Data-centric Machine Learning Workshop, ICLR (2024) Submitted 2/24; Revised ; Published

Enabling High Data Throughput Reinforcement Learning on GPUs: A Domain Agnostic Framework for Data-Driven Scientific Research

Tian Lan*, **Huan Wang, Caiming Xiong and Silvio Savarese**
Salesforce Research, USA
TIAN.LAN@SALESFORCE.COM

Abstract

We introduce *WarpSci*, a domain agnostic framework designed to overcome crucial system bottlenecks encountered in the application of reinforcement learning to intricate environments with vast datasets featuring high-dimensional observation or action spaces. Notably, our framework eliminates the need for data transfer between the CPU and GPU, enabling the concurrent execution of thousands of simulations on a single or multiple GPUs. This high data throughput architecture proves particularly advantageous for data-driven scientific research, where intricate environment models are commonly essential.

Keywords: throughput, reinforcement learning, gpu acceleration, data-driven science

1 Introduction

Reinforcement Learning (RL) stands out as a powerful algorithm for training AI agents, applicable in diverse domains such as strategy games(OpenAI, 2018; Vinyals et al., 2019), robotics (Gu et al., 2017; Ibarz et al., 2021), and large language models (Ouyang et al., 2022). Notably, there has been a recent surge in interest regarding the application of RL techniques in scientific research, encompassing diverse fields such as multi-agent[1] modeling in economics, climatology, and epidemiology (Zheng et al., 2022; Trott et al., 2021; Zhang et al., 2022); signal processing in astrophysics (Nousiainen, J. et al., 2022; Yatawatta, 2023); and investigating reaction paths in chemistry (Lan and An, 2021; Yoon et al., 2021). However, numerous engineering and scientific challenges persist in the adoption of RL in scientific investigations. The performance of RL implementations can decelerate significantly when simulations become data-intensive, particularly in scenarios involving numerous agents or high-dimensional state or action spaces, resulting in experiments that span weeks. The comparatively low data throughput of RL further contributes to the emergence of non-stationary and strongly correlated data sequences, while the finite-horizon roll-out in RL introduces bias over the value function estimation (Mnih et al., 2016; Zhang et al., 2020; Lan and An, 2021). Regrettably, such complexity and challenges are commonplace in data-driven scientific modeling. For instance, in economic simulations, the construction of a realistic environment necessitates hundreds of agents and numerous actions (Zhang et al., 2022). Similarly, the study of catalytic reaction pathways involves navigating a chemical potential energy landscape that can easily exceed twenty dimensions with extreme noise (Lan and An, 2021). While distributed systems are employed to scale RL performance,

1. An *agent* is an actor in an environment. An *environment* is an instance of a simulation and may include many agents with complex interactions. An agent is neither an environment nor a policy model.

the associated costs of worker communication and data transfer can be very high (Espeholt et al., 2018, 2020; Hoffman et al., 2020; Pretorius et al., 2021), as detailed in Appendix A.

2 Contribution

The primary objective of this *Extended Abstract* is to bring attention to the challenge of RL in scientific research arising from the data throughput, and introduce our comprehensive solution, *WarpSci*. *WarpSci* is a computational framework specifically designed to achieve massively high-throughput and domain-agnostic RL simulation in the context of data-driven scientific research. The framework builds upon the foundation of WarpDrive (Lan et al., 2022) which is accessible at `https://github.com/salesforce/warp-drive`.

WarpSci performs the entire RL workflow on a single or multiple GPUs, utilizing a unified and in-place data store within GPUs for simulation roll-outs and training. This minimizes the data transfer between CPU and GPU or within GPU, reducing simulation and training time significantly. The framework also leverages GPU parallelization to concurrently run thousands of RL simulations, operating independently in the dedicated GPU blocks and concurrently producing exceptionally large batches of experience. *WarpSci* offers simple Python classes located on the CPU to streamline all relevant CPU-GPU communication and interactions essential for RL, and offer simple toolings for constructing custom RL environments connected to the CUDA back-end.

This high throughput yet cost-effective architecture proves particularly advantageous for data-driven scientific research, where enormous data consumption, complex agent interactions, and diverse environments are usually indispensable. More details of the design choice and the computational architecture are provided in Appendix B.

3 Examples

We present three examples: *gym* classic control (Brockman et al., 2016) for benchmarking, a multi-agent economic simulation (Trott et al., 2021), and generalizable catalytic reaction paths modeling (Lan and An, 2021; Lan et al., 2024). All experiments ran on a single Nvidia A100 GPU on the Google Cloud Platform. Due to space constraints, we provide a brief summary in this section, with more information in Appendix C.

Throughput: *WarpSci* achieves significantly higher (at least $10 - 100\times$) throughput than the distributed systems at low cost (a single A100 GPU). For example, 8.6M environment steps/second for 10K concurrent cartpole environments, 0.12M for 1K concurrent economic simulations and 0.95M for catalytic reaction modeling with 2K concurrent environments [2]. Scaling almost linearly to thousands of environments or agents, *WarpSci* demonstrates near-perfect parallelism. It can also train across multiple GPUs for further throughput scaling. **Convergence**: Our study indicates that training with an increased data throughput generated by concurrent environments achieves faster and more stable global convergence. **Environments Agnostic**: *WarpSci* offers tools to develop custom environments for diverse scientific research topics, and supports actor-critic algorithms for both discrete and continuous actions.

2. In certain experiments, we employed a reduced level of concurrency to optimize the trainer's capacity and memory space.

References

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym, 2016. URL `https://github.com/openai/gym`.

Benjamin W. J. Chen, Lang Xu, and Manos Mavrikakis. Computational methods in heterogeneous catalysis. *Chemical Reviews*, 121(2):1007–1048, 2021. doi: 10.1021/acs.chemrev.0c01060. URL `https://doi.org/10.1021/acs.chemrev.0c01060`. PMID: 33350813.

Jingguang G Chen, Richard M Crooks, Lance C Seefeldt, Kara L Bren, R Morris Bullock, Marcetta Y Darensbourg, Patrick L Holland, Brian Hoffman, Michael J Janik, Anne K Jones, Mercouri G Kanatzidis, Paul King, Kyle M Lancaster, Sergei V Lymar, Peter Pfromm, William F Schneider, and Richard R Schrock. Beyond fossil fuel-driven nitrogen transformations. *Science*, 360(6391):eaar6611, 2018. ISSN 1095-9203 (Electronic), 0036-8075 (Print). doi: 10.1126/science.aar6611.

Steven Dalton, Iuri Frosio, and Michael Garland. Accelerating reinforcement learning through gpu atari emulation, 2020. URL `https://github.com/NVlabs/cule`.

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. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures, 2018.

Lasse Espeholt, Raphaël Marinier, Piotr Stanczyk, Ke Wang, and Marcin Michalski. Seed rl: Scalable and efficient deep-rl with accelerated central inference, 2020. URL `https://github.com/google-research/seed_rl`.

C. Daniel Freeman, Erik Frey, Anton Raichuk, Sertan Girgin, Igor Mordatch, and Olivier Bachem. Brax – a differentiable physics engine for large scale rigid body simulation, 2021. URL `https://github.com/google/brax`.

Shixiang Gu, Ethan Holly, Timothy Lillicrap, and Sergey Levine. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 3389–3396. IEEE, 2017.

Matteo Hessel, Manuel Kroiss, Aidan Clark, Iurii Kemaev, John Quan, Thomas Keck, Fabio Viola, and Hado van Hasselt. Podracer architectures for scalable reinforcement learning, 2021.

Matt Hoffman, Bobak Shahriari, John Aslanides, Gabriel Barth-Maron, Feryal Behbahani, Tamara Norman, Abbas Abdolmaleki, Albin Cassirer, Fan Yang, Kate Baumli, Sarah Henderson, Alex Novikov, Sergio Gómez Colmenarejo, Serkan Cabi, Caglar Gulcehre, Tom Le Paine, Andrew Cowie, Ziyu Wang, Bilal Piot, and Nando de Freitas. Acme: A research framework for distributed reinforcement learning, 2020. URL `https://github.com/deepmind/acme`.

Julian Ibarz, Jie Tan, Chelsea Finn, Mrinal Kalakrishnan, Peter Pastor, and Sergey Levine. How to train your robot with deep reinforcement learning: lessons we have learned. *The International Journal of Robotics Research*, 40(4–5):698–721, January 2021. ISSN 1741-3176. doi: 10.1177/0278364920987859. URL `http://dx.doi.org/10.1177/0278364920987859`.

Tian Lan and Qi An. Discovering catalytic reaction networks using deep reinforcement learning from first-principles. *J. Am. Chem. Soc.*, 143(40):16804, 2021. URL `https://pubs.acs.org/doi/abs/10.1021/jacs.1c08794`.

Tian Lan, Sunil Srinivasa, Huan Wang, and Stephan Zheng. Warpdrive: Extremely fast end-to-end deep multi-agent reinforcement learning on a gpu, 2021.

Tian Lan, Sunil Srinivasa, Huan Wang, and Stephan Zheng. Warpdrive: Fast end-to-end deep multi-agent reinforcement learning on a gpu. *Journal of Machine Learning Research*, 23(316):1–6, 2022. URL `http://jmlr.org/papers/v23/22-0185.html`.

Tian Lan, Huan Wang, and Qi An. Massively high-throughput deep reinforcement learning with first principles: A generalizable approach to investigating catalytic reaction mechanisms (in review), 2024.

Viktor Makoviychuk, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller, Nikita Rudin, Arthur Allshire, Ankur Handa, and Gavriel State. Isaac gym: High performance gpu-based physics simulation for robot learning, 2021. URL `https://github.com/NVIDIA-Omniverse/IsaacGymEnvs`.

Johannes T. Margraf, Hyunwook Jung, Christoph Scheurer, and Karsten Reuter. Exploring catalytic reaction networks with machine learning. *Nature Catalysis*, 6:112–121, 2023. doi: 10.1038/s41929-022-00896-y.

Lisiane V. Mattos, Gary Jacobs, Burtron H. Davis, and Fábio B. Noronha. Production of hydrogen from ethanol: Review of reaction mechanism and catalyst deactivation. *Chemical Reviews*, 112(7):4094–4123, 2012. doi: 10.1021/cr2000114. URL `https://doi.org/10.1021/cr2000114`. PMID: 22617111.

Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning, 2016.

Nousiainen, J., Rajani, C., Kasper, M., Helin, T., Haffert, S. Y., Vérinaud, C., Males, J. R., Van Gorkom, K., Close, L. M., Long, J. D., Hedglen, A. D., Guyon, O., Schatz, L., Kautz, M., Lumbres, J., Rodack, A., Knight, J. M., and Miller, K. Toward on-sky adaptive optics control using reinforcement learning - model-based policy optimization for adaptive optics. *AA*, 664:A71, 2022. doi: 10.1051/0004-6361/202243311. URL `https://doi.org/10.1051/0004-6361/202243311`.

OpenAI. Openai five. `https://blog.openai.com/openai-five/`, 2018.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback, 2022.

Arnu Pretorius, Kale-ab Tessera, Andries P. Smit, Claude Formanek, St John Grimbly, Kevin Eloff, Siphelele Danisa, Lawrence Francis, Jonathan Shock, Herman Kamper, Willie Brink, Herman Engelbrecht, Alexandre Laterre, and Karim Beguir. Mava: a research framework for distributed multi-agent reinforcement learning, 2021. URL `https://github.com/instadeepai/Mava`.

Minhua Shao, Qiaowan Chang, Jean-Pol Dodelet, and Regis Chenitz. Recent advances in electrocatalysts for oxygen reduction reaction. *Chemical Reviews*, 116(6):3594–3657, 2016. doi: 10.1021/acs.chemrev.5b00462. URL `https://doi.org/10.1021/acs.chemrev.5b00462`. PMID: 26886420.

Xiangcheng Shi, Xiaoyun Lin, Ran Luo, Shican Wu, Lulu Li, Zhi-Jian Zhao, and Jinlong Gong. Dynamics of heterogeneous catalytic processes at operando conditions. *JACS Au*, 1(12):2100–2120, 2021. doi: 10.1021/jacsau.1c00355. URL `https://doi.org/10.1021/jacsau.1c00355`.

Yujin Tang, Yingtao Tian, and David Ha. Evojax: Hardware-accelerated neuroevolution, 2022. URL `https://github.com/google/evojax`.

Alexander Trott, Sunil Srinivasa, Douwe van der Wal, Sebastien Haneuse, and Stephan Zheng. Building a foundation for data-driven, interpretable, and robust policy design using the ai economist, 2021. URL `https://github.com/salesforce/ai-economist`.

Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575 (7782):350–354, 2019.

Sarod Yatawatta. Hint assisted reinforcement learning: an application in radio astronomy, 2023.

Junwoong Yoon, Zhonglin Cao, Rajesh K Raju, Yuyang Wang, Robert Burnley, Andrew J Gellman, Amir Barati Farimani, and Zachary W Ulissi. Deep reinforcement learning for predicting kinetic pathways to surface reconstruction in a ternary alloy. *Machine Learning: Science and Technology*, 2(4):045018, aug 2021. doi: 10.1088/2632-2153/ac191c. URL `https://dx.doi.org/10.1088/2632-2153/ac191c`.

Kaiqing Zhang, Alec Koppel, Hao Zhu, and Tamer Başar. Global convergence of policy gradient methods to (almost) locally optimal policies, 2020.

Tianyu Zhang, Andrew Williams, Soham Phade, Sunil Srinivasa, Yang Zhang, Prateek Gupta, Yoshua Bengio, and Stephan Zheng. Ai for global climate cooperation: Modeling global climate negotiations, agreements, and long-term cooperation in rice-n, 2022. URL `https://github.com/mila-iqia/climate-cooperation-competition`.

Stephan Zheng, Alexander Trott, Sunil Srinivasa, David C. Parkes, and Richard Socher. The ai economist: Taxation policy design via two-level deep multiagent reinforcement learning. *Science Advances*, 8(18):eabk2607, 2022. doi: 10.1126/sciadv.abk2607. URL `https://www.science.org/doi/abs/10.1126/sciadv.abk2607`.

Appendix A. Scalable Reinforcement Learning

Common scalable RL systems often employ a combination of distributed roll-out and trainer workers. Roll-out workers execute the environment to produce roll-outs, utilizing actions sampled from policy models on either roll-out workers or trainer workers. Typically, roll-out workers operate on CPU machines, occasionally utilizing GPU machines for richer environments.(Pretorius et al., 2021; Hoffman et al., 2020; Espeholt et al., 2018). Trainer workers gather roll-out data asynchronously from roll-out workers and iteratively optimize policies on either CPU or GPU machines. While such a distributed design is scalable, worker communication and data transfer cost is expensive and individual machine utilization can be poor. To improve performance, GPU and TPU-based RL frameworks exist (Tang et al., 2022; Hessel et al., 2021), but have focused on single-agent and domain-specific environments, e.g., for Atari (Dalton et al., 2020), or learning robotic control in 3-D rigid-body simulations (Freeman et al., 2021; Makoviychuk et al., 2021). Consequently, building efficient RL pipelines for simulations with intricate agent interactions, substantial data consumption, and diverse environments, as usually seen in scientific research, remains a challenging endeavor.

Appendix B. Details of Architecture

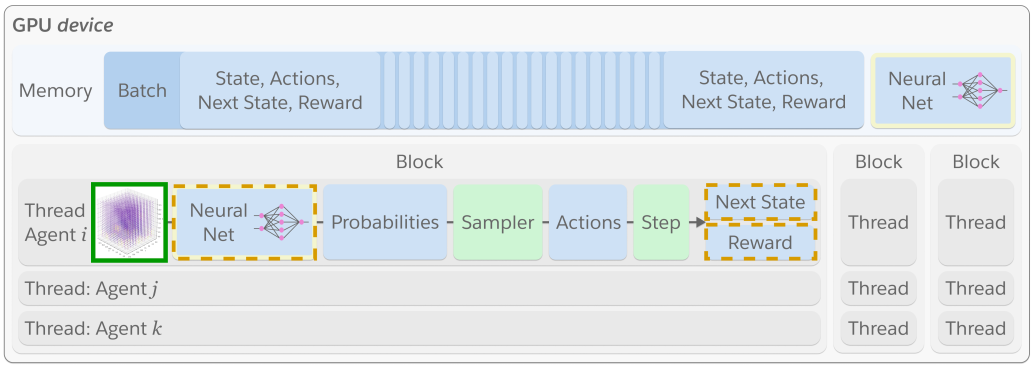

Figure 1: A flow chart depicting *WarpSci*. Computations within this framework are organized into GPU blocks, each comprising multiple threads to facilitate concurrent environment roll-outs. Each thread is responsible for operating an agent that samples actions and computes rewards. These blocks have access to the global GPU memory, which houses the RL environment (depicted as a 3-D grid in a green-bordered box) with local variations, and deep policy models. Additionally, they store in-place roll-out data for training purposes. The dashed brown boxes represent references (not copies) of the policy model objects and data placeholders managed by blocks and hosted in the global memory. Users have the flexibility to compose and upload their custom environment setups to finalize the environment construction.

As shown in Fig. 1, *WarpSci* executes the entire RL workflow seamlessly on a single GPU or multiple GPUs, utilizing a unified data storage hosted within the GPU for simulation roll-outs, action inference, reset and training. This approach minimizes CPU-GPU data communication and eliminates the need for additional data transfer within the GPU, resulting in a substantial reduction in both simulation and training times. Furthermore, our framework achieves parallelization at low cost by concurrently running thousands of single-agent or multi-agent simulations, capitalizing on the inherent parallel processing capabilities of GPUs. Each environment instance operates independently within a dedicated GPU block. Within each block, individual agents run on unique GPU threads, enabling interactions across threads. Each instance maintains a reference (not a copy) to the environment with local variations or random configurations, significantly reducing the storage overhead associated with the environment setup.

WarpSci offers simple Python classes located on the CPU to streamline all relevant CPU-GPU communication and interactions essential for RL. These classes connect to the CUDA back-end and offer simple APIs for constructing high-level Python applications. Users only need supply the *step* function to finalize the custom environment definition. As a default environment composer, we employ Numba, a user friendly, just-in-time compiler for Python. Finally, our framework automatically loads and integrates the environment *step* into the environment-agnostic CUDA backend for the RL simulation.

Appendix C. Example Details

All experiments ran on a single Nvidia A100 GPU, *a2-highgpu-1g*, on the Google Cloud Platform.

Classic Control. In the field of RL, classic control environments usually serve as fundamental benchmarks to evaluate the performance of various RL algorithms and systems. These environments typically involve simple physics-based systems, yet their challenges lie in achieving stable and optimal control. Iconic examples, such as CartPole and Acrobot in *gym* environment(Brockman et al., 2016), offer controlled scenarios with well-defined dynamics, making them ideal for benchmarking the throughput scalability and the learning capability of *WarpSci*.

Fig. 2(a) shows that *WarpSci*'s performance in classic control environments scales linearly to 10K of environments, yielding perfect parallelism. For example, *WarpSci* runs at 8.6 million environment steps per second with 10K Cartpole-v1 or Acrobot-v1 environments. Fig. 2(b) and (c) displays the convergence speed of *WarpSci* as a function of the number of environment replicas running in parallel. The data reveal that, under consistent fixed hyperparameters, the simulations operating with an increased number of concurrent environments attain global convergence faster and more stably. Particularly, simulations with 10K Cartpole and Acrobot environment replicas reach the global optimum within 30 and 5 minutes respectively, while 10 environment replicas can barely exhibit satisfactory convergence in such a short period.

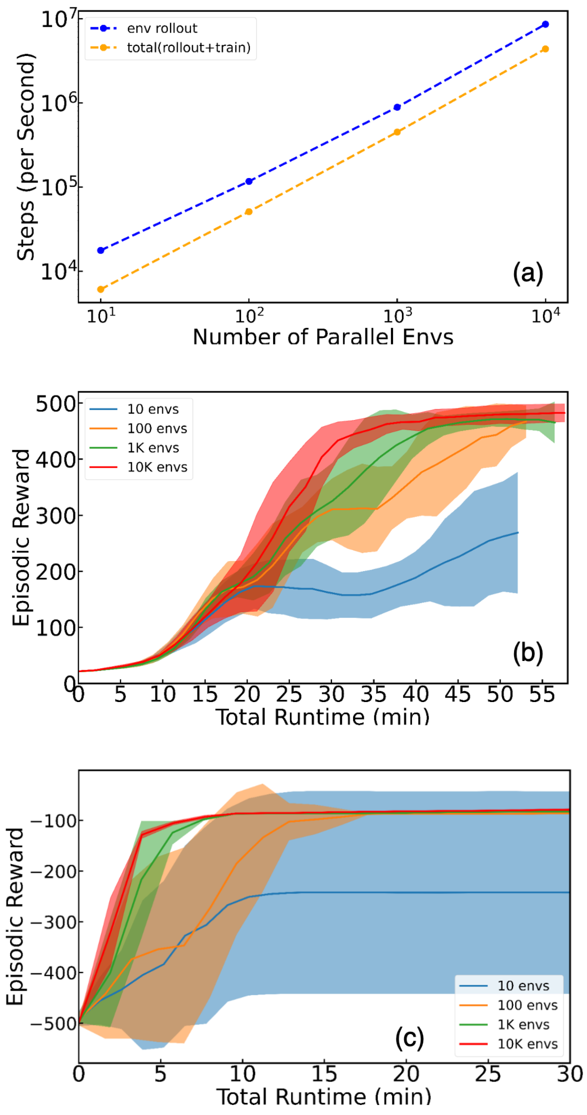

Figure 2: Scalability, convergence and learning speed for *WarpSci* applied to *gym* classic control environments. (a) Roll-out and training throughput in Cartpole-v1 and Acrobot-v1 versus the number of parallel environments (log-log scale) to 10K concurrent environments with random local initialization: the throughput scales linearly. The average episodic reward (the accumulated total reward collected from the start to the terminal state) versus the training time (wall-clock minutes) for (b) Cartpole-v1 and (c) Acrobot-v1 running at various concurrency levels. The model was trained on a single Nvidia A100 GPU. For robustness, the depicted results are averaging over eight independent runs from scratch with different initialization seeds and the same hyperparameters. The shadow regions represent the error bar of eight independent runs.

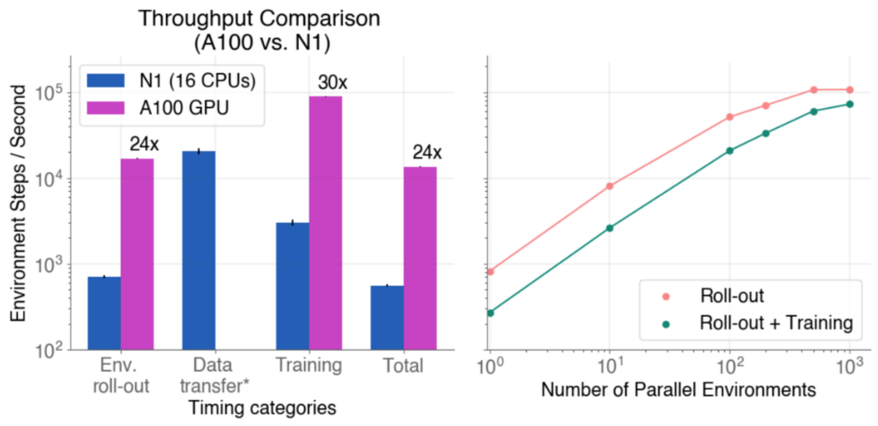

Figure 3: *WarpSci* performance in the COVID-19 economic simulation in log scale. Left: Note that there is no data transfer with *WarpSci*. With 60 parallel environments, *WarpSci* achieves 24 times higher throughput over CPU-based distributed training architectures ("total"). Moreover, both the roll-out and training phase are an order of magnitude faster than on the distributed N1 node. Right: Environment steps per second and end-to-end training speed scale almost linearly with the number of environments. (Credit: Lan et al. (2021))

Multi-Agent Economics. We demonstrate the scalability of *WarpSci* to more intricate environments through its evaluation in a COVID-19 simulation. This simulation, grounded in real-world data, models the interplay between health and economic dynamics during the COVID-19 pandemic. Notably, the simulation step is significantly more complex compared to the *gym* classic control problems, consuming a larger fraction of each iteration's runtime.

The simulation involves 52 agents, with 51 representing governors for each U.S. state and Washington D.C., and an additional agent for the federal government of the USA. This constitutes a complex two-level multi-agent environment, where state agents determine the stringency level of the pandemic response, and the federal government provides subsidies to eligible individuals. The actions of each agent influence health and economic outcomes, such as deaths, unemployment, and GDP. Moreover, the federal government's actions can alter the health-economic trade-off and optimization objective for the U.S. states, rendering it a complex and dynamic two-level RL problem. Interested readers seeking additional scientific background and technical details are encouraged to refer to Trott et al. (2021); Zheng et al. (2022).

For this study, *WarpSci* achieves 24 times higher throughput with 60 environment replicas, compared to a 16 CPU node, *n1-standard-16*, on the Google Cloud Platform. Across different timing categories as shown in Fig. 3, the performance gains comprise a 24 times speed-up during the environment roll-out, a zero data transfer time, and a 30 times speed-up for training the policy models. Moreover, *WarpSci* can scale almost linearly to 1K parallel COVID-19 environments, resulting in even higher throughput gains.

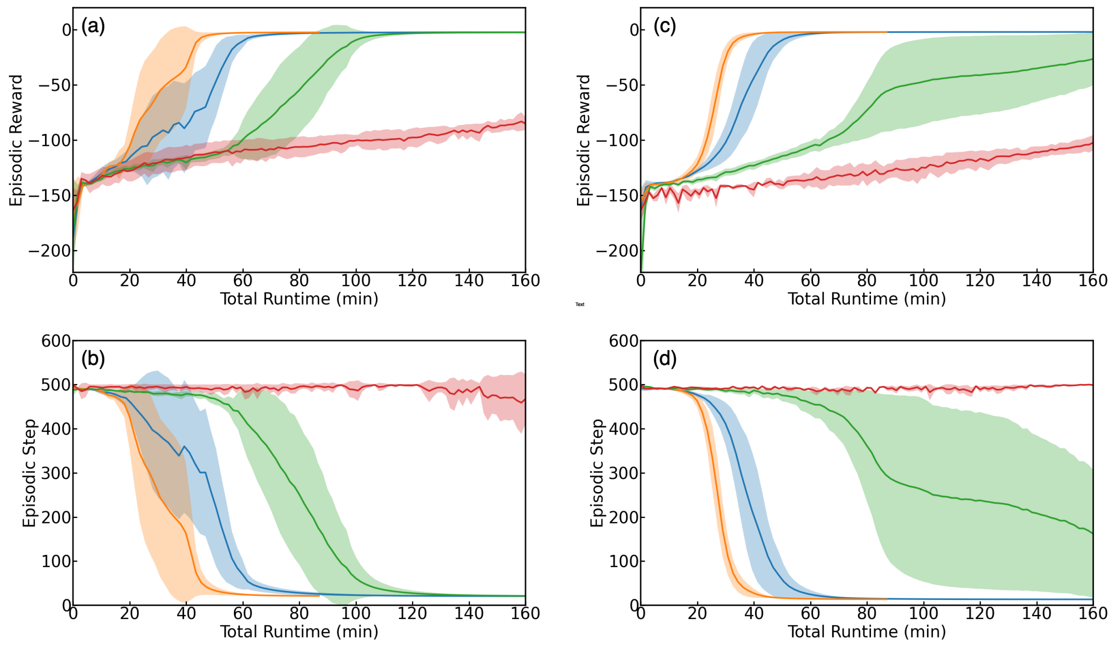

Figure 4: Convergence and learning speed, quantified by total runtime in wall-clock minutes, were assessed for Langmuir-Hinshelwood (a, b) and Eley-Rideal (c, d) hydrogenation reactions of NH_2 to NH_3. Varied numbers of concurrent environment instances were employed: 4 in red, 20 in green, 100 in blue, and 500 in yellow. The episodic reward denotes the mean accumulated reward that H atom actors gather from the initial to the terminal state of (a) Langmuir-Hinshelwood and (c) Eley-Rideal. Episodic step indicates the average total steps to reach the terminal state for (b) Langmuir-Hinshelwood and (d) Eley-Rideal. Training utilized a single Nvidia A100 GPU. For robustness, the displayed results are averages over five independent runs from scratch with different initialization seeds and identical hyperparameters. Shadow regions depict the error bars of the five independent runs. (Credit: Lan et al. (2024))

Catalytic Reactions. Comprehending catalytic reaction pathways is essential for advancing our understanding of chemical processes, refining conditions, and designing robust catalysts. These pathways offer insights into reaction mechanisms, facilitating the creation of more selective catalysts (Mattos et al., 2012; Shao et al., 2016). However, exploring these pathways presents significant challenges, including the complexity of multi-step reactions, short-lived intermediates, and experimental intricacies (Chen et al., 2021; Shi et al., 2021; Lan and An, 2021). RL shows promise in overcoming these challenges by providing an automated approach to navigating reaction networks. However, RL encounters scientific and engineering obstacles, primarily limited by simulation throughput. Consequently, current RL research in chemical reactions often concentrates on specific reactions, relying on model simplifications with state vector encodings or heuristic rules. This approach limits generalizability and requires substantial empirical design. Exploration is also confined to predefined sets of reaction networks, hindering the discovery of unknown mechanisms (Yoon et al., 2021; Lan and An, 2021; Margraf et al., 2023). Therefore, the pursuit of a more versatile RL solution to explore undiscovered reaction mechanisms remains a significant challenge in the field.

In this study, we present a reaction-agnostic methodology facilitated by *WarpSci*. The RL environment is constructed solely based on the potential energy landscape derived from first principles. This approach intrinsically defines the chemical reaction environment as a function of atomic positions, eliminating the necessity for laborious empirical or semi-empirical design of reaction-specific representations in RL environments. The outstanding generalizability and training speed are supported by the remarkable high-throughput capacity enabled by our architecture.

We forecast the reaction pathway for the crucial hydrogenation step in the Haber-Bosch (H-B) process on the Fe(111) surface. The H-B process holds a pivotal role in Earth's nitrogen cycle and represents over 2 percent of global energy consumption, yielding 160 million tons of ammonia annually. Despite a century of concentrated research to improve the H-B process, progress has been slow (Chen et al., 2018). Our framework has the potential to significantly contribute to process optimization, potentially reducing production costs and CO_2 emissions while enabling the establishment of smaller and more widespread plants.

Figure 4 displays the convergence speed as *WarpSci* processes the Langmuir-Hinshelwood reaction as a function of the number of environment replicas, running in parallel. The data reveal that, under consistent fixed hyperparameters, the simulations operating with an increased number of concurrent environments attain global convergence faster and more stably. The generalizable RL environment with the same hyperparameters is directly applicable to the study of Eley-Rideal reaction mechanism. The results highlight the critical role of massively high data throughput in RL for effectively exploring a broad range of reaction mechanisms through a generalizable RL environment representation built solely upon atomic positions.

Our findings reveal that the Langmuir-Hinshelwood mechanism shares the same transition state as the Eley-Rideal mechanism for H migration to NH2, forming ammonia. Furthermore, the reaction path identified by our model exhibits a lower energy barrier compared to that through nudged elastic band calculation. In this *Extended Abstract*, we focus on presenting the generalizability, training speed and convergence stability facilitated by the high throughput of *WarpSci*. Interested readers seeking additional scientific background and technical details of this study are encouraged to refer to Lan and An (2021); Lan et al. (2024); Margraf et al. (2023).