Optimizing and Extending the Functionality of EXARL for Scalable Reinforcement Learning

SAI CHENNA*, KATHERINE COSBURN*, UCHENNA EZEOBI*, MAXIM MORARU*, HYUN LIM*, JULIEN LOISEAU*, JAMAL MOHD-YUSOF*, ROBERT PAVEL*, VINAY RAMAKRISHNAIAH*, ANDREW REISNER*, and KAREN TSAI*, Los Alamos National Laboratory, USA

Easily eXtendable Architecture for Reinforcement Learning (EXARL) is a scalable reinforcement learning framework, part of the Exascale Computing Project (ECP) funded ExaLearn program, designed to facilitate reinforcement learning (RL) research for complex scientific environments. In RL, agents are algorithms that interact and learn from an environment, such as a game or a scientific simulation, with an aim to maximize reward. We improve the existing EXARL framework by optimizing and extending the current functionality by incorporating additional agents, such as Advantage Actor Critic (A2C/A3C) and Twin Delayed Deep Deterministic Policy Gradient (TD3) to the framework, as well as improve upon these and existing agents using v-trace and prioritized experience replay. We also improve the scalability of the framework by optimizing communication across multiple learners and between actors and learners, as well as accelerate the data generation pipeline of the Deep Q-Network (DQN) agent by leveraging environment processes.

CCS Concepts: • **Computing methodologies** → **Multi-agent reinforcement learning**; **Distributed algorithms**.

Additional Key Words and Phrases: reinforcement learning, high-performance computing, distributed computing

ACM Reference Format:
Sai Chenna, Katherine Cosburn, Uchenna Ezeobi, Maxim Moraru, Hyun Lim, Julien Loiseau, Jamal Mohd-Yusof, Robert Pavel, Vinay Ramakrishnaiah, Andrew Reisner, and Karen Tsai. 2021. Optimizing and Extending the Functionality of EXARL for Scalable Reinforcement Learning. In . ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Reinforcement learning (RL) is a subset of machine learning wherein an agent interacts and learns from its environment, with the overall goal of achieving a maximum reward from its environment over time. Apart from games [11–13] and computer vision applications [2], RL is being used increasingly in scientific disciplines to solve many control problems, specifically those that can be formulated as a Markov Decision Process (MDP). The environments used in scientific RL studies are often complex and take substantial computation time to run, even when run in parallel. The ability to scale these computations to multiple nodes can considerably reduce computation time, allowing for more robust testing and prototyping of ideas. EXARL is a scalable RL framework for scientific applications which addresses these problems directly by providing a scalable framework of customizable environments, agents, and workflows which help improve performance and reduce execution time. In the EXARL framework, the agent is divided into actors and learners. Here the actors are responsible for interacting with the environment and generating trajectories (training data), which are fed

*All authors contributed equally to this research.

Manuscript submitted to ACM

to the learner to update the policy. Finally, the updated policy is shared with the actors, which perform suitable actions to maximize reward. Similar to the IMPALA [3] architecture, the actors and learners are decoupled in order to scale the reinforcement learning. We now summarize the improvements to performance we have implemented in the EXARL framework to help with this scaling, as well as the agents and algorithmic improvements we have added and tested.

2 PERFORMANCE IMPROVEMENTS

One key performance bottleneck while using Deep Q Network (DQN) [8] agent is in the data generation pipeline. DQN uses the Bellman Optimality Equation to calculate the target states on a set of trajectories in order to update the policy. Calculating the target state on the entire training data is a compute-intensive operation and currently takes 90% of the total computation time. To accelerate the time taken for data generation, we made use of the fact that both actor and environment interact sequentially, hence the environment processes remain idle while the actor process is generating the training data, thus we could offload the Bellman optimality calculation onto the environment processes.

One of the most important goals of the EXARL framework is to achieve good scalability on different HPC platforms. Current approaches focus mainly on the convergence and stability aspects rather than the execution time performance. The challenging part of EXARL is to achieve both of these objectives. In order to improve the scalability of EXARL we propose a multi-learner workflow based on MPI [5] one-sided communication paradigm by building upon an existing multi-learner workflow using multiple RMA windows and an existing queue data structure functionality. Here each actor has a local queue data structure remotely accessible by all learners. Learners can retrieve training data from these queues and share the updated weights via a global RMA window. Actors and learners are synchronized using passive target RMA locks and the training part is performed in parallel by using Horovod [10]. The proposed communication pattern allows us to benefit from high-performance interconnect technologies, permitting us to scale on multiple nodes.

3 ADDITIONAL AGENTS

One area for improving upon EXARL is that of developing additional agents, such as (Asynchronous) Advantage Actor Critic (A2C/A3C) [7] and Twin Delayed Deep Deterministic Policy Gradient (TD3) [4]. The decoupled nature of the actors and learners in the EXARL framework make it hard for on-policy agents such as A2C/A3C to stay on policy for long, so we utilize the v-trace algorithm [3] to help stabilize training. We observe that, with v-trace, A2C/A3C outperforms DQN on the OpenAI gym's CartPole environment and the custom scientific environment ExaBooster, which was developed for a booster at the FNAL particle accelerator at FermiLab as way to use RL for controlling beam quality. We also add Prioritized Experience Replay [9] to the off-policy agents Deep Deterministic Policy Gradient (DDPG) [6] and find that it speeds up convergence on OpenAI gym's Pendulum environment.

4 RESULTS, CONCLUSIONS, AND FUTURE WORK

We introduced EXARL, a scalable reinforcement learning framework for complex scientific environments. We improved upon the existing framework by accelerating the data generation pipeline for faster convergence and observed a 3.3x speedup while scaling the computation onto 4 processes. Preliminary results using efficient RMA communication patterns for the asycnhronous workflow demonstrate improved scalability performance, where we saw a 77% decrease in total execution time when using 20 learners and 120 actors on 4 nodes. We evaluated our prototype on an Intel Broadwell CPU cluster, containing 36 cores per node and an InfiniBand [1] ConnectX-4 interconnect. Finally, we have expanded the capability of EXARL by including additional agents like A2C/A3C and TD3 and tested their convergence using an NVIDIA A100 GPU. Going forward, we would like to scale our framework on large-scale systems.

ACKNOWLEDGMENTS

All authors worked on this project in the Computer, Computational, and Statistical Sciences Division at Los Alamos National Laboratory during the 2021 ASC/CNLS/NSEC LANL Co-Design Summer School. Additionally, this research was supported by the Exascale Computing Project (17-SC-20-SC), a collaborative effort of the U.S. Department of Energy Office of Science and the National Nuclear Security Administration. This work was approved for unlimited release under LA-UR-21-27795.

REFERENCES

[1] Rajkumar Buyya, Toni Cortes, and Hai Jin. 2002. *An Introduction to the InfiniBand Architecture.* 616–632. https://doi.org/10.1109/9780470544839.ch42

[2] Qingxing Cao, Liang Lin, Yukai Shi, Xiaodan Liang, and Guanbin Li. 2017. Attention-aware face hallucination via deep reinforcement learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.* 690–698.

[3] Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Volodymir Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, Shane Legg, and Koray Kavukcuoglu. 2018. IMPALA: Scalable Distributed Deep-RL with Importance Weighted Actor-Learner Architectures. arXiv:1802.01561 [cs.LG]

[4] Scott Fujimoto, Herke van Hoof, and David Meger. 2018. Addressing Function Approximation Error in Actor-Critic Methods. arXiv:1802.09477 [cs.AI]

[5] Torsten Hoefler, James Dinan, Rajeev Thakur, Brian Barrett, Pavan Balaji, William Gropp, and Keith Underwood. 2015. Remote Memory Access Programming in MPI-3. *ACM Trans. Parallel Comput.* 2, 2, Article 9 (June 2015), 26 pages. https://doi.org/10.1145/2780584

[6] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. 2019. Continuous control with deep reinforcement learning. arXiv:1509.02971 [cs.LG]

[7] Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous Methods for Deep Reinforcement Learning. arXiv:1602.01783 [cs.LG]

[8] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. 2013. Playing Atari with Deep Reinforcement Learning. arXiv:1312.5602 [cs.LG]

[9] Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. 2016. Prioritized Experience Replay. arXiv:1511.05952 [cs.LG]

[10] Alexander Sergeev and Mike Del Balso. 2018. Horovod: fast and easy distributed deep learning in TensorFlow. *arXiv preprint arXiv:1802.05799* (2018).

[11] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. 2016. Mastering the game of Go with deep neural networks and tree search. *nature* 529, 7587 (2016), 484–489.

[12] Gabriel Synnaeve, Nantas Nardelli, Alex Auvolat, Soumith Chintala, Timothée Lacroix, Zeming Lin, Florian Richoux, and Nicolas Usunier. 2016. Torchcraft: a library for machine learning research on real-time strategy games. *arXiv preprint arXiv:1611.00625* (2016).

[13] Oriol Vinyals, Timo Ewalds, Sergey Bartunov, Petko Georgiev, Alexander Sasha Vezhnevets, Michelle Yeo, Alireza Makhzani, Heinrich Küttler, John Agapiou, Julian Schrittwieser, et al. 2017. Starcraft ii: A new challenge for reinforcement learning. *arXiv preprint arXiv:1708.04782* (2017).