XLRON: Accelerated Reinforcement Learning Environments for Optical Networks

Michael Doherty[1,*] **Alejandra Beghelli**[1]

[1] *Optical Networks Group, University College London (UCL), Roberts Building. Torrington Place, London WC1E 7JE, United Kingdom*

[*] *michael.doherty.21@ucl.ac.uk*

Abstract: We present XLRON: an open source project enabling, for the first time, GPU-accelerated reinforcement learning on optical network problems. We demonstrate 100-1000x speed-up in training time over similar tools, thereby opening new research possibilities.

1. Introduction and Motivation

Reinforcement learning (RL) has been the subject of interest as a solution method to resource allocation problems in optical networks since the publication of DeepRMSA in 2019 [1]. Many works have shown RL methods to achieve lower service blocking probability than greedy heuristics for problems such as routing modulation and spectrum assignment (RMSA) and virtual optical network embedding (VONE) [2]. However, many challenges remain before RL-trained policies can be deployed in production networks, such as scalability, robustness and resilience, interpretability, and further performance improvements.

Several factors have impeded progress on these challenges in the optical networks research community. 1) Training RL agents is data- and compute-intensive, requiring the evaluation of many combinations of hyperparameters and millions to billions of environment transitions. Fast generation of training samples requires clusters of CPU servers to host simulation environments, with a GPU server for action inference and parameter updates (Figure 1 left). Such infrastructure is not trivial to setup or maintain and debugging programs can be difficult. 2) Despite efforts from the open source optical-rl-gym [3], there is a lack of standard learning environments and benchmarks in the field. Reproduction and comparison of research findings is not straightforward and progress is slowed by the need to reimplement work. 3) The reliability of research is limited by closed source code and the lack of any comprehensive unit tests to verify the behaviour of simulation environments.

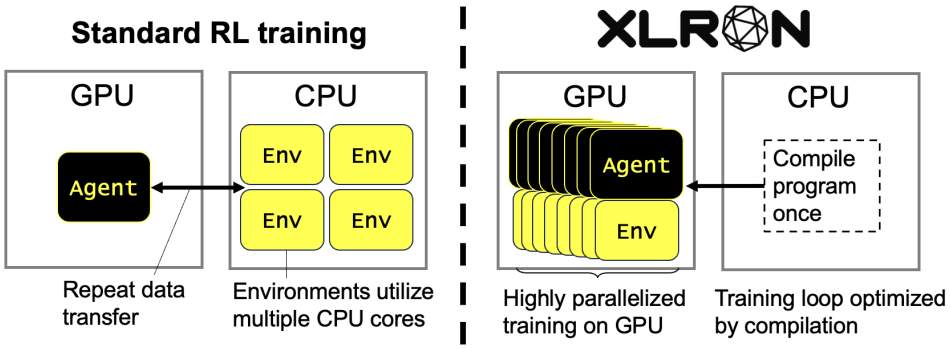

Fig. 1: Illustration of XLRON training paradigm. The entire training program is compiled and optimised before running in parallel on GPU. Data resides on the GPU during training to avoid costly data transfers.

XLRON [4] ("ex-el-er-on") ("**X**"-elerated **L**earning and **R**esource allocation for **O**ptical **N**etworks) aims to overcome these obstacles by significantly decreasing the time required to train RL agents using limited infrastructure, with open source code for reproducibility and reliable unit tests. The improved training speed is achieved by implementing GPU-compatible simulations and training code, to allow massive parallelization (Figure 1 right). This allows researchers to run large hyperparameter sweeps, train on multiple random seeds in parallel, and quickly iterate on their ideas. XLRON is developed in adherence to core software engineering principles of test-driven development, modularity, extensibility, and usability, to ensure correctness of simulations and allow easy customisation for research into new problem variants. It currently provides a set of configurable learning environments

for RMSA, routing and spectrum assignment (RSA), routing and wavelength assignment (RWA), and VONE. The aim of the project is to accelerate research into RL for optical networks and become the de facto standard library of learning environments in the field.

2. Methodology

To achieve the training time speed-ups detailed in section 3, XLRON draws inspiration from novel RL training architectures proposed by Google DeepMind [5]. By using the JAX numerical computing framework [6] for Python to implement both the RL training algorithm [7] and the simulation environment, the entire training loop can be compiled as a single program and loaded to GPU, where it is parallelized. By using JAX, XLRON enables easy deployment to the free Google TPU research cloud, to benefit researchers without direct GPU access.

Using JAX to implement a GPU-compatible optical network simulation imposes constraints on the program structure and the data on which it operates. For data constraints, all values must be either scalar or array, and the dimensions of arrays must be static i.e. known at compile time. The network model used by XLRON therefore comprises a series of arrays to track occupancy of frequency slot units (FSU) on links, node resources, service duration times, network topology connectivity, and pre-computed shortest paths. As a program constraint, JAX enforces a functional programming paradigm. Adherence to these constraints allows just-in-time (JIT) compilation of the program and deployment to accelerator hardware e.g. GPU or TPU.

The functional programming style facilitates thorough testing of simulation code. To ensure correctness, XLRON features over 800 unit test cases, including test variants for running on different hardware, with test coverage tracked by the online code repository. Reproducibility is improved by the sophisticated pseudo-random number generation in JAX and the simple command-line interface offered by XLRON to re-run experiments. Motivated readers are invited to explore the project's GitHub repository [4].

3. Case Studies

To demonstrate the speed-up enabled by XLRON, we compare with training using the popular Stable-Baselines3 (SB3) [8] library of RL algorithms and the optical-rl-gym [3] for the environment. The training setup is matched to that in DeepRMSA-FLX [1], using the NSFNET topology with 100 FSU per link, 150 Erlang traffic load, identical training hyperparameters and observation and action spaces. Both SB3 and XLRON use the PPO RL algorithm in this comparison and are trained for 10^4 episodes of 10^3 timesteps each. Each episode starts with an empty network. XLRON training used an Nvidia A100 80GB GPU and SB3 used a 10-core Apple M1 Pro CPU.

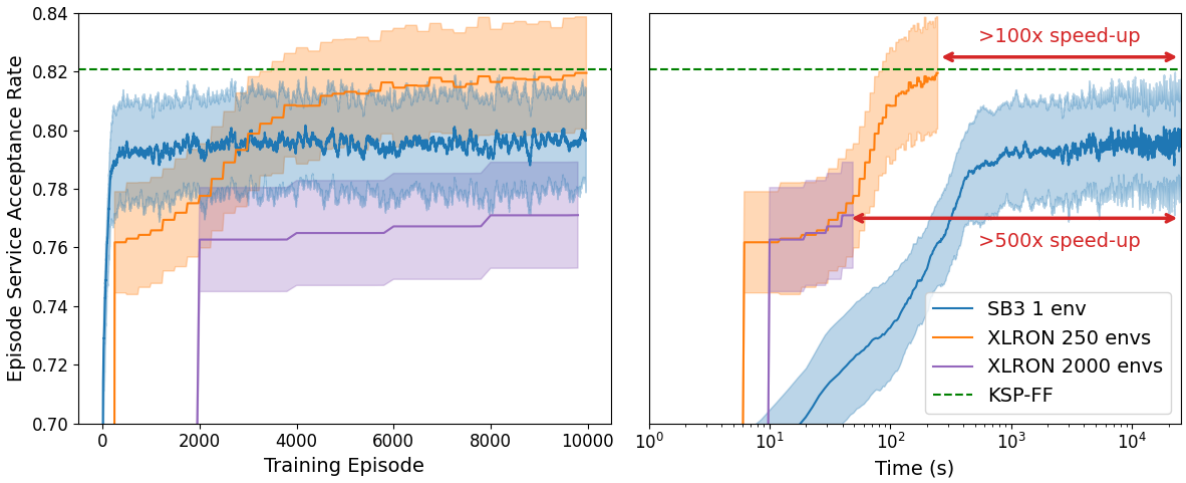

Fig. 2: Comparison of training curves for RL agents using StableBaselines3 and optical-rl-gym ("SB3") or XLRON. Left figure shows training progress over 10^4 episodes (10^7 timesteps total). Right figure shows same curves plotted with wall-clock training time on a log scale. Shaded areas indicate standard deviation of acceptance rate across environments for XLRON and across 3 separate training runs for SB3.

Figure 2 shows the service acceptance rate per episode during training. XLRON training is shown for 250 (orange) and 2000 (purple) parallel environments. The total number of timesteps is divided across environments. The final service acceptance rate achieved by each differs due to the different number of parallel environments and fixed hyperparameters, but XLRON is able to match the performance of the K-shortest path first-fit (KSP-FF) heuristic (green line) within the 10^7 timesteps training budget. The speed-up is over 100x for XLRON compared to

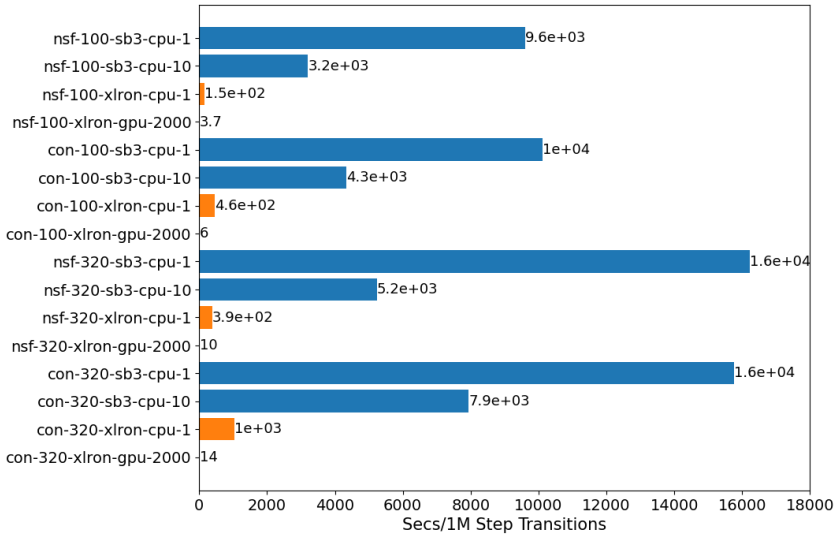

Fig. 3: Comparison of time taken to train on one million environment steps for the VONE problem. Experiment names on y-axis follow the naming convention: topology (NSFNET or CONUS) - number of FSU per link - XLRON or SB3 - device type - number of vectorised environments.

SB3, or 500x for the same training steps with 2000 parallel environments. It should be noted that the DeepRMSA action space, observation space, and default parameters are selected so as to create a fair and standard comparison. Optimising performance on the task is not the goal of this paper, however XLRON is easily customisable to allow novel action and observation spaces and neural network architectures to improve performance.

For a second case study, Figure 3 shows the time taken to achieve one million environment steps in four different instances of the VONE problem. The environment, similar to optical-rl-gym, is from [2]. VONE requires, at each timestep, the allocation of node resources and spectral paths between nodes in order to host a virtual network. Each instance used either the NSFNET or CONUS topology and 100 or 320 FSUs per link. NSFNET topology has 14 nodes and 22 links, CONUS has 75 nodes and 99 links. Compared to SB3 (blue), XLRON (orange) is consistently $\sim$10x faster when compiled for and run on CPU and can be over 1000x faster with 2000 environments run on GPU. For VONE on the CONUS topology with 320 FSUs per link, XLRON generates and trains on one million samples in 14 seconds compared to 2 hours 10 minutes for 10 SB3 environments on CPU, a speed-up of $\sim$560x.

4. Conclusions

Comparisons of XLRON with existing RL frameworks for optical networks indicate a speed-up of approximately 10x for a single environment on CPU and 100-1000x when parallelised on GPU. This allows research in these areas to enter a new, high-data regime. It is anticipated that the highly scalable learning environments presented by XLRON will enable breakthroughs in the application of RL to resource allocation problems in optical networks.

Acknowledgements. Financial support from EPSRC Centre for Doctoral Training in Connected Electronic and Photonic Systems (CEPS CDT) and EPSRC Programme Grant TRANSNET (EP/R035342/1) is gratefully acknowledged.

References

1. Chen, X. *et al.* en. *J. Light. Technol.* **37,** 4155–4163 (Aug. 2019).
2. Doherty, M. *et al. 2023 Eur. Conf. on Opt. Commun. (ECOC)* (Oct. 2023).
3. Natalino, C. *et al. 2020 22nd Int. Conf. on Transparent Opt. Networks (ICTON),* 1–5 (July 2020).
4. Doherty, M. *XLRON* https://github.com/micdoh/XLRON.git.
5. Hessel, M. *et al.* arXiv:2104.06272 [cs]. Apr. 2021.
6. Bradbury, J. *et al.* http://github.com/google/jax.
7. Lu, C. *et al. Adv. Neural Inf. Process. Syst.* **35,** 16455–16468 (2022).
8. Raffin, A. *et al. J. Mach. Learn. Res.* **22,** 1–8 (2021).