Reinforcement Learning

Framework

Mario Martin
Universitat politècnica de Catalunya
Dept. LSI

Motivation

- Target is to learn a behavior instead of learning a concept
- Agent discovers the behavior
  - No examples required for learning
  - No need of an expert to know beforehand the behavior to be learnt
- Leaning is grounded in the environment of the agent
- No bias in learning.
- Agent learns the *optimal* behavior.

Motivation

Reinforcement Learning captures essential features of agent learning

Costs

- Some assumptions are required
- Long computational time and a lot of memory required

- Fortunately these inconveniences can be reduced in some cases.

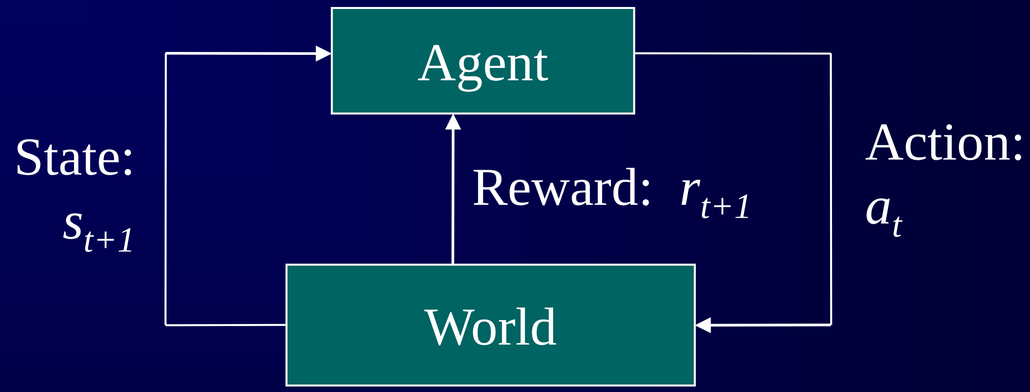
Definition of Reinforcement Learning

1. Learning about, from, and while interacting with an environment to achieve a goal

… read as …

2. Learning a mapping from situations to actions to maximize long-term reward, without using a model of the world

The Agent-Environment Interaction



Agent and environment interact at discrete time steps: $t = 0, 1, 2, \ldots$

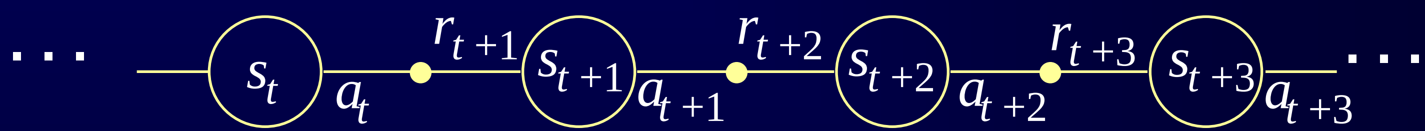
Agent observes state at step t: $s_t \in S$

produces action at step t: $a_t \in A(s_t)$

gets resulting reward: $r_{t+1} \in \Re$

and resulting next state: s_{t+1}

Trace of a trial



Markovian Decision Process (MDP)

- Agent-env. interaction becomes a MDP when
  - Finite set of situations
  - Finite set of actions
  - Transition probabilities:

$$P_{ss'}^{a} = \Pr\left\{ s_{t+1} = s' \,\middle|\, s_t = s, a_t = a \right\} \quad \text{for all } s, s' \in S,\, a \in A(s).$$

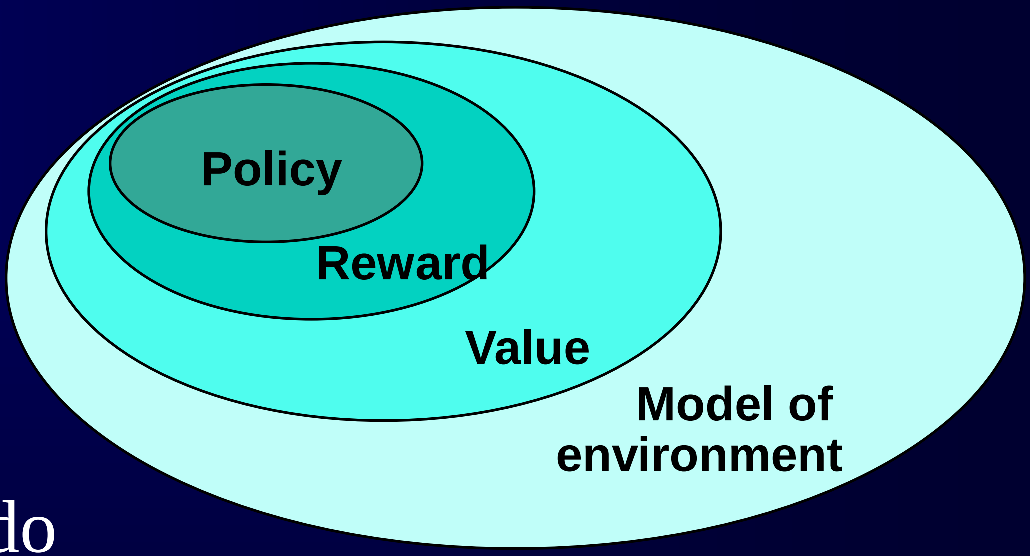
  - Reward probabilities:

$$R_{ss'}^{a} = E\left\{ r_{t+1} \,\middle|\, s_t = s, a_t = a, s_{t+1} = s' \right\} \quad \text{for all } s, s' \in S,\, a \in A(s).$$

The Markovian Property

- When actions and states are not finite, discretize the set of actions and states
- When transition probabilities do not depend only on the current state,
  - The agent could represent states as structures built up over time from sequences of sensations (for instance, for the agent the state is a window of the recent states)
  - Use POMDP learning algorithms

Elements of RL



- **Policy**: what to do
- **Reward**: what is good
- **Value**: what is good because it *predicts* reward
- **Model**: what follows what

Components of an RL Agent (I)

- Policy (behavior)
  - Mapping from states to actions

$$\pi(s) \longrightarrow a$$

- Reward
  - Local reward in state t

$$r_t$$

Components of an RL Agent (II)

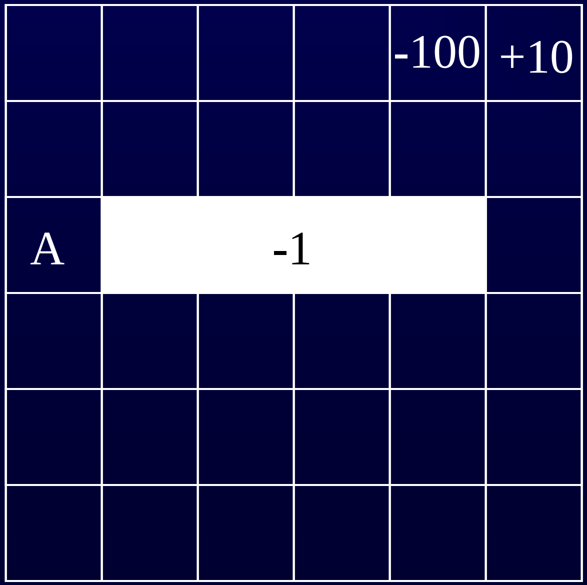
- Model
  - $T(s,a,s')$: probability of transition from state s to s' executing action a
- The transition probabilities depends only on these parameters (Markovian model)
- Not known by the agent

Components of an RL Agent (III)

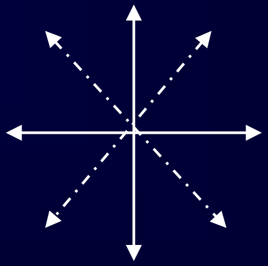
- Value functions
  - $V^{\pi}(s)$: Long-term reward estimation from state *s* following policy π
  - $Q^{\pi}(s,a)$: Long-term reward estimation from state *s* executing action *a* and then following policy π

Simple Example

- Markovian model, reward function, set of actions, set of states
- Maze:



Actions, 90% reliable



Getting the Degree of Abstraction Right

- Actions can be low level (e.g., voltages to motors), or high level (e.g., accept a job offer), “mental” (e.g., shift in focus of attention), etc.
- States can low-level “sensations”, or they can be abstract, symbolic, based on memory, or subjective (e.g., the state of being “surprised” or “lost”).
- An RL agent is not like a whole animal or robot, which consist of many RL agents as well as other components.
- Reward computation is in the agent’s environment because the agent cannot change it arbitrarily.