Hierarchical Deep Reinforcement Learning-based Load Balancing Algorithm for Multi-domain Software-Defined Networks (draft)

Robert Kołakowski*◇ [0000-0002-5451-8847], Sławomir Kukliński◇, Lechosław Tomaszewski* [0000-0003-3836-7900]
*Orange Polska, ◇Warsaw University of Technology

Abstract

Software Defined Networking (SDN) is a well-established networking paradigm that enables granular network control and optimisation via Traffic Engineering (TE). A promising approach to SDN TE is to use centralised Deep Reinforcement Learning (DRL) enabling automated operation and optimisation both short and long-term. Despite excellent performance, the centralised DRL suffers from scalability and convergence issues, limiting its applicability. On the other hand, DRL exploitation in a multi-domain SDN environment is not well explored yet despite several benefits coming from operations distribution, such as better scalability or reduced impact of latency on Data Plane metrics collection. This paper presents the DRL-based routing approach targeting load balancing in a hierarchical multi-controller SDN. The concept yields network capacity gains over conventional routing methods. Apart from the improved scalability, the approach facilitates application in hybrid network deployments with limited interaction and visibility of domains' internals due to used abstractions of topology, metrics and path operations.

Index Terms

6G, SDN, AI, DRL, User Plane, Data Plane, Control Plane, traffic engineering, load balancing, multi-agent system, DDPG, multi-domain routing

Acronyms

The following acronyms are used in this manuscript:

BN	Border Node
CP	Control Plane
DDPG	Deep Deterministic Policy Gradient
DDQN	Double Deep Q-Network
DLBA	Domain Load Balancing Agent
DP	Data Plane
DRL	Deep Reinforcement Learning
E2E	End-to-End
GLBA	Global Load Balancing Agent
GMC	Global Metrics Calculator
GSDNC	Global SDN Controller
HDRL-LB	Hierarchical Deep Reinforcement Learning Load Balancer
ILP	Integer Linear Programming
MC	Metrics Calculator
MDP	Markov Decision Process
OSPF	Open Shortest Path First
QoS	Quality of Service
RL	Reinforcement Learning
SDN	Software-Defined Network
SDNC	SDN Controller
TCP	Transport Control Protocol
TE	Traffic Engineering
UDP	User Datagram Protocol
UP	User Plane
WD	Weighted Dijkstra

I. Introduction

6G networks are commonly assumed to embed self-optimisation mechanisms in the future [1]. Software-Defined Network (SDN) is a well-established networking paradigm foreseen to play an important role in this context. The separation of Control Plane (CP) and Data Plane (DP) facilitates network traffic control and optimisation by Traffic Engineering (TE) applications. The classical SDN concept, due to centralisation, raises scalability issues in large networks. To solve this problem, distributed

ETHER project has received funding from the Smart Networks and Services Joint Undertaking (SNS JU) under the European Union's Horizon Europe research and innovation programme under Grant Agreement No. 101096526. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union. Neither the European Union nor the granting authority can be held responsible for them.

SDN architectures are often adopted [2], which complicate End-to-End (E2E) TE. The conventional TE methods, *e.g.*, Open Shortest Path First (OSPF) [3], are not well suited for long-term optimisation, underperform in complex environments with rapid traffic fluctuations and lack the visibility of E2E network dynamics. This issue will be aggravated by the multi-domain character of future mobile User Plane (UP) and needed support for high mobility scenarios over the edge environments (involving user and application mobility). A promising TE approach is to use Deep Reinforcement Learning (DRL), which dynamically adapts to variable conditions and is able to exploit complex environment properties.

Today, very few concepts combine DRL and distributed SDN to provide local and E2E optimisation simultaneously. Usually, a centralised model is adopted, which is unsuitable for carrier-grade networks due to poor scalability [4]. This paper proposes a novel multi-agent Deep Deterministic Policy Gradient (DDPG)-based routing algorithm, called Hierarchical Deep Reinforcement Learning Load Balancer (HDRL-LB), improving load distribution and total network capacity at both domain and global levels in a hierarchical multi-domain SDN. To reduce CP operations and SDN Controllers (SDNCs) load, the algorithm does not involve rerouting. HDRL-LB introduces abstractions of topology, metrics and CP operations to support hybrid environments. The evaluation showed over 10% improvement of throughput compared to the baseline state-of-the-art methods and fast policy convergence.

II. Related work

One of the envisioned evolution directions of the mobile network is moving from the centralised model towards the distributed one, featuring heterogeneous ecosystems and requiring high granularity of control over the UP traffic to provision E2E Quality of Service (QoS) [1]. TE will need to consider the current network state and imperceptible properties manifesting, *e.g.*, in different traffic peaks seasonality, traffic types share, mobility patterns, *etc.* The DRL is promising in this context due to its ability to adapt to the environment traits by using the action-reward mechanism. The academia proposed several Reinforcement Learning (RL)/DRL frameworks for centralised SDN featuring automatic routing [5], [6] or load-balancing [7] to improve throughput and delay. The solutions, however, have not been tested in multi-domain SDN.

A routing concept for hierarchical multi-controller SDN has been proposed in [8]. The SDNCs cooperate to find weighted shortest paths at the domain and global levels to avoid congestion. The collaborative multi-domain routing framework exploiting Integer Linear Programming (ILP) has been proposed in[9]. It ensures delay and bandwidth and maximises network usage. The DisTE approach provides max-min fair bandwidth allocation for flows and maximises resource usage in a multi-domain SDN[10]. The domains' synchronisation mechanism mitigates the selfish SDNCs' behaviour to obtain a consistent policy. The multi-agent cross-domain routing framework has been proposed in [11], which uses prediction to improve measurement reliability and DRL agents' performance.

Whereas TE in SDN is a well-known problem, the E2E optimisation of multi-domain SDN is not well addressed yet. Also, the scalability of SDNC operations is usually neglected. Hereby, we propose a scalable DRL-based TE algorithm, which can be efficiently used in hierarchical multi-domain SDN setups.

III. Hierarchical Deep Reinforcement Learning Load Balancer (HDRL-LB)

A. Concept description

The key issues regarding wide-scale SDN implementation are SDN scalability and E2E TE. Distribution of control, while offloading each SDNCs and increasing scalability, increases the complexity of E2E TE. First, the optimisation is performed internally within each domain without a global view of its impact on the whole network. This can lead to a load imbalance on the inter-domain links and potentially to congestion and QoS degradation in the neighbouring domains. The uncoordinated approach to multi-domain TE can also lead to traffic imbalance in the network domains and inefficient resource usage. Second, reaching optimal local states does not imply achieving the global optimum. Therefore, to achieve E2E network optimisation in multi-domain SDN, there is a need to combine local and global TE to i) continuously search for a global optimum; ii) enable exploration of local search spaces; and iii) provide global optimisation without disruption of local operations.

The HDRL-LB algorithm addresses the above-mentioned issues. We adopt a multi-domain hierarchical SDN architecture (cf. Figure 1) composed of i) Global SDN Controller (GSDNC) – a global network control entity that handles inter-domain operations (routing, metrics collection, E2E path enforcement by delegating path creation to respective SDNCs) using the network graph abstractions (cf. Section III-B); ii) domain SDNCs responsible for intra-domain operations; iii) agents responsible for the global- and domain-level optimisations, *i.e.*, Global Load Balancing Agent (GLBA) and Domain Load Balancing Agents (DLBAs), respectively. The latter ones are responsible for the periodic calculation of routing graphs(cf. Sec. III-C), which are used by SDNC/GSDNC for routing and path enforcement (using SDN CP). Also, to stabilise the network and enable domain-level routing adaptation, the global routing graph is updated much less frequently than domain ones.

The components jointly provide the E2E routing and E2E TE targeting improvement of network load distribution and throughput (cf. Sec. III-C). To improve scalability, GSDNC sees only the abstracted view of the network composed of Border Nodes (BNs) – nodes connected to data sources/sinks or terminating inter-domain links, hosts, inter-domain links, and abstracted links – the connections between BNs pairs belonging to the same domain (cf. Figure 1). Also, to reduce the number of CP operations and offload SDNCs, the domain and overlay routing graphs do not affect the already routed flows, only the new

ones. This approach, combined with the ability to distribute the E2E routing across multiple SDNCs, contributes to the SDN scalability.

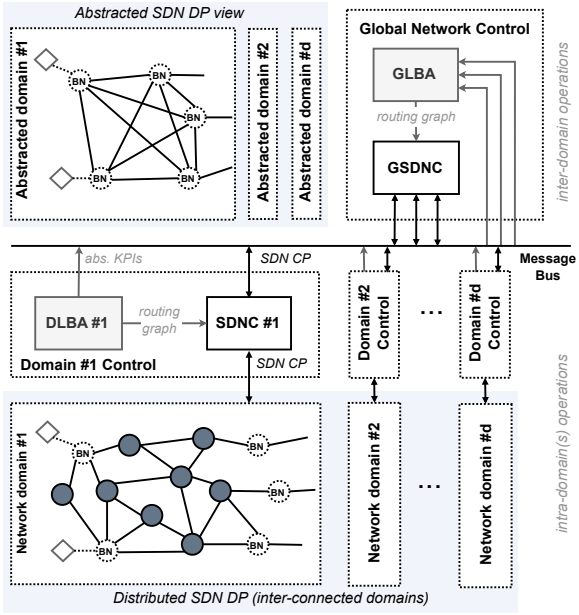

Fig. 1. HDRL-LB entities, interactions and DP views, *i.e.*, network switches (circles) and hosts (rectangles), on each hierarchy level

When a new flow arrives in the network, SDNC identifies the routing case. If the target lies within domain boundaries, SDNC performs intra-domain routing. Otherwise, the flow metadata are forwarded to GSDNC, which performs the inter-domain routing. The output E2E path, in this case, is composed of BNs between the flow's source and target. Next, the path is split based on the BNs domain membership and enforced by respective SDNCs (performing intra-domain routing for the node pairs). In both cases, SDNCs and GSDNC compute the shortest paths using the Dijkstra algorithm on routing graphs with edge weights computed by agents.

To enable dynamic and autonomous UP optimisation, HDRL-LB uses modified DDPG [12], which uses Double Deep Q-Network (DDQN)-based [13] critic to mitigate initial over-optimism [14] and improve policy convergence. Its sample efficiency also contributes to SDN scalability by enabling less frequent DP sampling. In HDRL-LB, DLBAs exchanges with GLBA abstracted metrics. It supports the operation in hybrid multi-provider SDN environments, implementing different TE mechanisms within the domains. Hereby, we consider full operator's control over the underlying domains to assess the full HDRL-LB benefits. Finally, to leverage quasi-periodic demand peaks in the UP (*i.e.*, busy hours) [15], we add time to the environment's state vector.

B. Network model

We consider the multi-domain network supervised by the centralised entity with a limited view of the domains' internals (cf. Section III-A). The topology is represented by an undirected graph $G(V, E)$, where V is a set of vertices and E is a set of edges. Hosts H (connected to vertices v of V) constitute traffic sources and sinks. We consider two network views: a global (abstracted) and domain one, denoted as $G^g(V^g, E^g)$ and $G^d(V^d, E^d)$ respectively ($d \in D$, where D is a set of domains). Domain switches combine a set $V^d = [v_1^d, v_2^d, ..., v_n^d]$ and edges a set $E^d = [e_{12}^d, e_{13}^d, ..., e_{ij}^d]$, where d denotes the node's domain and i, j the vertices v_i, v_j ($V^g = [v_1^g, v_2^g ..., v_n^g]$ and $E^g = [e_{12}^g, e_{13}^g, ..., e_{ij}^g]$ for the global graph). Domain hosts constitute a set $H^d = [h_1^d, ..., h_m^d]$, where m is their number (all hosts in H^g case). The link connecting h_m^d with a switch has capacity $\rfloor_m^d$ and bandwidth $\lfloor_m^d$. All hosts are visible from the global view, *i.e.*, $H^d \subset H^g$. In terms of vertices, the global network view is limited to the domain gateways (switches connected to hosts or inter-domain links), *i.e.*, $V^g \subset V$, where $v_i^d \in V^g \; iff \; \exists_{v_j^d \in N(v_i)} v_j^d \in H^d \; \vee \; d_{v_i} \neq d_{v_j}$. The components at the global level see the abstracted edges (cf. Figure 1). Each link e_{ij} has capacity c_{ij}, and handles aggregate traffic b_{ij} resulting in utilisation u_{ij} (eq. 1).

$$u_{ij} = \frac{b_{ij}}{c_{ij}} \tag{1}$$

For each domain, a set $U^d = [u_{12}^d, u_{13}^d, ...u_{ij}^d]$ describing the utilisation of the links E^d is defined ($U^g = [u_{12}^g, u_{13}^g, ...u_{ij}^g]$ for the global graph links E^g). The capacity of abstracted links observed by GLBA is calculated using f_{max} function (*e.g.*, by using the Ford–Fulkerson algorithm) as shown in eq. 2.

$$c_{ij}^{g} = \begin{cases} c_{ij} & iff\ d_{v_i} \neq d_{v_j} \\ f_{max}(v_i, v_j) & iff\ d_{v_i} = d_{v_j} \end{cases} \tag{2}$$

The utilisation u_{ij}^{g} of the abstracted edge e_{ij}^{g}, is calculated by subtracting the current capacity under traffic (c^{gb}) from the nominal one (c^{g}) and normalisation, *i.e.*, $u_{ij}^{g} = (c_{ij}^{g} - c_{ij}^{gb})/c_{ij}^{g}$. Each traffic flow is described by a tuple $f = (b, t, h_{src}, h_{dst})$, containing consumed bandwidth b (variable in time), flow duration t, source h_{src} and destination nodes h_{dst}.

C. Algorithm principles

HDRL-LB aims to solve the dynamic flow allocation problem to improve load distribution across the network links and domains and increase aggregate global and domain-level throughput. The optimised metrics are load balancing factor (eq. 3) and total host throughput (eq. 4).

$$\varphi = \sum_{\substack{i=1 \\ i \neq j}}^{n} \sum_{j=1}^{n} |u_{ij} - avg(U)| \tag{3}$$

$$\Gamma = \sum_{i=1}^{m} \lfloor_i \tag{4}$$

The HDRL-LB target is the combination of conflicting goals for domain and global level entities (eq. 5 and 6).

$$min(\varphi^{d}) \quad max(\Gamma^{d}) \tag{5}$$

$$min(\varphi^{g}) \quad max(\Gamma^{g}) \tag{6}$$

We model the multi-domain hierarchical SDN network as the combination of standard RL settings (*i.e.*, a set of stochastic domain environments constituting a stochastic global environment) and model each one as Markov Decision Process (MDP) represented by tuples $\mathcal{M} = (\mathcal{S}; \mathcal{A}; \mathcal{T}; \mathcal{R})$, where $\mathcal{S}$ – set of states s_t, $\mathcal{A}$ – set of actions a, $\mathcal{T}$ – transition probability from state s_t to s_{t+1} at time t after taking action a_t, $\mathcal{R}$ – reward function specifying reward r_t for s_t to s_{t+1} transition. We model the environment state s as the vector composed of utilisation u_{ij}^{d} of domain links (utilisation u_{ij}^{g} of abstracted links in GLBA case) and the normalised time $\sqcup_{norm} = (\sqcup_{cur} \mod \sqcup_{dur})/\sqcup_{dur}$, where $\sqcup_{cur}$ is the current time and $\sqcup_{dur}$ the duration of a day. Based on the state information, the DRL agents' output actions a_t – the routing graphs used by SDNCs/GSDNC. The agents' policies are evaluated using the reward functions shown in eq. 7 and eq. 8 (DLBAs and GLBA, respectively).

$$r^{d} = \Gamma^{d} - (\varphi^{d} + \varphi^{da}) \tag{7}$$

$$r^{g} = \Gamma^{g} - \varphi^{g} \tag{8}$$

Where φ^{da} denotes the abstracted domain load balancing factor (*i.e.*, φ calculated using the single domain graph abstraction as perceived by GLBA) and aggregate throughput Γ^{g} is calculated as shown in eq. 9.

$$\Gamma^{g} = \sum_{d=1}^{D} \Gamma^{d} \tag{9}$$

GLBA is focused on the load distribution across the abstracted and inter-domain links and total throughput as both contribute to the network operator's profits (capacity and energy savings due to even load per node). The DLBA reward considers the load balancing factors of domain φ^{d} and the abstracted domain φ^{da}. The latter makes DLBA pursue not only the "selfish" domain-level goals but also the global ones by considering the load imbalance on domains' abstracted links. As DLBAs try to reach domain goals, GLBA stabilises the traffic distribution across the whole network, neglecting the impact of rapid intra-domain routing changes. The interactions between the agents are shown in Figure 2.

The network state is acquired by dedicated entities. For DLBA, it is done by the Metrics Calculator (MC), which assembles state information s_t^{d} provided by SDNC (*i.a.*, $b_{ij}, c_{ij}, u_{ij}, \lfloor_m$) and calculates Γ^{d} and φ^{d}. The DLBA's actor consumes s_t^{d} to provide domain routing graph a_t^{d}. MC also derives the abstracted domain graph to obtain the abstracted domain load balancing factor φ^{da} for DLBA reward calculation. The abstracted domain graph is extended by the visible inter-domain links (with associated parameters) to obtain abstracted domain state s_t^{da} provided to Global Metrics Calculator (GMC). GMC joins the s_t^{da} received from each domain (eliminating link duplication) to get a global network state s_t^{g}, which is used to obtain rewards and global routing graph a_t^{g}, as in DLBA case. The routing graphs are sent to relevant SDNCs/GSDNC to be used for Dijkstra-based routing until the next update by DLBA/GLBA. Due to SDN specifics, to reduce SDNC load, the policy is updated every time interval T^{x} (T^{d} for DLBA, T^{g} for GLBA, $T^{g} >> T^{d}$), equal to the state sampling frequency. Each domain can, therefore, operate at a different time scale, as the network flows are routed using the current domains' routing graphs.

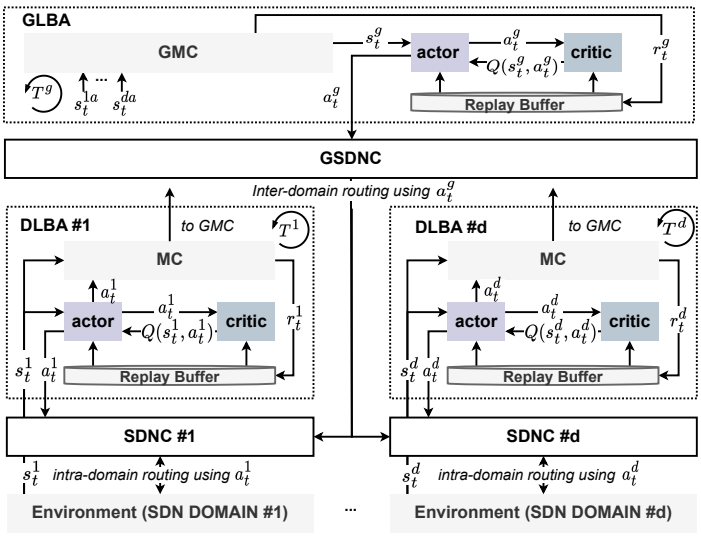

Fig. 2. Data exchange between the domain and global level entities

IV. Concept Evaluation and Results

The primary goal of the algorithm is to improve the load balancing in the network, allowing it to accommodate more traffic in the network and improve load-balancing policy convergence (compared to the centralised approach). To evaluate the HDRL-LB benefits, we conducted a set of tests verifying: i) performance gains under different network loads and traffic types; ii) the impact of the GLBA's overlay routing graph on the E2E performance. The tests were conducted under the Geant2019 [16] topology (40 nodes, 61 links), instantiated using the TopologyZoo dataset[17]. We divided topology into three domains (Fluid Communities algorithm), each managed and optimised by an SDNC/DLBA pair (cf. Figure 3) and added a total of 15 hosts. Each link capacity was set to 20 Mbps. The influx of flow requests was modelled with the Poisson process with $\lambda = 3$ (20 flows per minute). Six test scenarios were conducted, corresponding to 50%, 75%, and 100% network loads for Transport Control Protocol (TCP) and User Datagram Protocol (UDP) traffic (cf. Table I), which was generated using iPerf3 [18] library. As the basis for comparison, we used Weighted Dijkstra (WD), which selects the best path based on the current link utilisation.

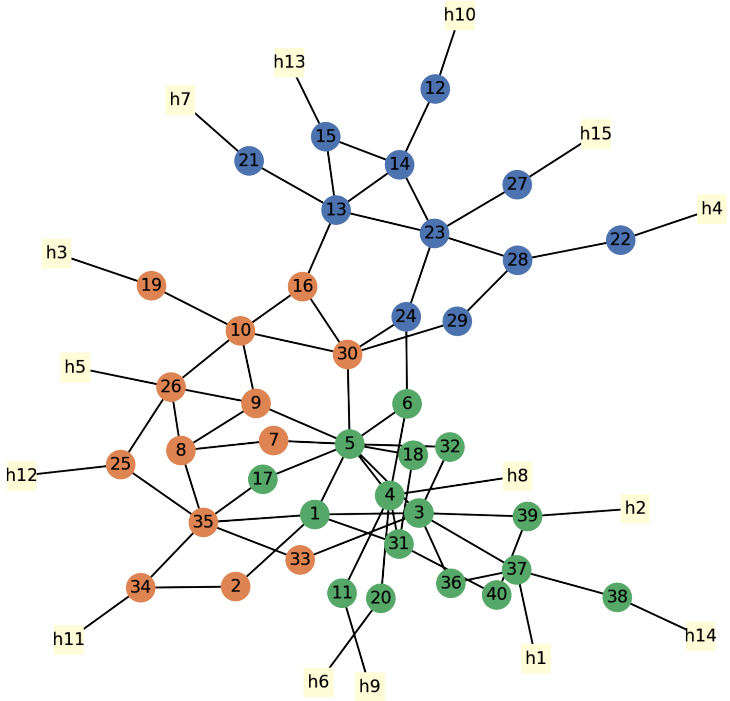

Fig. 3. GÉANT topology used for HDRL-LB evaluation [16] split into 3 domains, each handled by individual SDNC

TABLE I
TEST SCENARIOS

Scenario	Flows Throughput [Mbps]	Avg. Flows Throughput [Mbps]	Type	Network Load [%]
SC1	$\mathcal{U}(5.2, 8.6)$	6.9	TCP	100
SC2	$\mathcal{U}(3.9, 6.5)$	5.2	TCP	75
SC3	$\mathcal{U}(2.6, 4.3)$	3.5	TCP	50
SC4	$\mathcal{U}(5.2, 8.6)$	6.9	UDP	100
SC5	$\mathcal{U}(3.9, 6.5)$	5.2	UDP	75
SC6	$\mathcal{U}(2.6, 4.3)$	3.5	UDP	50

Experiments were conducted in the hierarchical multi-domain SDN emulation using Mininet [19] with OpenFlow-enabled Open vSwitches [20], Ryu [21] SDNC and Python-based DLBA, GLBA, and GSDNC. The agents used Keras with TensorFlow back-end [22].

A. Performance

We evaluated the performance gains using the following metrics: capacity improvements (total volume conveyed by the network), network availability (number of served/missed flows) and user-perceived throughput (obtained using time- and episode-correlated client/server iPerf3 logs). Figure 4 presents the improvements regarding the throughput Γ experienced by individual hosts.

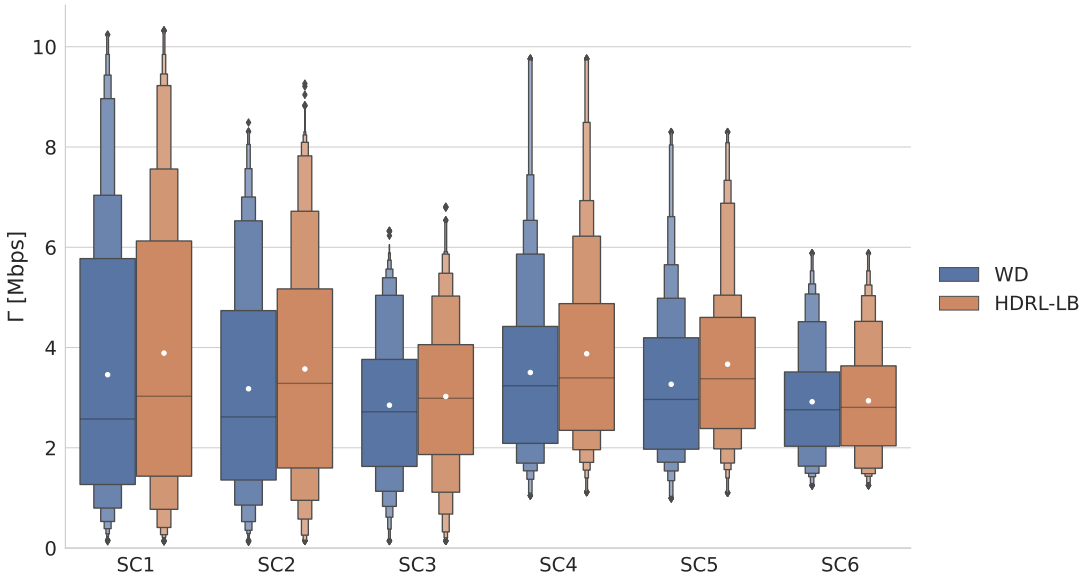

Fig. 4. Average Γ experienced by hosts (white dots – mean value)

In every case using HDRL-LB has led to improved average Γ. The first and third quartiles of Γ are higher for HDRL-LB than for the reference WD method. The general Γ improvement implies better throughput of individual network switches, linked with reduced buffering in the switches' ingress and egress queues. The latter indicates better load distribution across the network components improving the total carrying capacity. For the highest load scenarios, some values exceed the nominal maximum throughput specified in the test configurations (cf. Table I). This is due to the iPerf3 library configuration, which takes the average throughput as the input without the possibility of setting the upper bound of generated traffic for bursts in each interval. The results show similar trends for TCP and UDP traffic.

The biggest gains of HDRL-LB are achieved in the highest load scenarios. The cumulative gains compared to the WD algorithm are presented in Table II. Using HDRL-LB increases the aggregate data volume sent in the network (over 8.5% gain on average) and average throughput from 11.5% to almost 12.5% in the case of WD and TCP traffic (respectively, over 8% and 10.5% for UDP flows). Also, an almost 50% decrease of missed TCP flows can be observed. In UDP case, consecutive policy improvement led to a 30% decrease of flow drops after episode 50. In both cases, the improvement is achieved due to better load distribution and lowered congestion.

B. Convergence

In Figure 5 and Figure 6, the rewards obtained by the agents for all TCP and UDP traffic scenarios are presented.

TABLE II
CUMULATIVE PERFORMANCE COMPARISON OF HDRL-LB VS. WD ACROSS TRAINING EPISODES FOR HIGHEST LOAD SCENARIOS

Episode	Aggregate Data [GB]			Average Γ [Mbps]			Missed Flows [%]			SC
	HDRL-LB	WD	Gain [%]	HDRL-LB	WD	Gain [%]	HDRL-LB	WD	Gain [%]	
10	26.5	24.4	**8.61**	3.87	3.47	**11.6**	0.31	0.62	**50.0**	SC1 (TCP)
20	51.7	56.1	**8.51**	3.88	3.46	**11.9**	0.28	0.56	**50.0**	
30	78.7	85.7	**8.89**	3.89	3.46	**12.4**	0.25	0.55	**54.5**	
40	106.1	115.2	**8.58**	3.89	3.46	**12.4**	0.24	0.48	**50.0**	
50	133.3	144.8	**8.63**	3.89	3.46	**12.4**	0.24	0.45	**46.7**	
10	24.4	26.4	**8.2**	3.51	3.87	**10.3**	0.36	0.34	**-5.6**	SC4 (UDP)
20	51.5	55.7	**8.2**	3.51	3.88	**10.5**	0.33	0.32	**-3.0**	
30	78.7	85.1	**8.1**	3.51	3.87	**10.5**	0.30	0.33	**10.0**	
40	105.9	114.5	**8.1**	3.51	3.9	**10.6**	0.26	0.32	**23.1**	
50	133.1	143.8	**8.0**	3.50	3.9	**10.7**	0.24	0.32	**33.3**	

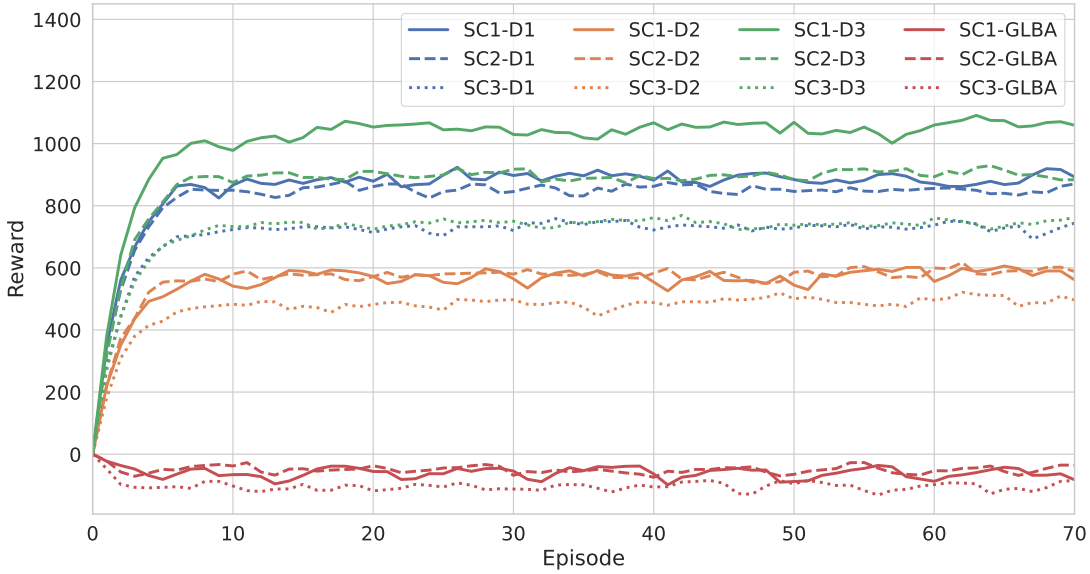

Fig. 5. Total rewards accumulated by the HDRL-LB agents across the training episodes under different TCP traffic loads

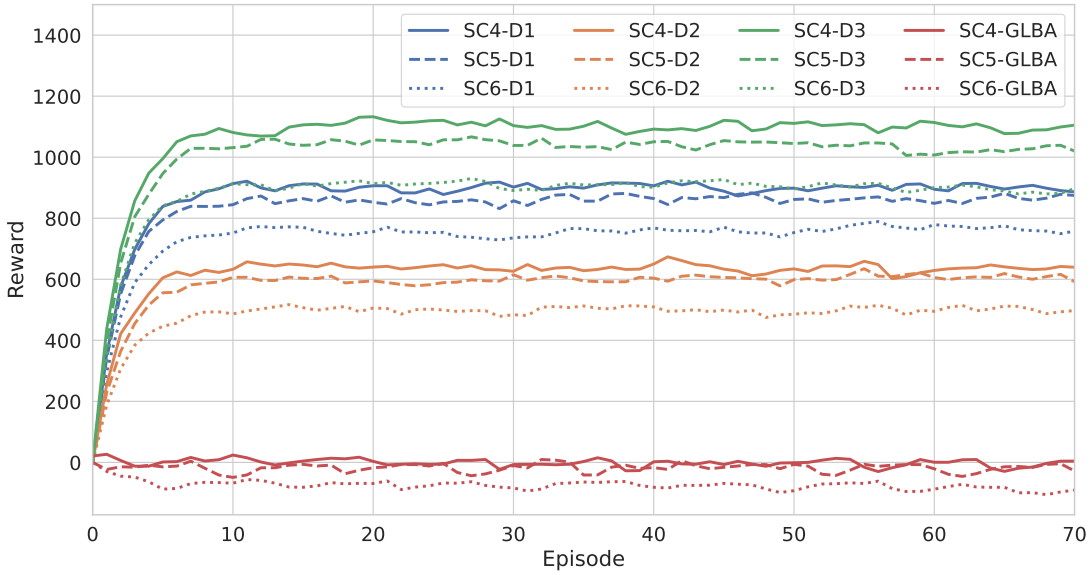

Fig. 6. Total rewards accumulated by the HDRL-LB agents across the training episodes under different UDP traffic loads

In every case, the convergence of the domain-level agents is relatively fast as the decent policy is reached after episode 20

with only minor improvements later on. The disproportions across the agents' rewards are linked with the domain topology, size, and the number of hosts. GLBA features the worst convergence due to the effects of conflicting domain-level policies. Nonetheless, a slight improvement during the training process can be seen. The poor stability of GLBA's learning is caused by the small number of domains corresponding to a limited number of states observed by the agent (and limited possibilities regarding load balancing across domains). Also, HDRL-LB uses the DDPG without convergence-oriented extensions, *e.g.*, prioritised replay [7].

C. Impact of overlay routing graph

We evaluated the impact of using the overlay routing graph by running the highest load scenarios (SC1, SC4) without the operating GLBA, *i.e.*, with the inter-domain routing performed using WD and intra-domain routing using modified DDPG with the same setup as in HDRL-LB case). The rewards accumulated by the agents in SC1 and SC4 are presented in Figure 7 and Figure 8, respectively.

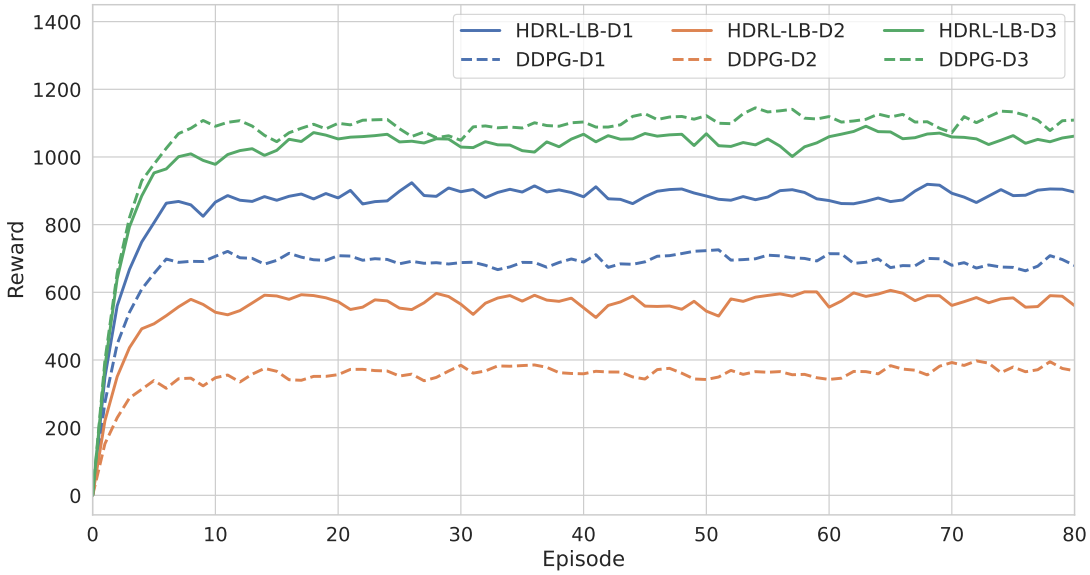

Fig. 7. Total rewards obtained for scenario SC1 using HDRL-LB algorithm vs. uncoordinated DDPG agents (without GLBA)

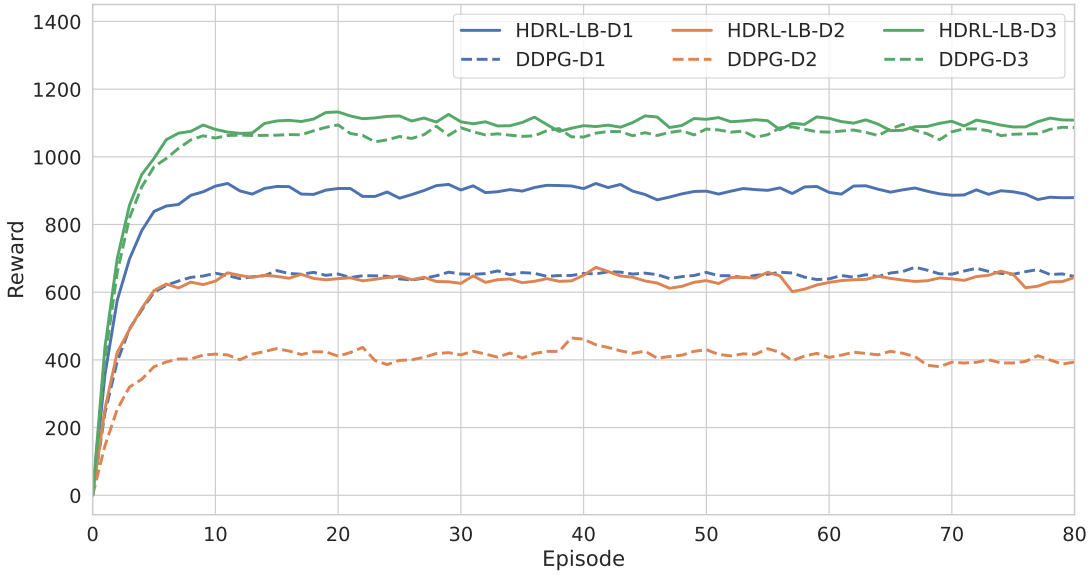

Fig. 8. Total rewards obtained for scenario SC4 using HDRL-LB algorithm vs. uncoordinated DDPG agents (without GLBA)

Similar policy convergence can be seen for supervised and non-supervised cases. While the global routing graph enforces some general rules on domain-level routing (*i.e.*, selects the BNs), there is no apparent impact on the convergence of DLBAs policies. However, the following benefits of using the overlay routing graph can be observed (cf. Figure 7 and Figure 8):

- more even reward distribution across domains, which reflects efficient domain load balancing;
- higher aggregate reward for domain agents (SC1: around 50% gain for D1, D2 and around 10% for D3; SC4: 30% for D1, 50% D2, up to 10% for D3);
- higher total reward obtained by the agents.

The above gains come solely from supervising GLBA and using network abstractions. The latter contributes to increased privacy and reusability as HDRL-LB can be applied in heterogeneous environments (comprising SDN and non-SDN domains) with limited access to the domains' internals and operations.

V. Summary and conclusions

This paper presents a novel multi-agent DDPG-based routing algorithm for dynamic load-balancing in hierarchical multi-domain SDN (HDRL-LB). The domain-level optimisation is done at a fast scale to support rapid traffic fluctuation, while the global one stabilises the E2E network operation. The concept uses network abstractions to improve the scalability of SDN and TE and support multi-provider environments. The CP and TE distribution improves metrics accuracy (due to SDNC and switches collocation), convergence speed (reduced solution search spaces) and applicability in large networks. HDRL-LB also considers normalised time to improve the agents' behaviour during peak network traffic. The tests show that HDRL-LB yields significant throughput (12% increase for TCP, 10% for UDP flows) and carrying capacity gains (over 8%) compared to WD while reducing the number of missed flows (over 50% for TCP, 30% for UDP). Future plans involve increasing HDRL-LB performance (GLBA policy convergence improvement), extensions towards fairness provisioning and tests in other topologies (under various network loads), while using various metric abstractions and reward functions.

References

[1] T. Magedanz and M.-I. Corici, "Getting ready for 6G research – understanding technological drivers towards 6G and emerging 6G management requirements," Fraunhofer Fokus, Tutorial at IEEE/IFIP NOMS, April 25th, 2022, 2022, https://owncloud.fokus.fraunhofer.de/index.php/s/4oIylfMlANQTZCB/download.

[2] A. Abuarqoub, "A review of the control plane scalability approaches in software defined networking," *Future Internet*, vol. 12, no. 3, 2020, doi: 10.3390/fi12030049.

[3] IETF NWG, "OSPF Version 2," IETF, Tech. Rep., 1998. [Online]. Available: https://www.ietf.org/rfc/rfc2328.txt

[4] M. Karakus and A. Durresi, "A survey: Control plane scalability issues and approaches in software-defined networking (SDN)," *Computer Networks*, vol. 112, pp. 279–293, 2017, doi: 10.1016/j.comnet.2016.11.017.

[5] Y. Hu, Z. Li, J. Lan, J. Wu, and L. Yao, "EARS: Intelligence-driven experiential network architecture for automatic routing in software-defined networking," *China Communications*, vol. 17, no. 2, pp. 149–162, 2020, doi: 10.23919/JCC.2020.02.013.

[6] D. M. Casas-Velasco, O. M. C. Rendon, and N. L. S. da Fonseca, "Intelligent routing based on reinforcement learning for software-defined networking," *IEEE Transactions on Network and Service Management*, vol. 18, no. 1, pp. 870–881, 2021, doi: 10.1109/TNSM.2020.3036911.

[7] J. Chen, Y. Wang, J. Ou, C. Fan, X. Lu, C. Liao, X. Huang, and H. Zhang, "ALBRL: Automatic load-balancing architecture based on reinforcement learning in software-defined networking," *Wireless Communications and Mobile Computing*, vol. 2022, pp. 1–17, May 2022, doi: 10.1155/2022/3866143.

[8] J.-J. Huang, Y.-Y. Chen, C. Chen, and Y. H. Chu, "Weighted routing in hierarchical multi-domain SDN controllers," in *2015 17th Asia-Pacific Network Operations and Management Symposium (APNOMS)*, 2015, pp. 356–359, doi: 10.1109/APNOMS.2015.7275362.

[9] T. Moufakir, M. F. Zhani, A. Gherbi, and O. Bouachir, "Collaborative multi-domain routing in SDN environments," *Journal of Network and Systems Management*, vol. 30, Jan 2022, doi: 10.1007/s10922-021-09638-0.

[10] Y. Liu, L. Zhao, J. Hua, W. Qu, S. Zhang, and S. Zhong, "Distributed traffic engineering for multi-domain SDN without trust," *IEEE Transactions on Cloud Computing*, vol. 10, no. 4, pp. 2481–2496, 2022, doi: 10.1109/TCC.2021.3067456.

[11] M. Ye, L. Huang, X. Deng, Y. Wang, Q. Jiang, H. Qiu, and P. Wen, "A new intelligent cross-domain routing method in SDN based on a proposed multiagent reinforcement learning algorithm," 2023, doi: 10.48550/arXiv.2303.07572.

[12] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," 2019, doi: 10.48550/arXiv.1509.02971.

[13] H. van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," 2015, doi: 10.48550/arXiv.1509.06461.

[14] H. van Hasselt, "Double Q-learning," in *Advances in Neural Information Processing Systems*, vol. 23. Curran Associates, Inc., 2010, pp. 2613–2621, doi: 10.5555/2997046.2997187.

[15] R. Kołakowski, S. Kukliński, and L. Tomaszewski, "Time-of-day-aware slice admission control," in *2023 IEEE International Mediterranean Conference on Communications and Networking (MeditCom)*, 2023, pp. 199–204, doi: 10.1109/MeditCom58224.2023.10266645.

[16] "GÉANT network," last accessed 10 May 2024. [Online]. Available: https://network.geant.org/gn4-3n/

[17] "The Internet Topology Zoo," last accessed 14 February 2024. [Online]. Available: http://www.topology-zoo.org/toolset.html

[18] ESnet, "iPerf - The ultimate speed test tool for TCP, UDP and SCTP," last accessed 10 May 2024. [Online]. Available: https://www.iperf.fr/

[19] "Mininet," last accessed 10 May 2024. [Online]. Available: https://mininet.org/

[20] Linux Foundation, "Open vSwitch," last accessed 10 May 2024. [Online]. Available: https://www.openvswitch.org/

[21] "Ryu SDN Framework," last accessed 10 May 2024. [Online]. Available: https://ryu-sdn.org/

[22] "Keras," last accessed 10 May 2024. [Online]. Available: https://keras.io/