Controller Synthesis from Deep Reinforcement Learning Policies

Florent Delgrange[1,2]* Guy Avni[3] Anna Lukina[4]
Christian Schilling[5] Ann Nowé[1] Guillermo A. Pérez[2,6]

[1]Vrije Universiteit Brussel, BE [2]UAntwerpen, BE [3]University of Haifa, IL
[4]TU Delft, NL [5]Aalborg University, DK [6]Flanders Make, BE

We consider the fundamental problem of constructing control *policies* for environments modeled as *Markov decision processes* (MDPs) with formal guarantees. We suggest a framework that combines two techniques with complementary benefits and drawbacks, which we describe next.

The first technique is *reinforcement learning* (RL) in which the designer chooses how rewards are issued, and policies are trained to optimize rewards. In particular, *deep RL* (DRL, e.g., [4]) is successful in domains of *high-dimensional feature spaces with unknown dynamics*, surpassing human capabilities. On the downside, designing a reward function is a challenging engineering task in which the designer needs to both train the agent to exhibit desired behavior and train it efficiently. Specifically, for long-term objectives, one needs to deal with the notorious problem of sparse rewards [2] by guiding the agent to the intended behavior [3]. This in turn, adds more problems as the "desired behavior" is specified via rewards, and reward engineering leads to behavior that may not align with the user's intentions.

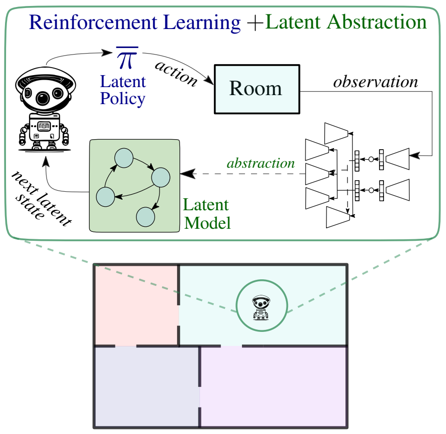

Fig. 1: The agent is trained to exit *each room*, in *every possible direction*. The training is done through *parallel simulations* where an abstraction of the environment is learned via NNs, yielding a latent model for each room. Simultaneously, a policy is learned via DRL on the learned latent representation, which guarantees the agent's low-level behavior conformity through PAC bounds.

The second technique is *reactive synthesis* [5], which constructs an optimal policy *based on a model of the environment* and *objectives specified as a logical formula*. In contrast to DRL, *synthesis provides guarantees that the policy satisfies the specification* and *allows users an intuitive and natural specification languages*. The reliance on an explicit environment model is its key disadvantage; the technique struggles with scalability and domains in which dynamics are partially known.

We propose a framework that aims to gain the best of both worlds. We require little prior knowledge of the structure of the environment: the input is a *map* given as a *graph*, where each vertex embeds an (unknown) room, modeled as an MDP. We argue this is a natural requirement in many domains. Think of a robot that need deliver a package in a warehouse divided into rooms amid moving obstacles (e.g., forklifts, workers, or other robots). While it is infeasible to provide a model describing all the possible interactions the agent may have within the warehouse and the dynamics of the moving obstacles, one can reasonably assume a *map* is provided.

Our framework proceeds as follows. We first train DRL policies to achieve *short-horizon, low-level objectives* in the rooms, e.g., act safely and exit a room via a designated target (Fig. 1). We then construct a high-level *planner* that chooses which policy acts in a room: based on the low-level policies and the given map, we apply synthesis to achieve a *long-horizon objective*, e.g., reach the target location (Fig. 2). A key challenge is obtaining an environment model for synthesis, i.e., a model of the operation of the low-level policies. We develop a novel DRL procedure that learns a *latent* model of *each room* where the satisfaction of the low-level objective can be formally verified.

To summarize, we present a novel framework that incorporates DRL into the synthesis process, which offers the following key advantages. First and foremost, it *provides guarantees on the operation of the controller.* As mentioned, it enjoys the best of both worlds: it *enables synthesis with theoretical guarantees in large partially-known environments.* It allows a "separation of concerns": reward engineering is only done locally while *high-level tasks are given in an intuitive specification language.* In addition, it offers a remedy for the notorious challenges of sparse rewards in RL. Interestingly, it also enables *reusability*: the policies in rooms and their guarantees are reusable across similar rooms and when the high-level task or structure change.

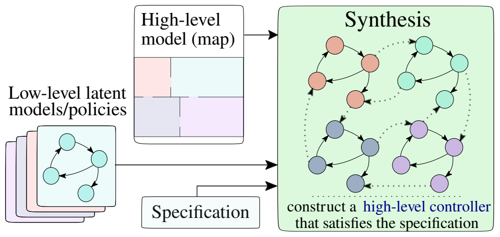
High-level model (map)
Synthesis
Low-level latent models/policies
Specification
construct a high-level controller that satisfies the specification

Fig. 2: Given (i) a high-level description of the environment, (ii) a collection of low-level models and policies for each room, and (iii) the specifications, synthesis outputs a high-level controller guaranteed to satisfy the specifications. The challenge resides in the way the low-level components are merged to apply synthesis while maintaining their guarantees.

A full version of this paper is available in [1].

Acknowledgements

We thank Sterre Lutz and Willem Röpke for providing valuable feedback during the preparation of this manuscript.

This research received support from the Belgian Flemish Government's AI Research Program and DESCARTES iBOF project, the Dutch Research Council (NWO) Talent Programme (VI.Veni.222.119), Independent Research Fund Denmark (10.46540/3120-00041B), DIREC - Digital Research Centre Denmark (9142-0001B), Villum Investigator Grant S4OS (37819), and ISF grant (1679/21).

References

1. Delgrange, F., Avni, G., Lukina, A., Schilling, C., Nowe, A., Perez, G.: Controller synthesis from deep reinforcement learning policies. In: Seventeenth European Workshop on Reinforcement Learning (2024), `https://openreview.net/forum?id=KDiCsArAKs`
2. Ladosz, P., Weng, L., Kim, M., Oh, H.: Exploration in deep reinforcement learning: A survey. Inf. Fusion **85**, 1–22 (2022). `https://doi.org/10.1016/J.INFFUS.2022.03.003`, `https://doi.org/10.1016/j.inffus.2022.03.003`
3. Liu, M., Zhu, M., Zhang, W.: Goal-conditioned reinforcement learning: Problems and solutions. In: IJCAI. pp. 5502–5511. ijcai.org (2022). `https://doi.org/10.24963/IJCAI.2022/770`, `https://doi.org/10.24963/ijcai.2022/770`
4. Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., Riedmiller, M.A.: Playing atari with deep reinforcement learning. CoRR **abs/1312.5602** (2013), `http://arxiv.org/abs/1312.5602`
5. Pnueli, A., Rosner, R.: On the synthesis of a reactive module. In: POPL. pp. 179–190. ACM Press (1989). `https://doi.org/10.1145/75277.75293`, `https://doi.org/10.1145/75277.75293`