gym-saturation: Gymnasium environments for saturation provers (system description) *

Boris Shminke

Université Côte d'Azur, CNRS, LJAD, France
boris.shminke@univ-cotedazur.fr

Introduction This work describes a new version (0.10.0, released 2023.04.25) of a previously published [15] Python package — gym-saturation: a collection of OpenAI Gym [3] environments for guiding saturation-style provers with reinforcement learning (RL) algorithms. The new version partly implements the ideas of our project proposal [16]. The main changes from the previous release (0.2.9, on 2022.02.26) are:

- guiding two popular provers instead of a single experimental one
- pluggable first-order logic formulae embeddings support
- examples of experiments with different RL algorithms
- following the updated Gymnasium [19] API instead of the outdated OpenAI Gym

gym-saturation works with Python 3.8+. One can install it by pip install gym-saturation or conda install -c conda-forge gym-saturation. Then, provided Vampire and/or iProver binaries are on PATH, one can use it as any other Gymnasium environment:

```
import gymnasium
import gym_saturation
env = gymnasium.make("Vampire-v0")  # or "iProver-v0"
# edit and uncomment the following line to set a non-default problem
# env.set_task("a-TPTP-problem-path")
observation, info = env.reset()
print("Starting proof state:")
env.render()
terminated, truncated = False, False
while not (terminated or truncated):
    # apply policy (e.g. a random available action)
    action = env.action_space.sample(mask=observation["action_mask"])
    print("Given clause:", observation["real_obs"][action])
    observation, reward, terminated, truncated, info = env.step(action)
print("Final proof state:")
env.render()
env.close()
```

*This work has been supported by the French government, through the 3IA Côte d'Azur Investment in the Future project managed by the National Research Agency (ANR) with the reference numbers ANR-19-P3IA-0002.

Related work Guiding provers with RL is a hot topic. Recent projects in this domain include TRAIL (Trial Reasoner for AI that Learns) [1], FLoP (Fiding Longer Proofs) [20], and lazyCoP [13]. lazyCoP guides a new prover created for that purpose, FLoP builds on fCoP [7], an OCaml rewrite of older leanCoP [9], and TRAIL relies on a modified E [14]. Contrary to that, `gym-saturation` works with unmodified stable versions of Vampire [8] and iProver [5].

Environment architecture We run Vampire in a manual clause selection mode [6]. Using Python package `pexpect`, we attach to Vampire's standard input and output, pass the action chosen by the agent to the former and read observations from the latter. iProver recently added support of being guided by external agents. An agent has to be a TCP server satisfying a particular API specification. To make it work with `gym-saturation`, we implemented a *relay server*. It accepts a long-running TCP connection from a running iProver thread, stores its requests to a thread-safe queue, and pops a response from another such queue filled by `gym-saturation` thread.

Representation subsystem To apply any deep RL algorithm, one needs a representation of the environment state in a tensor form first. In [10], the authors proposed a particular neural network architecture they called *Recursive Tree Grammar Autoencoders (RTG-AE)*, which encodes abstract syntax trees produced by a programming language parser into real vectors. They also published the pre-trained model for Python [11]. To make use of it for our purpose, we furnished several technical improvements to their code (our contribution is freely available [1]):

- a TorchServe [12] handler for HTTP POST requests for embeddings
- request caching with the Memcached server [4]
- Docker container to start the whole subsystem easily on any operating system

To integrate the `ast2vec` server with `gym-saturation` environments, we added several Gymnasium observation wrappers, transforming a clause in the TPTP [18] language to a Python script.

Experiment examples We provide examples of experiments easily possible with `gym-saturation` as a supplementary code to this paper [2]. We don't consider these experiments as being of any scientific significance per se, serving merely as illustrations and basic usage examples. We coded these experiments in the Ray framework, which includes an RLlib — a library of popular RL algorithms. In the experiments, we try to solve `SET001-1` from the TPTP by limiting the maximal number of clauses in a proof state to 20. In one experiment, we organise clauses in two priority queues (by age and weight) and use an action wrapper to map from a queue number (0 or 1) to the clause number. It transforms our environment into a semblance of a 2-armed bandit, and we use Thompson sampling [2] to train. This experiment reflects ideas similar to those described in [17]. In another experiment, we use `ast2vec` server for getting clause embeddings and train a Proximal Policy Optimisation (PPO) algorithm as implemented in the Ray RLlib. Such an approach is more similar to [20].

Acknowledgements We thank Konstantin Korovin for the productive discussion and for adding the external agents' communication feature to iProver.

[1] https://gitlab.com/inpefess/ast2vec
[2] https://github.com/inpefess/ray-prover/releases/tag/v0.0.3

References

[1] Ibrahim Abdelaziz, Maxwell Crouse, Bassem Makni, Vernon Austel, Cristina Cornelio, Shajith Ikbal, Pavan Kapanipathi, Ndivhuwo Makondo, Kavitha Srinivas, Michael Witbrock, and Achille Fokoue. Learning to Guide a Saturation-Based Theorem Prover. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(1):738–751, 2023.

[2] Shipra Agrawal and Navin Goyal. Thompson Sampling for Contextual Bandits with Linear Payoffs. In Sanjoy Dasgupta and David McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 127–135, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR.

[3] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. OpenAI Gym. *arXiv*, abs/1606.01540, 2016.

[4] Danga Interactive, Inc. Memcached. https://github.com/memcached/memcached, mar 2023.

[5] André Duarte and Konstantin Korovin. Implementing Superposition in iProver (System Description). In Nicolas Peltier and Viorica Sofronie-Stokkermans, editors, *Automated Reasoning - 10th International Joint Conference, IJCAR 2020, Paris, France, July 1-4, 2020, Proceedings, Part II*, volume 12167 of *Lecture Notes in Computer Science*, pages 388–397. Springer, 2020.

[6] Bernhard Gleiss, Laura Kovács, and Lena Schnedlitz. Interactive Visualization of Saturation Attempts in Vampire. In Wolfgang Ahrendt and Silvia Lizeth Tapia Tarifa, editors, *Integrated Formal Methods*, pages 504–513, Cham, 2019. Springer International Publishing.

[7] Cezary Kaliszyk, Josef Urban, and Jiři Vyskočil. Certified Connection Tableaux Proofs for HOL Light and TPTP. In *Proceedings of the 2015 Conference on Certified Programs and Proofs*, CPP '15, page 59–66, New York, NY, USA, 2015. Association for Computing Machinery.

[8] Laura Kovács and Andrei Voronkov. First-Order Theorem Proving and Vampire. In Natasha Sharygina and Helmut Veith, editors, *Computer Aided Verification*, pages 1–35, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[9] Jens Otten and Wolfgang Bibel. leanCoP: lean connection-based theorem proving. *Journal of Symbolic Computation*, 36(1):139–161, 2003. First Order Theorem Proving.

[10] Benjamin Paaßen, Irena Koprinska, and Kalina Yacef. Recursive tree grammar autoencoders. *Machine Learning*, Aug 2022.

[11] Benjamin Paassen, Jessica McBroom, Bryn Jeffries, Irena Koprinska, and Kalina Yacef. Mapping Python Programs to Vectors using Recursive Neural Encodings. *Journal of Educational Data Mining*, 13(3):1–35, Oct. 2021.

[12] PyTorch Serve Contributors. TorchServe. https://github.com/pytorch/serve, feb 2023.

[13] Michael Rawson and Giles Reger. lazyCoP: Lazy Paramodulation Meets Neurally Guided Search. In Anupam Das and Sara Negri, editors, *Automated Reasoning with Analytic Tableaux and Related Methods*, pages 187–199, Cham, 2021. Springer International Publishing.

[14] Stephan Schulz, Simon Cruanes, and Petar Vukmirović. Faster, Higher, Stronger: E 2.3. In Pascal Fontaine, editor, *Automated Deduction – CADE 27*, pages 495–507, Cham, 2019. Springer International Publishing.

[15] Boris Shminke. gym-saturation: an OpenAI Gym environment for saturation provers. *Journal of Open Source Software*, 7(71):3849, 2022.

[16] Boris Shminke. Project proposal: A modular reinforcement learning based automated theorem prover. *arXiv*, abs/2209.02562, 2022.

[17] Martin Suda. Vampire Getting Noisy: Will Random Bits Help Conquer Chaos? (System Description). In Jasmin Blanchette, Laura Kovács, and Dirk Pattinson, editors, *Automated Reasoning*, pages 659–667, Cham, 2022. Springer International Publishing.

[18] Geoff Sutcliffe. The TPTP Problem Library and Associated Infrastructure - From CNF to TH0, TPTP v6.4.0. *Journal of Automated Reasoning*, 59(4):483–502, 2017.

[19] The Farama Foundation. Gymnasium. https://github.com/Farama-Foundation/Gymnasium, mar

2023.

[20] Zsolt Zombori, Adrián Csiszárik, Henryk Michalewski, Cezary Kaliszyk, and Josef Urban. Towards Finding Longer Proofs. In Anupam Das and Sara Negri, editors, *Automated Reasoning with Analytic Tableaux and Related Methods*, pages 167–186, Cham, 2021. Springer International Publishing.