Self-monitoring Adaptive AI Agents Operating in Open Worlds

Wiktor Piotrowski[1], **Roni Stern**[2], **Sachin Grover**[1], **Shiwali Mohan**[1]
[1] PARC, a part of SRI International, CA, USA
[2] Ben-Gurion University of the Negev, Beer-Sheva, Israel
{wiktor.piotrowski,sachin.grover,shiwali.mohan}@sri.com, sternron@bgu.ac.il

Artificial Intelligence (AI) and Machine Learning (ML) research on sequential decision-making usually relies on the assumption of a static world. That is, all relevant characteristics of the environment are known ahead of deployment, during agent design time and remain unchanged during runtime. However, the real world is open, evolving, and can change without any prior indication. Online adaptation and learning is a desirable property of any real-world AI system. Prominent model-free learning techniques such as reinforcement learning acquire knowledge in non-interpretable representations (as network weights, biases etc.), posing significant challenges in assessment, validation, and regulation of what has been learned. Our research explores open-world adaptation in model-based reasoning systems which encode knowledge explicitly. Explicit representation of acquired knowledge enables inspection of what has been learned, supporting assessment, validation, and regulation of the system's evolving behavior in an open world.

In this paper, we study model-based reasoning in intelligent agents that can robustly operate in environments whose characteristics change while the agent is operational. We term such a shift in environmental characteristics as a *novelty* (Boult et al. 2021). Novelties pose a significant challenge to the deployment of intelligent agents. For model-based agents, the knowledge about the environment they rely on may be incomplete, incorrect, or outdated due to a novelty. For model-free learning agents, the policy they learned may be ineffective due to a novelty, and they may need numerous interactions with the environment to learn a new policy. Consequently, both types of agents may fail catastrophically during deployment. An effective *open world* agent should autonomously *detect* when a novelty has been introduced in the environment, *characterize* it, as it pertains to what it knows about the environment, and then *accommodate* it by changing its decision-making strategies. Ideally, it *transfers* relevant operational knowledge from before novelty is introduced to after, i.e., it learns without fully retraining and in orders of magnitude less time. This challenge of designing such an open-world agent, where novelties can appear at an unspecified time, has been gaining significant interest in the literature (Senator 2019; Kejriwal et al. 2022).

To meet this challenge, we present HYDRA a domain-independent architecture for implementing a novelty-aware agent in complex, mixed discrete and continuous domains

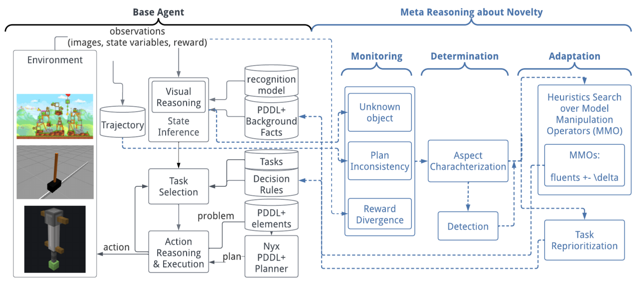

Figure 1: HYDRA: a self-monitoring adaptive architecture

(Mohan et al. 2023; Piotrowski et al. 2023a,b; Stern et al. 2022; Piotrowski et al. 2021). The HYDRA architecture includes a *base agent* and *novelty meta-reasoning* components designed to detect novelties and adapt the base agent's behavior to them. A notional architecture is shown in Figure 1. The base agent (Figure 1 - left) implements a *perceive-decide-act* cycle in the environment. The novelty meta-reasoning components in HYDRA (Figure 1 - right) detect the presence of novelty, characterize, and accommodates it by updating its knowledge bases.

Base Agent The HYDRA base agent follows standard agent architectures such as Belief-Desire-Intention (BDI) (Georgeff et al. 1999), Soar (Laird 2019), and ICARUS (Choi and Langley 2018). Its main components are: (1) *state inference*, which maintains and updates beliefs about the current state of the environment and augments them with background assumptions; (2) *task selection*, which selects the intermediate tasks the agent intends to perform; and (3) *action reasoning and execution*, which uses a planner to determine the sequence of actions to execute in order to perform the selected task. Each component reasons about the current beliefs using a variety of long-term knowledge encoded in the agent (shown in cylinders). HYDRA can use any type of planner to determine the sequence of actions to execute in order to perform the active task. To support rich environments with complex dynamics and discrete and numeric state variables, we implemented HYDRA with a planner that accepts domains specified in PDDL+ (Fox and Long 2006). This PDDL+ model defines the composition of the target system and its evolution via various *happenings* that include actions the agent may perform, exogenous events that may be triggered, and durative

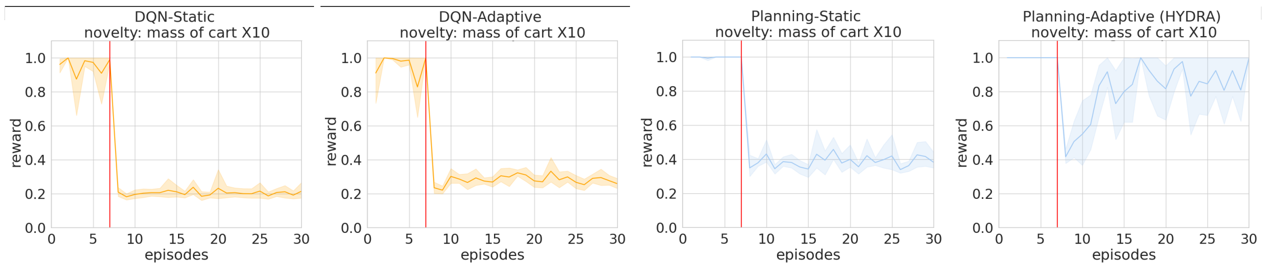

Figure 2: Adaptation performance of static and adaptive version of planning and DQN agents in CartPole++. Red line denotes the introduction of novelty which *increases the mass of the cart 10-fold*.

processes that may be active. To reason and plan in the environment, the HYDRA base agent first generates a PDDL+ planning problem representing the goal it aims to achieve and the current state. Then, HYDRA solves the resulting planning task, yielding a valid temporal plan. The plan is then sequentially executed in the environment, and the execution trace is collected for future analysis.

Novelty Meta Reasoner HYDRA introduces a novel a meta-reasoning process to detect, characterize, and accommodate novelties. The process maintains explicit expectations about the agent's observations, transitions in the state space due to agent actions and extraneous dynamics, as well as its performance. Their violation indicates that a novelty has been introduced in the environment which must be further inspected, characterized, and accommodated for. Introducing such a process frames learning as a volitional activity undertaken by the agent to which resources are devoted only when an opportunity presents itself (as indicated by violation of expectations). This is in stark contrast with the classical machine learning setup in which agent learning is controlled externally. Specifically, HYDRA (Figure 1) implements (1) a set of *novelty monitors* that maintain a variety of explicit expectations about the environment evolution and monitor divergence; (2) *novelty determination*, that aggregates the information from the monitors and determines if a novelty has been introduced and requires adaptation; and (3) a set of *novelty adaptation strategies*.

Novelty Monitors and Determination Novelty monitors are a set of components that maintain explicit expectations about various aspects of agent behavior in the environment. They capture violations of those expectations which are processed by further downsteram reasoning. Monitors are dedicated to different parts in the base agent reasoning cycle and are implemented using diverse computational techniques. We implemented several monitors:

- **Unknown Objects and Entities.** This monitor encodes an expectation that all observed entities processed by the state inference module are *known* i.e., they are of a type that is encoded in the agent's planning model.
- **Plan Inconsistency.** This monitor measures how accurately the observed environment dynamics, i.e., the behavior of the different happenings (actions, effects, and processes), match the agent's internal domain model. This novelty monitor is thus geared towards detecting novelties that change these environment transitions.
- **Reward Divergence.** This monitor maintains expectations about the quality of its own performance. Any change in performance quality can indicate that a novelty has been introduced in the environment. This monitor is applicable in environments that supply an explicit reward signal, e.g., game environments that return a score.

Each novelty monitor generates information about the existence of novelty based on various aspects of agent behavior (unknown objects in the observation space, environment transitions, and performance quality). The novelty determination component collects information from all novelty monitors as well as some other observations about the agent (e.g., success or failure at the overall task) and determines if novelty has been introduced and the agent's domain needs to be adapted. Upon exploring several approaches, we eventually implemented domain-specific decision rules.

Adaptation Strategy The main adaptation strategy adopted by HYDRA is a *heuristic search-based model repair* approach that manipulates the base agent's PDDL+ model to be consistent with the detected novelty. The search-based model repair adaptation strategy works by searching for a *domain repair*, which is a sequence of model modifications that, when applied to the agent's internal domain, returns a domain that is consistent with the observations. To find such a domain repair, repair algorithm accepts as input a set possible basic *Model Manipulation Operators* (MMOs) and outputs a sequence of one or more basic MMOs. An example of an MMO is to add a fixed amount to one of the numeric domain constants.

Evalaution We implemented HDYRA on three research domains: CartPole++ (a higher dimension variant of a classic control problem CartPole), ScienceBirds (an AI competition domain for the Angry Birds game), and PogoStick (a task in the Minecraft game). Our results show that for certain types of novelties, HYDRA agents can adapt *quickly* with few interactions with the environment. Additionally, the adaptations produced by HYDRA are *interpretable* by design - they are represented in terms of changes to the elements of its model (shown below), enabling inspection of proposed changes.

Repair 1: `mass_cart:0,length_pole:0.3,mass_pole:0,force_mag:0,gravity:0,angle_limit:0,x_limit:0`
Repair 2: `mass_cart:0,length_pole:0,mass_pole:0,force_mag:0, gravity:1.0,angle_limit:0,x_limit:0`
This property of a HYDRA agent is a considerable advantage when developing adaptive systems that can be assessed and validated by human experts. Figure 2 shows example results for Cartpole++ domain compared with static, DQN based adaptive, planning based static and HYDRA agents.

References

Boult, T.; Grabowicz, P.; Prijatelj, D.; Stern, R.; Holder, L.; Alspector, J.; Jafarzadeh, M.; Ahmad, T.; Dhamija, A.; Li, C.; et al. 2021. Towards a Unifying Framework for Formal Theories of Novelty. In *AAAI Conference on Artificial Intelligence*, 15047–15052.

Choi, D.; and Langley, P. 2018. Evolution of the ICARUS Cognitive Architecture. *Cognitive Systems Research*, 48: 25–38.

Fox, M.; and Long, D. 2006. Modelling Mixed Discrete-Continuous Domains for Planning. *Journal of Artificial Intelligence Research*, 27: 235–297.

Georgeff, M.; Pell, B.; Pollack, M.; Tambe, M.; and Wooldridge, M. 1999. The Belief-Desire-Intention Model of Agency. In *Intelligent Agents V: Agents Theories, Architectures, and Languages*, 1–10. Springer.

Kejriwal, M.; Shrivastava, A.; Kildebeck, E.; Bhargava, B.; and Vondrick, C. 2022. Designing Artificial Intelligence for Open Worlds. https://usc-isi-i2.github.io/AAAI2022SS/.

Laird, J. E. 2019. *The SOAR Cognitive Architecture*. MIT press.

Mohan, S.; Piotrowski, W.; Stern, R.; Grover, S.; Kim, S.; Le, J.; and De Kleer, J. 2023. A Domain-Independent Agent Architecture for Adaptive Operation in Evolving Open Worlds. *arXiv preprint arXiv:2306.06272*.

Piotrowski, W.; Sher, Y.; Grover, S.; Stern, R.; and Mohan, S. 2023a. Heuristic Search For Physics-Based Problems: Angry Birds in PDDL+. In *International Conference on Automated Planning and Scheduling*, 518–526.

Piotrowski, W.; Stern, R.; Klenk, M.; Perez, A.; Mohan, S.; de Kleer, J.; and Le, J. 2021. Playing Angry Birds with a Domain-Independent PDDL+ Planner. *International Conference on Automated Planning and Scheduling (Demo Track)*.

Piotrowski, W.; Stern, R.; Sher, Y.; Le, J.; Klenk, M.; deKleer, J.; and Mohan, S. 2023b. Learning to Operate in Open Worlds by Adapting Planning Models. *AAMAS International Conference on Autonomous Agents and Multiagent Systems*.

Senator, T. 2019. Science of AI and Learning for Openworld Novelty (SAIL-ON). Technical report, DARPA.

Stern, R.; Piotrowski, W.; Klenk, M.; de Kleer, J.; Perez, A.; Le, J.; and Mohan, S. 2022. Model-Based Adaptation to Novelty for Open-World AI. In *Proceedings of the ICAPS Workshop on Bridging the Gap Between AI Planning and Learning*.