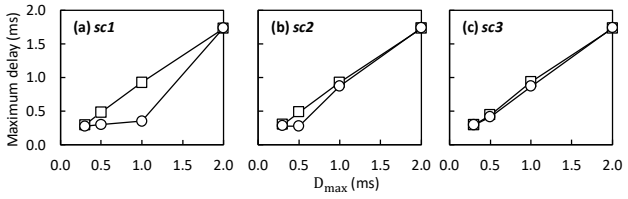

Fig. 5 Delay performance under different scenarios

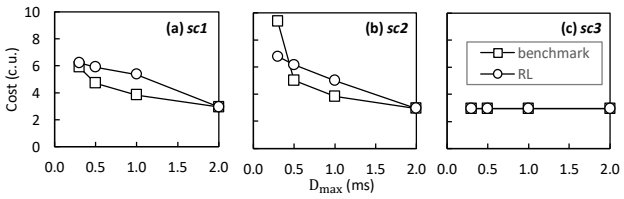

Fig. 6 Cost under different scenarios

IV. RESULTS AND DISCUSSION

For evaluation purposes, a Python-based simulator reproducing the topology in Fig. 2 was implemented and realistic traffic flow behavior was accurately emulated following a similar configuration as in [4]. As in Fig. 2, we assume that *if1*, *if2*, and *if3* follow different routes in the multilayer network, so that three different delay behaviors are emulated. The cost of each interface was configured inversely proportional to the expected end-to-end delay through that interface.

Let us first study the performance of the reward function defined in the previous section. To this aim, we fix β=6, D_{max} = 0.5 ms, and compared three different optimization targets by defining different configurations of the tuple (α_{delay}, α_{cost}), namely: (0,1) (cost minimization), (1,0) (delay assurance), and (1,1) (multi-objective). Fig. 3 illustrates the overall performance under all three cases in terms of maximum delay and cost. It is worth noting that the DRL learned a model producing stable and good-rewarded actions in all the scenarios after 5000 episodes. As seen in Fig. 3, the first configuration (0,1) achieves the expected best solution in terms of cost, at the expense of having a large maximum delay.

In addition, Fig. 4 shows the detail of the traffic routed to every interface and the delay in one day after DRL converged to an accurate model. Fig. 4a shows that traffic is fully routed through the cheapest *if1*, as well as delay always exceed D_{max}, which is unacceptable in terms of QoS. On the opposite, configuration (1,0) finds a different solution, i.e., maximum delay is clearly below D_{max}, while remarkably increasing network cost. This increment is due to the use of the most expensive but delay efficient *if1*, while reducing the traffic thought the other interfaces (Fig. 4b). However, due to the nature of the penalization of maximum delay violation (eq. (2)), delay is always kept too far from the limit. Interestingly, Fig. 3 shows that configuration (1,1) achieves the best performance, since maximum delay is below D_{max} and network cost is reduced up to 58% to that of configuration (1,0). The detailed analysis in Fig. 4c allowed us concluding that DRL learned to evenly balance the routing between *if2* and *if3*, which produces a delay closer but always below D_{max} and achieves cost reduction by avoiding using expensive *if1*. Therefore, the proposed DRL is able to converge to different solutions in order to achieve heterogeneous target optimization criteria.

Fixing configuration (1,1), we conducted a second numerical study to evaluate different scenarios for D_{max} ranging from 0.3 to 2 ms. Three network scenarios were considered: *i*) using the same configuration of virtual and optical routes and costs as in the previous results (*sc1*); *ii*) configuring all interfaces with parallel and equal routes but keeping those differentiated costs in *sc1* (*sc2*); and *iii*) configuring the routes in *sc1* but fixing the same cost for all (*sc3*). Fig. 5 and Fig. 6 shows the maximum delay and cost, respectively, for all the scenarios and D_{max} values. For benchmarking purposes, we plot the performance of an *a posteriori* method consisting in assigning the percentage of traffic routed to every interface that achieves maximum delay as close as possible to D_{max} with the minimum cost. Although this method cannot be applied in real operation since it assumes perfect knowledge of long-term traffic evolution, it serves as reference for DRL performance evaluation purposes. As it can be observed from both Fig. 5 and Fig. 6, DRL closely matches the performance of such *a posteriori* benchmarking method for all the tested scenarios. Although it has some room for improvement for moderated QoS requirements, it is worth noting that DRL equals and even slightly improves (*sc2*) the performance of *a posteriori* method when D_{max} is very low. This is a very meaningful result, since it anticipates outstanding performance of DRL-based autonomous routing in the presence of those beyond 5G / 6G services targeting stringent QoS requirements.

To conclude, this paper proposed a distributed and autonomous flow routing based on DRL with a specific reward function. Results show that our approach can beat a posteriori method with perfect knowledge of long-term traffic evolution in both cost and QoS.

ACKNOWLEDGMENT

The research leading to these results has received funding from the H2020 B5G-OPEN (G.A. 101016663), the MINECO-NextGenerationEU TIMING (TSI-063000-2021-145), the MICINN IBON (PID2020-114135RB-I00), and the ICREA Institution.

REFERENCES

[1] D. Rafique and L. Velasco, "Machine Learning for Optical Network Automation: Overview, Architecture and Applications," (Invited Tutorial) IEEE/OSA Journal of Optical Communications and Networking (JOCN), 2018.

[2] L. Velasco *et al.*, "Monitoring and Data Analytics for Optical Networking: Benefits, Architectures, and Use Cases," IEEE Network Magazine, 2019.

[3] V. Francois-Lavet *et al*., "An Introduction to Deep Reinforcement Learning," Foundations and Trends in Machine Learning, 2018.

[4] S. Barzegar, M. Ruiz, and L. Velasco, "Packet Flow Capacity Autonomous Operation based on Reinforcement Learning," MDPI Sensors, vol. 21, pp. 8306, 2021.

[5] L. Velasco *et al.*, "Autonomous and Energy Efficient Lightpath Operation based on Digital Subcarrier Multiplexing," IEEE Journal on Selected Areas in Communications, vol. 39, pp. 2864-2877, 2021.

[6] M. Wooldridge, *An introduction to multiagent systems*, John Wiley & Sons, 2009.