# Reinforcement Learning Lab

## Lesson Extra: Policy Iteration and Value Iteration

Gabriele Roncolato and Alberto Castellini

University of Verona
*email: gabriele.roncolato@univr.it*

Academic Year 2024-25

# Environment Setup

The first step for the setup of the laboratory environment is to update the repository and load the miniconda environment.

## Safe Procedure

Always back up the previous lessons' solutions before executing the repository update.

- Update the repository of the lab:

```
cd RL-Lab
git stash
git pull
git stash pop
```

- Activate the *miniconda* environment:

```
conda activate rl-lab
```

# Today Assignment

In today's lesson, we implement the value iteration and policy iteration algorithms in Python. In particular, the file to complete is:

```
RL-Lab/lessons/lesson_extra_code.py
```

Inside the file, two functions are only partially implemented. The objective of this lesson is to complete them.

- **def value_iteration()**
- **def policy_iteration()**

Expected results can be found in:

```
RL-Lab/results/lesson_extra_results.txt
```

# Pseudocode - Policy Iteration (a)

**function** POLICY-ITERATION($mdp$) **returns** a policy
  **inputs**: $mdp$, an MDP with states $S$, actions $A(s)$, transition model $P(s' \mid s, a)$
  **local variables**: $U$, a vector of utilities for states in $S$, initially zero
    $\pi$, a policy vector indexed by state, initially random

  **repeat**
    $U \leftarrow$ POLICY-EVALUATION($\pi, U, mdp$)
    $unchanged? \leftarrow$ true
    **for each** state $s$ **in** $S$ **do**
      **if** $\max_{a \in A(s)} \sum_{s'} P(s' \mid s, a)\ U[s'] > \sum_{s'} P(s' \mid s, \pi[s])\ U[s']$ **then do**
        $\pi[s] \leftarrow \operatorname{argmax}_{a \in A(s)} \sum_{s'} P(s' \mid s, a)\ U[s']$
        $unchanged? \leftarrow$ false
  **until** $unchanged?$
  **return** $\pi$

Figure: Pseudocode for the policy iteration algorithm, the implementation is from the Russell and Norvig book: *Artificial Intelligence: A Modern Approach*

# Pseudocode - Policy Iteration (b)

The *policy evaluation* of the policy iteration algorithm implements the following function:

$$U_i(s) = R(s) + \gamma \sum_{s'} P(s' \mid s, \pi_i(s)) U_i(s') \,.$$

Figure: Policy Evaluation function.

**Hint:**

In the assignments, the update functions require discounting the future reward (e.g., $r + \gamma \cdot \textit{future}$). Remember that for the terminal states, there is no future! Update only with $r$ in such cases.

# Pseudocode - Value Iteration

**function** VALUE-ITERATION($mdp, \epsilon$) **returns** a utility function
  **inputs**: $mdp$, an MDP with states $S$, actions $A(s)$, transition model $P(s' \mid s, a)$,
    rewards $R(s)$, discount $\gamma$
    $\epsilon$, the maximum error allowed in the utility of any state
  **local variables**: $U$, $U'$, vectors of utilities for states in $S$, initially zero
    $\delta$, the maximum change in the utility of any state in an iteration

  **repeat**
    $U \leftarrow U'$; $\delta \leftarrow 0$
    **for each** state $s$ **in** $S$ **do**
      $$U'[s] \leftarrow R(s) + \gamma \max_{a \in A(s)} \sum_{s'} P(s' \mid s, a)\, U[s']$$
      **if** $|U'[s] - U[s]| > \delta$ **then** $\delta \leftarrow |U'[s] - U[s]|$
  **until** $\delta < \epsilon(1-\gamma)/\gamma$
  **return** $U$

Figure: Pseudocode for the value iteration algorithm, the implementation is from the Russell and Norvig book *Artificial Intelligence: A Modern Approach*