4.2. Integration of Reinforcement Learning in Planning: PDDLGym

Ángel Aso Mollar
Eva Onaindía de la Rivaherrera

Planificación Inteligente

Curso 2023-2024

Máster Universitario en Inteligencia Artificial, Reconocimiento de Formas e Imagen Digital

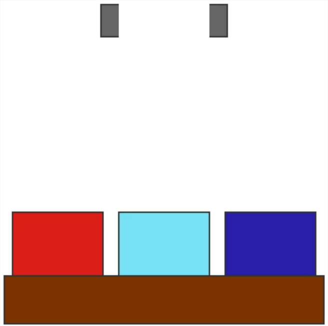
DEPARTAMENTO DE SISTEMAS INFORMÁTICOS Y COMPUTACIÓN
UNIVERSITAT POLITÈCNICA DE VALÈNCIA

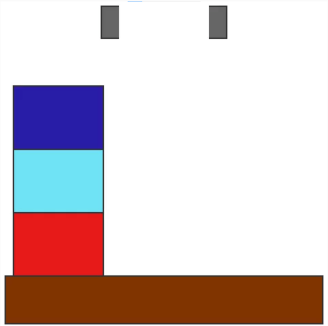
Contents

Planning + RL integration

Solving concrete Planning problems with RL

◇ Reinforcement Learning can be used to discover solutions of concrete problems inside a Planning domain.



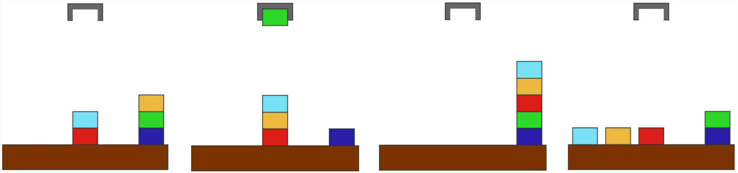


Solving concrete Planning problems with RL

- States (and actions) of the MDP will be Planning states (and actions).
- The initial state will also be the initial Planning state.
- The learner will iterate over and over starting with a random distribution, and will refine its policy receiving a reward of 1 if it reaches the goal, and 0 in any other case.
- The discounted return-value will make the agent to potentially achieve the optimal path.
- Let's see an example using the well-known *Blocksworld* domain.

GENERALIZED PLANNING WITH RL

◇ Reinforcement Learning can also be used to discover a strategy for a wide variety of problems within a domain.



◇ In Generalized Planning, we want to achieve solutions that potentially solve any problem of a domain. Hence, we lose optimality.

◇ For example, a well known general strategy for the *Blocksworld* domain is to place all blocks on the table and stack them in the right order.

Problem: Sparse-rewards

- Sparse-rewarded problems in RL are those that rarely receive signals from the environment when applying actions (at each time-step).
- This type of problems need a large number of episodes to reach a solution, as the amount of information that the environment provides is minimal and relies almost purely on luck.
- To mitigate this problem is an open field of study in neuro-symbolic learning.

Problem: The curse of dimensionality

- For pure RL algorithms we need to store in certain structures the values that the agent is retrieving from the environment.
- For example, in Q-learning we need to keep track of every state and action that the agent visited.
- This leads to huge structures that cannot escalate, neither in computational time nor in memory.
- Planning is a very clear example of this: the state space is so large that a simple Q-learning training can take several hours.
- This is why we need to think of solutions that reduce (or eliminate) the need for these large structures.

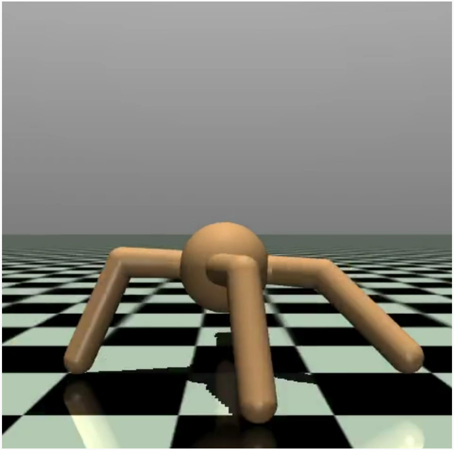
A framework for RL and Planning

OpenAI Gymnasium

- Maintained fork of **OpenAI's Gym** library[1].
- API for single-agent Reinforcement Learning environments. Includes implementations of common environments
- Contains four key functions: **make**, **reset**, **step** and **render**.
- RL MDPs are represented as **environments** (Env class).
- Every environment specifies the action and state space.

[1] `https://gymnasium.farama.org/index.html`

Example: ANT



https://gymnasium.farama.org/

PDDLGym

- **PDDLGym**[2] is a framework that automatically constructs OpenAI Gym environments from PDDL domains and problems.
- Observations and actions in PDDLGym are relational (extracted from PDDL files).
- It provides an API to train an RL agent to solve planning problems.
- This framework is the one we are going to use for the practical exercises.

[2] https://github.com/tomsilver/pddlgym