A Demonstration of Refinement Acting, Planning and Learning System Using Operational Models

Sunandita Patra,[1] **James Mason,**[1] **Malik Ghallab,**[2] **Paolo Traverso,**[3] **Dana Nau**[1]
[1] University of Maryland, College Park, MD 20742, USA
[2] LAAS-CNRS, 31077, Toulouse, France
[3] Fondazione Bruno Kessler, I-38123, Povo-Trento, Italy
patras@umd.edu, jmason12@terpmail.umd.edu, malik@laas.fr, traverso@fbk.eu, nau@cs.umd.edu

Abstract

We demonstrate a system with integrated acting, planning and learning algorithms that uses hierarchical operational models to perform tasks in dynamically changing environments. In AI research, synthesizing a plan of action has typically used *descriptive models* of the actions that abstractly specify *what* might happen as a result of an action, and are tailored for efficiently computing state transitions. However, executing the planned actions has needed *operational* models, in which rich computational control structures and closed-loop online decision-making are used to specify *how* to perform an action in a nondeterministic execution context, react to events and adapt to an unfolding situation. *Deliberative actors*, which integrate acting and planning, have typically needed to use both of these models together—which causes problems when attempting to develop the different models, verify their consistency, and smoothly interleave acting and planning. As an alternative, we demonstrate an acting and planning engine in which both planning and acting use the same operational models. These rely on hierarchical task-oriented *refinement methods* offering rich control structures. In addition, we also have learning strategies that guide the actor and the planner.

Demo video: https://youtu.be/T1ZkBmHrERk
Source code: https://bitbucket.org/sunandita/rae/

Introduction

Consider autonomous AI actors performing a diverse set of tasks in dynamically changing environments. Such actors (Patra et al. 2021a, 2020, 2019, 2018; Ghallab, Nau, and Traverso 2016) need to be reactive and they need to act in a purposeful deliberative way. The tight integration of acting and planning is important in domains with dynamic events, concurrent tasks, sensing actions with an unfolding execution context. Such requirements are fulfilled by our system by combining a reactive approach and a plan-based approach using hierarchical operational models. The acting components (RAE/APE), are inspired by the well-known PRS system. At each decision step, they can get advice from a planner (APEPlan/RAEPlan/UPOM) for a near-optimal choice with respect to a utility function. In contrast to APE (Patra et al. 2018), RAE (Patra et al. 2021a) can also take the advice from a learning strategy when there is not enough time to plan. APEPlan can only be used with the actor, APE. RAEPlan and UPOM are used with RAE. UPOM is an anytime planner that uses a UCT-like Monte Carlo Tree Search procedure whose rollouts are simulations of the actor's operational models. RAE and UPOM can take the advice from learning strategies, LearnM, LearnH and LearnMI that acquire, from online acting experiences and/or simulated planning results, a mapping from decision contexts to method instances as well as a heuristic function to guide the planner.

Architecture and Representation

Our approach integrates reactive and deliberative capabilities on the basis of hierarchical operational models. The main ingredients of the operational models are tasks (τ), actions and refinement methods. Refinement methods (m) provide alternative procedures for accomplishing a task. The state s is represented using a Python object with variable bindings. (Patra et al. 2021a) has the details of our formalism. Our system focuses on a reactive perspective, integrated with planning and learning capabilities. The popular three-layer architectures for autonomous deliberative actors usually combine a platform layer with sensory-motor modules, a reactive control layer, and deliberative planning layer. Our approach merges the last two layers within a reactive-centered perspective.

The central component of the architecture (labelled "RAE" in Figure 1) interacts with the environment for sensing and actuation through an execution platform, from which it receives events and world state updates. It also interacts with users getting tasks to be performed and reporting on their achievement. The actor reacts to tasks and events through hierarchical refinements specified by a library of operational models. At each decision step, the actor may use the planner to make the appropriate choice. The planner performs a look-ahead by simulating available options in the current context. Supervised learning is used to speed-up the planner with a heuristic, avoiding very deep and costly Monte Carlo rollouts; it also provides a base policy for an anytime strategy when the actor has no time for planning.

Domain Deployment

The domains we focus on have dynamic events, dead ends, actions that sense some kind information from the environment, actor collaboration and concurrent tasks. Our system

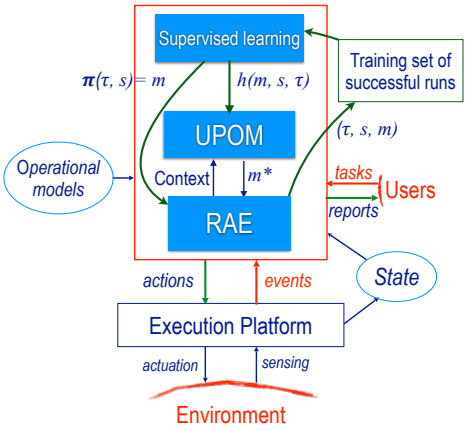

Figure 1: Architecture of our refinement acting, planning and learning system. Knowledge represented via hierarchical operational models.

comes with a few test domains whose features are summarized in Table 1 and Figure 2.

Domain	Dynamic events	Dead ends	Sensing	Robot collaboration	Concurrent tasks
S&R	✓	✓	✓	✓	✓
Explore	✓	✓	✓	✓	✓
Fetch	✓	✓	✓	–	✓
Nav	✓	–	✓	✓	✓
Deliver	✓	✓	–	✓	✓

Table 1: Features of the available test domains.

To deploy the proposed approach in a new domain, it is necessary to have an execution platform or the equivalent collection of sensing and primitive actions, as well as a set of methods. The human experts define the tasks and events, and a set of refinement methods in a corresponding domain file. The deployment of RAE and UPOM in a prototype application for security monitoring and recovery from attacks on Software-Defined Networks is described in (Patra et al. 2021b).

Environment Simulation The chosen acting engine executes methods and triggers the execution of primitive actions by the platform. The planner has to simulate both. The reliability of such a simulator affects the quality of plans. In some cases, available simulation tools can be useful, e.g., physics-based simulations, robotic simulations, automated manufacturing simulations and so forth. A fallback option would be to sample the possible outcomes of every action from probability distributions, which are initialized by a human expert.

Optimization Criteria and Evaluation Metrics The appropriate utility function can be application dependent. One may consider a function combining rewards for desirable or undesirable states, and costs for the time and resources of actions. The efficiency utility function maximizes values to easily account for failures. Using the success ratio utility, the actor seeks a method that has a good chance to succeed. The user can define their own utility function.

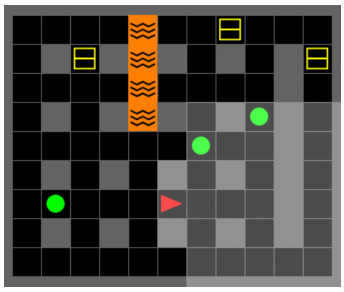

Figure 2: In all the domains of Table 1, the actor needs to accomplish concurrent tasks, e.g., collecting different kinds of balls and boxes. There are dynamically changing obstacles in its path and emergency events. There are dead ends, e.g, falling in lava, battery discharge, etc that the actor cannot recover from. The actor only has partial view of the environment (gray shaded region).

Conclusions

We have demonstrated a novel system for integrating acting, planning and learning using hierarchical operational models for domains with challenging features such as dynamicity, dead-ends, exogenous events, sensing and information gathering actions, collaborative and concurrent tasks. The actors may choose to get advice from an online planner to choose efficient methods for performing a task. The planners provide near-optimal method instances with respect to utility functions that may be quite general. Our system also has three learning strategies: LearnM and LearnMI to learn a mapping from a task in a given context to a good method, and LearnH, to learn a domain specific heuristic function for our hierarchical refinement framework. Rather than just evaluating the system's planning functionality, we have devised simulations and measurements that assess its overall acting performance, with and without planning and learning, taking into account exogenous events and failure cases.

References

Ghallab, M.; Nau, D. S.; and Traverso, P. 2016. *Automated Planning and Acting*. Cambridge University Press.

Patra, S.; Ghallab, M.; Nau, D.; and Traverso, P. 2018. APE: An Acting and Planning Engine. *Advances in Cognitive Systems* .

Patra, S.; Ghallab, M.; Nau, D.; and Traverso, P. 2019. Acting and planning using operational models. In *AAAI*, volume 33, 7691–7698.

Patra, S.; Mason, J.; Ghallab, M.; Nau, D.; and Traverso, P. 2021a. Deliberative Acting, Planning and Learning with Hierarchical Operational Models. *Artificial Intelligence* 103523.

Patra, S.; Mason, J.; Kumar, A.; Ghallab, M.; Traverso, P.; and Nau, D. 2020. Integrating Acting, Planning, and Learning in Hierarchical Operational Models. In *ICAPS*.

Patra, S.; Velasquez, A.; Kang, M.; and Nau, D. 2021b. Using Online Planning and Acting to Recover from Cyberattacks on Software-defined Networks. In *IAAI*.