DRL-TPE: Learning to Optimize TPE of Optical Interconnects to Accelerate Hitless Reconfiguration of OCS-based DCNs

Xiaoliang Chen[1], Wenbang Zheng[1,2], Shuoning Zhang[1], Xiaoyan Dong[1], Ke Meng[3], and Zuqing Zhu[1]

1. University of Science and Technology of China, Hefei, Anhui 230027, China, Email: zqzhu@ieee.org
2. Sun Yat-Sen University, Guangzhou, Guangdong 510080, China
3. Huawei Institute of Nanjing, Nanjing, Jiangsu 210012, China

Abstract: We leverage deep reinforcement learning (DRL) to optimize the topology engineering of optical circuit switching based data-center networks, such that the hitless reconfiguration to the obtained target physical topology can be finished with the fewest stages. Simulations confirm the benefits of our proposal over the state-of-the-art.

OCIS codes: (060.1155) All-optical networks; (060.4251) Networks, assignment and routing algorithms.

1. Introduction

The recent popularity of large language models (LLMs) has driven major data-center network (DCN) operators to consider replacing their electrical packet switching (EPS) based spine layers with optical interconnects based on optical circuit switching (OCS) [1]. As shown in Fig. 1(a), such an OCS-based DCN substitutes the traditional EPS-based spine layer with a set of OCS switches that compose a reconfigurable optical interconnect, to explore the benefits of OCS for large-throughput, adaptive and energy-efficient inter-pod communications [2]. Therefore, in addition to the traffic engineering (TE) in EPS-based DCNs, the OCS-based DCN enables topology engineering (TPE) to offer extra flexibility. Specifically, TPE reconfigures the optical interconnect and allows for steering of inter-pod connectivity to adapt to dynamic and skewed application traffic. This makes OCS-based DCNs particularly promising for the training of LLMs, since their collective communications alternate in predictable and skewed traffic patterns over time [1, 3].

In Fig. 1(a), the configurations of OCS switches determines the physical topology of the optical interconnect (each pair of pods are connected by which and how many connections through the OCS switches). We can abstract the physical topology to a logical one, which only records the number of connections between each pair of pods [2]. Then, a TPE operation can be detailed as: **Step 1**) getting a target logical topology based on the future inter-pod traffic matrix, **Step 2**) computing the target physical topology based on the target logical topology, and **Step 3**) planning implementable stages to reconfigure the optical interconnect from the current physical topology to the target one such that the common part of the current and future inter-pod traffic matrices can be migrated hitlessly. **Step 1** purely depends on the future traffic matrix (independent of the remaining two), and it can be easily solved with matrix decomposition [4]. However, **Steps 2** and **3** are pretty challenging to solve for: 1) their optimizations are both NP-hard, and 2) they are correlated (the target physical topology from **Step 2** determines the solution space of **Step 3**). In [2], the authors formulated an integer linear programming (ILP) model to tackle **Step 2** for a practical OCS-based DCN (Google's Gemini). They empirically set the optimization objective as minimizing the connection rewirings between current and target physical topologies, according to [5]. Nevertheless, as minimal rewiring ignores the traffic distribution on connections, it might not assist **Step 2** to obtain a target physical topology that is beneficial to **Step 3**. Moreover, solving ILP becomes intractable for large-scale problems, and without **Step 3**, service interruptions can happen during TPE. To the best of our knowledge, **Steps 2** and **3** have not been jointly optimized in the literature yet.

In this work, we explore the self-learning capability of deep reinforcement learning (DRL) to propose a TPE optimization method (namely, DRL-TPE), which jointly optimizes **Steps 2** and **3** effectively. DRL-TPE takes a target

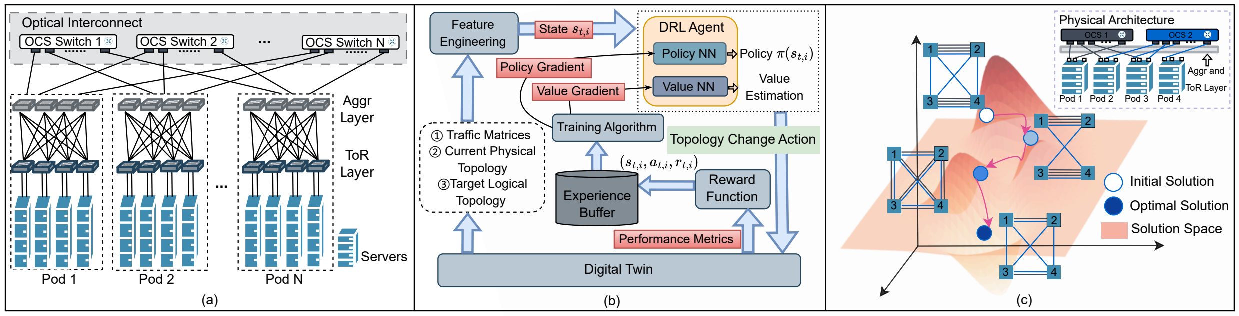

Fig. 1. (a) Architecture of OCS-based DCN, (b) operation principle of DRL-TPE, and (c) an illustrative example on progressive optimization done by DRL-TPE.

logical topology as input, and crafts a DRL agent based on graph convolutional networks (GCNs) to generate intermediate physical topologies, with which the current physical topology can be reconfigured toward a proper target physical topology progressively and hitless traffic migration is ensured for each topology change. As DRL-TPE solves each instance of the joint optimization with training, there is no need to generate artificial training data. This ensures its generalization ability across various network conditions and scales, which is also verified by our simulations.

2. Network Model and Design of DRL-TPE

In an OCS-based DCN (with the architecture in Fig. 1(a)), there are N pods that are directly connected to each other via N OCS switches. In particular, every pod/OCS switch has N ports, allowing them to wire a full mesh fabric. We denote the set of pods as V, let D_t be the common part of the current (at time $t-1$) and future (at time t) inter-pod traffic matrices, and represent the target logical topology as $G_t^*(V,E_t^*)$, where $e_t^*(u,v) \in E_t^*, u,v \in V$ tells the number of connections between pods u and v. Then, the joint optimization for TPE is to optimize the configuration $M_{t,n}$ of each OCS switch n and the reconfiguration procedure from the current physical topology ($\{M_{t-1,n}\}|_{n\in N}$) to the target physical topology ($\{M_{t,n}\}|_{n\in N}$), so that the actual target logic topology ($G_t(V,E_t)$) of the target physical topology approximates $G_t^*(V,E_t^*)$ well and hitless reconfiguration from $\{M_{t-1,n}\}|_{n\in N}$ to $\{M_{t,n}\}|_{n\in N}$ requires the fewest stages.

Fig. 1(b) illustrates the operation principle of DRL-TPE. <u>*Step 1*</u>: the feature engineering module fetches network state data (including D_t, G_{t-1} and $\{M_{t-1,n}\}$), and encodes the data as a state representation $s_{t,i}$ (i-th iteration), which is readily to be processed by the DRL agent. <u>*Step 2*</u>: the DRL agent extracts global representations from $s_{t,i}$ using the GCN-based policy neural network (NN) and hereby outputs a policy $\pi(a|s_{t,i})|_{a\in A}$ to direct modifications on the current physical topology. <u>*Step 3*</u>: the selected topology modification action $a_{t,i} \sim \pi(a|s_{t,i})$ is applied to update $\{M_{t-1,n}\}$ and E_{t-1} within the digital twin domain, which evaluates the performance of the obtained intermediate physical topology and returns an equivalent reward $r_{t,i}$. <u>*Step 4*</u>: the $(s_{t,i}, a_{t,i}, r_{t,i})$ tuple is stored in the experience buffer Γ and the agent repeats the above procedures. <u>*Step 5*</u>: every time k samples are collected, the agent performs training with the samples in Γ to update both the policy NN and value NN by reinforcing actions leading to higher long-term rewards. <u>*Steps 1-5*</u> are repeated until terminate conditions (*e.g.*, fixed number of iterations executed or gap to the target logical topology is below a threshold) are met. The above procedures are exemplified by Fig. 1(c). By learning such progressive physical topology modification policies, we optimize the target physical topology according to G_t^* and D_t. Note that, the computational complexity of DRL-TPE's online inferences is affordable, because only feed forwarding calculations (<u>*Step 5*</u> is skipped) of optimized numbers of iterations are involved. Finally, we apply a greedy method to reconfigure the current physical topology ($\{M_{t-1,n}\}$) to the obtained target one ($\{M_{t-1,n}\}$) with the fewest stages possible (*i.e.*, each stage tries to reconfigure the most connections according to the target physical topology until hitless traffic migration in the make-before-break manner cannot be achieved).

We apply the actor-critic framework to realize the above optimization for DRL-TPE. Specifically, we parameterize a policy and a value NN by $f_{\theta_p}(\cdot)$ and $f_{\theta_v}(\cdot)$, respectively, where θ_p and θ_v respectively represent the sets of NN weights. $f_{\theta_p}(s_{t,i})$ outputs a probability distribution $\pi(a|s_{t,i},\theta_p)$ over a discrete action space A, while $f_{\theta_v}(s_{t,i})$ estimates the attainable long-term reward (*i.e.*, the value). The detailed design of the DRL model for DRL-TPE is as follows.
<u>*State Input*</u>: since D_t, G_{t-1} and $\{M_{t-1,n}\}$ present strong topological characteristics and correlations, we encode them into a graph datum $s_{t,i}$ and employ a GCN to mine topological representations from $s_{t,i}$. Each node in $s_{t,i}$ corresponds to a pod and has a feature vector conveying its traffic demands and connectivity degrees in the current and target logical topologies. The edges in $s_{t,i}$ are created based on the current logical topology G_{t-1}.
<u>*Action Space*</u>: we design the actions for physical topology modification by leveraging local rewriting, *i.e.*, the actions can be adding, deleting and replacing certain connections in $\{M_{t,n}\}$.
<u>*NN Architecture*</u>: The policy NN $f_{\theta_p}(\cdot)$ and value NN $f_{\theta_v}(\cdot)$ employ the same GCN block, differing in only the readout modules, *i.e.*, the policy NN generates a probability array of length $|A|$ while the value NN outputs a single value estimation. The GCN block performs message passing and aggregation of multiple rounds to allow node and edge features to propagate throughout the graph, such that each node can perceive the global graph states [6]. Such topological processing also empowers the GCNs to potentially generalize over arbitrary graphs, effectively meeting the challenges of the scaling of OCS-based DCNs. The ultimate node features are combined and fed to the readout modules.
<u>*Reward Function*</u>: the DRL agent receives a reward $r_{t,i} = cs(G_{t-1,i}, G_t^*) - cs(G_{t-1,i-1}, G_t^*)$ in each iteration, where $cs(\cdot)$ computes the cosine similarity between two logical topologies.
<u>*Training*</u>: the loss function for training $f_{\theta_v}(\cdot)$ is defined as the mean squared error between the predicted and observed long-term rewards (*i.e.*, $\sum_{i\in[1,k]} \gamma^{i-1} \cdot r_{t,i}, \gamma \in (0,1]$ is the discount factor). In order to reinforce advantageous actions, the policy loss takes the form $-\frac{1}{k}\sum_{i\in[1,k]} \delta_{t,i} \cdot \log \pi(a_{t,i}|s_{t,i},\theta_p)$, where $\delta_{t,i}$ is the advantage of $a_{t,i}$, meaning how much the actual performance (total received reward) turns out to be better than the expected after taking $a_{t,i}$.

3. Performance Evaluation

We evaluate the performance of DRL-TPE with numerical simulations considering various network sizes, and compare it with a benchmark (Gemini), which is adapted based on the state-of-the-art in [2]. Specifically, Gemini first uses minimal rewiring as the optimization objective to obtain a target physical topology that approximates the target logical topology, and then leverages the same greedy method used by DRL-TPE to reconfigure the current physical topology to the target one. The number of pods in the OCS-based DCN is set as $|V| \in [8, 256]$, and each connection in the optical interconnect operates at 100 Gbps. As for the matrix of the inter-pod traffic (D_t) that needs to be migrated hitlessly during TPE, we make the total traffic in it consume 10%-50% of the optical interconnect's total capacity, and the number of non-zero elements in D_t is set within $[1, |V|^2 - |V|]$. The DRL agent of DRL-TPE is designed with a single-layer GCN for feature extraction and readout modules of one or three fully-connected hidden layers (128 neurons per layer) for generating local rewriting policies or value estimations. We set the episode size, learning rate, and maximum training episodes to be 50, 3×10^{-4}, and $5,000$, respectively.

We first show the training performance of DRL-TPE in Fig. 2(a). The evolution of cumulative reward received by the DRL agent indicates that the performance of the TPE solution for an OCS-based DCN with 8 pods improves rapidly until the training converges after $\sim$2,500 episodes, suggesting that the agent harnesses successful topology rewriting policies. Then, we apply the trained agent to DCNs in larger sizes to assess its generalization ability. Fig. 2(b) shows the similarity between actual and planned target logical topologies (the algorithms' performance on solving **Step 2**). Note that, as the optimization of **Step 2** is NP-hard, we might not always obtain a target physical topology whose logical topology (the actual logical topology) is exactly the same as the target one (the planned target topology). Although a gap might exist between the actual and planned target logical topologies, traffic provisioning performance would not degrade as long as the gap is minor and can be compensated by TE [7]. The results in Fig. 2(b) indicate that when there are no more than 32 pods, Gemini obtains actual logical topologies slightly closer to the planned ones because it is devoted to **Step 2** (topology optimization). However, when the OCS-based DCN further expands, Gemini fails to derive solutions due to the intractability of solving the ILP. In Fig. 2(b) and 2(c), we insert shaded bars to indicate the vacancy of feasible solutions. Whereas Fig. 2(b) verifies that our DRL-TPE performs steadily and consistently generates feasible solutions, confirming the generalization ability of the learned policies across different network sizes. Fig. 2(c) shows the average number of reconfiguration stages required by **Step 3**. On average, DRL reduces this number by 23.7% over Gemini, elucidating the advantage of joint optimization over independent designs.

4. Summary

We leveraged DRL to propose a TPE optimization method for OCS-based DCNs (namely, DRL-TPE), which jointly optimizes the target physical topology and the reconfiguration procedure to it based on a target logical topology and current and future inter-pod traffic matrices. Extensive simulations confirmed the generalization ability of DRL-TPE across various network conditions and scales, and demonstrated that it can reduce the required reconfiguration stages by 23.7% on average over the existing benchmark.

References

[1] H. Liu *et al.*, "Lightwave Fabrics: At-scale optical circuit switching for datacenter and machine learning systems," *ACM SIGCOMM 2023*.

[2] M. Zhang *et al.*, "Gemini: Practical reconfigurable datacenter networks with topology and traffic engineering," *arXiv:2110.08374*, 2021.

[3] H. Yang *et al.*, "Traffic-aware configuration of all-optical data center networks based on Hyper-FleX-LION," *IEEE/ACM Trans. Netw.*, pp. 2675-2688, Jun. 2024.

[4] G. Porter *et al.*, "Integrating microsecond circuit switching into the data center," *ACM SIGCOMM 2013*.

[5] S. Zhao *et al.*, "Minimal rewiring: Efficient live expansion for Clos data center networks," *NSDI 2019*.

[6] J. Bruna *et al.*, "Spectral networks and locally connected networks on graphs," arXiv:1312.6203, 2013.

[7] M. Teh *et al.*, "Enabling quasi-static reconfigurable networks with robust topology engineering," *IEEE/ACM Trans. Netw.*, vol. 31, pp. 1056-1070, Jun. 2023.

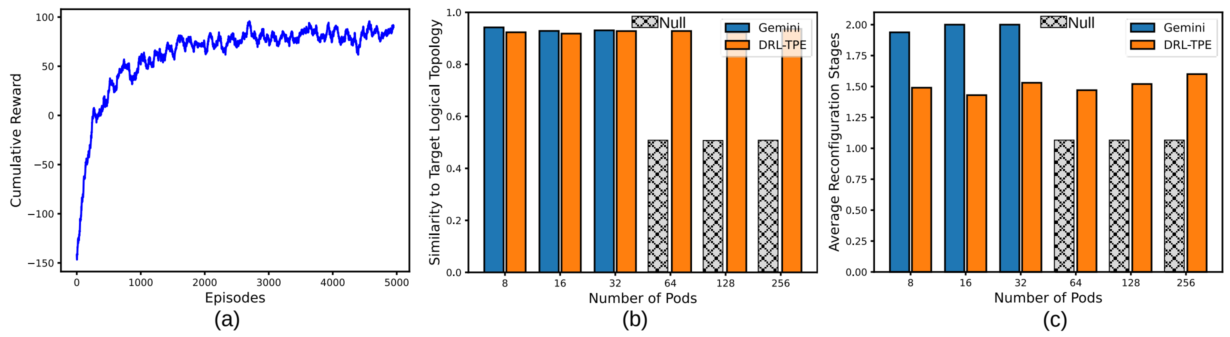

Fig. 2. (a) Training performance of DRL-TPE ($|V| = 8$), and (b)-(c) performance comparisons between DRL-TPE and Gemini ($|V| \in [8, 256]$): (b) similarity to planned target logical topologies, and (c) average reconfiguration stages.