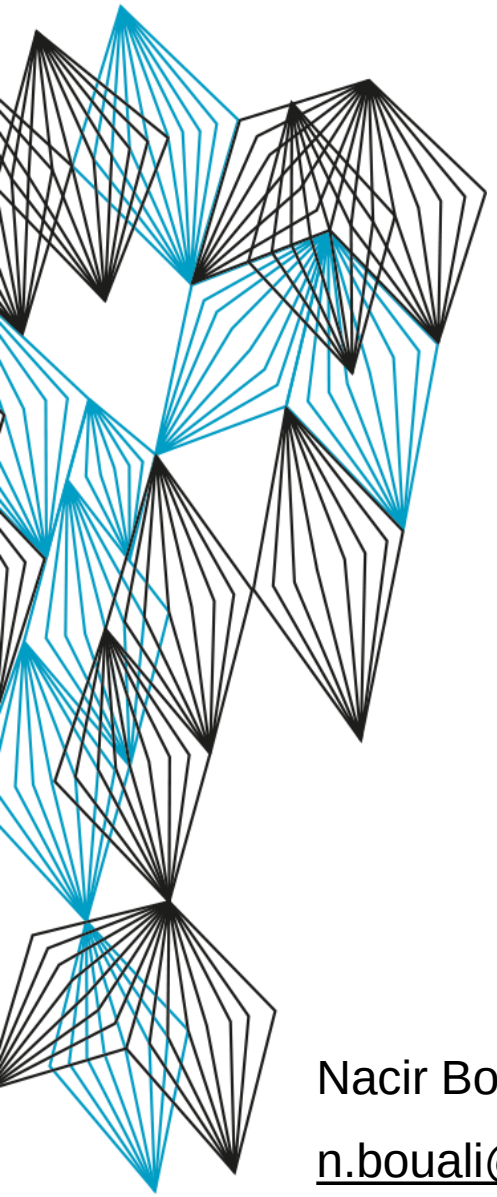


ARTIFICIAL INTELLIGENCE & CYBER SECURITY

REINFORCEMENT LEARNING
MARKOV DECISION PROCESS

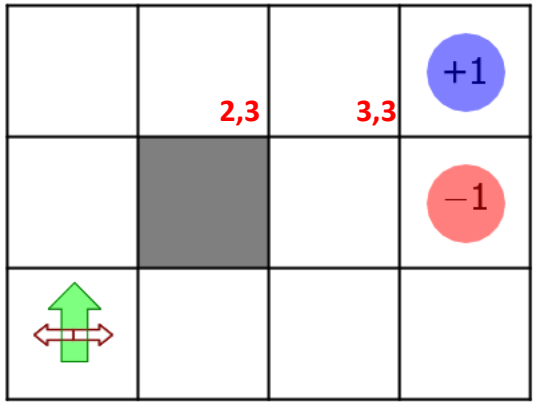
Nacir Bouali

n.bouali@utwente.nl

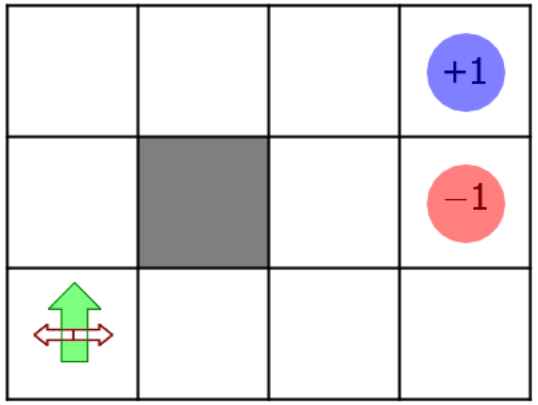
UNIVERSITY OF TWENTE.

MARKOV DECISION PROCESS

- Defined by;
  - Set of states, *S*
  - Set of actions A, and for each state s ∈ S a set of possible actions a ∈ A(s)
  - Probabilistic transition function, p(s′ | s, a)
  - Reward function R(s) (can be positive or negative)
- Sequential Decision Problem
- Fully Observable stochastic environment
- Markovian transition model
- Additive reward
- In a given MDP, what is the best course of action?
- In other words: what is the optimal policy?

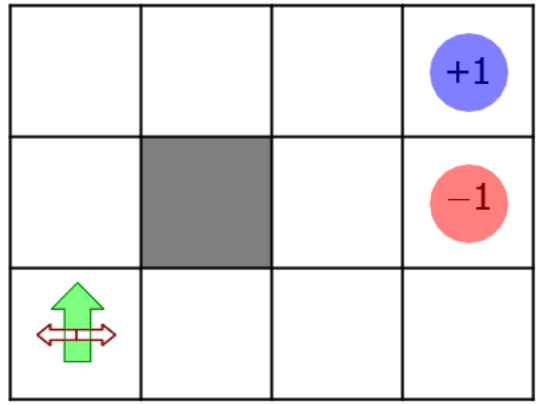


MARKOV DECISION PROCESS



- Example (Grid World)
- Sequential Decision Process
- Stochastic actions:
  - With probability 0.8, the intended action is executed
  - With probability 0.1, the agent moves perpendicular to the intended direction (on either side)
  - Collision with a wall: no movement
- Terminal states have rewards +1 and −1
- Non-terminal states have a reward of -0.04

MARKOV DECISION PROCESS

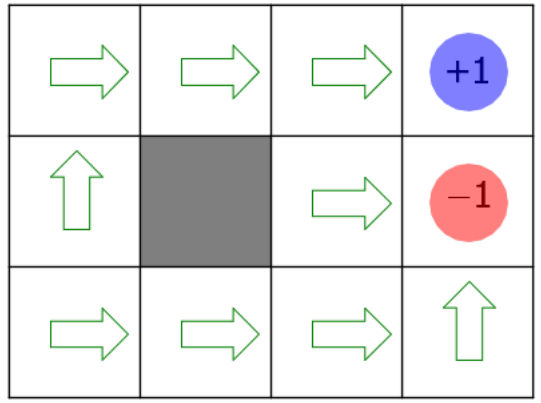


- How to deal with this problem space?
  - Not good: fixed set of actions
  - **Up, Up, Right, Right, Right**
  - **Right, Right, Up, Up, Right**
- Probability of reaching goal:
  - $.8^5 + .1^4 \times .8 = 0.32776$
- Policy: mapping from states to actions
  - $\pi : S \rightarrow A$
- How can we find a good policy?
  - Maximise Reward of state sequences
  - But a single policy results in many different sequences
  - Consider the expected utility

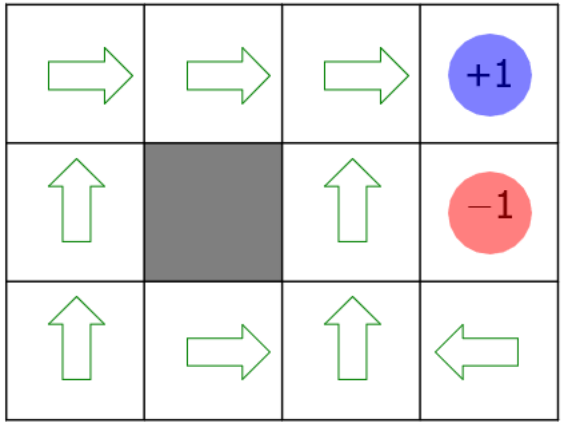
It works out fine five times, or wrong four times ("Up, Up, Right, Right") and right once ("Right").

MARKOV DECISION PROCESS

- Optimal policy $\pi*$:
  - yields the highest expected utility
  - Balances risk and reward
- Affected by reward of non-terminal states
  - Very low R: any termination is desirable
  - Low R: Speedy termination is desirable
  - Small negative R: avoid any risk
  - Positive R: avoid termination



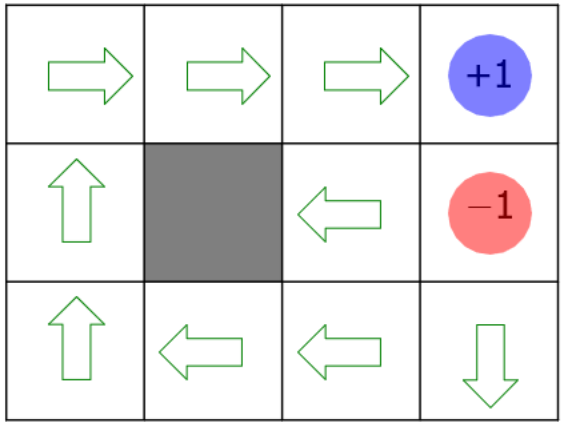
R=-3.0



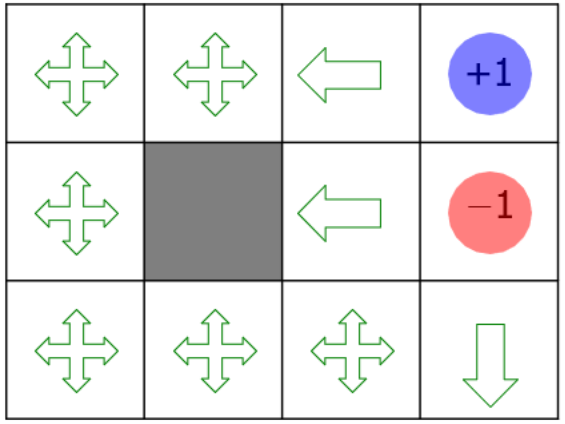
$R = -0.3$

MARKOV DECISION PROCESS

- Optimal policy $\pi*$:
  - yields the highest expected utility
  - Balances risk and reward
- Affected by reward of non-terminal states
  - Very low R: any termination is desirable
  - Low R: Speedy termination is desirable
  - Small negative R: avoid any risk
  - Positive R: avoid termination



$R = -0.01$



$R > 0$

MDP - UTILITIES

$\pi_1 \rightarrow +1 \quad +1 \quad +1 \quad +1 \ldots$

$\pi_2 \rightarrow +2 \quad +2 \quad +5 \quad +5 \ldots$

- Additive rewards: $U_h[s0, s1, s2, \ldots] = R(s0) + R(s1) + R(s2) + \cdots$
- Discounted rewards: $U_h[s0, s1, s2, \ldots] = R(s0) + \gamma R(s1) + \gamma^2 R(s2) + \cdots$
  - Infinite sequences
  - Absence of terminal states
  - Finite horizon

$$U^{\pi}(s) = \mathbb{E} \sum_{t=0}^{\infty} \gamma^t R(S_t)$$

- Utility of a state s
  - Stems from: utility of sequences starting in s, and their probability of occurring
  - Probability of a sequence depends on: policy, transition probability

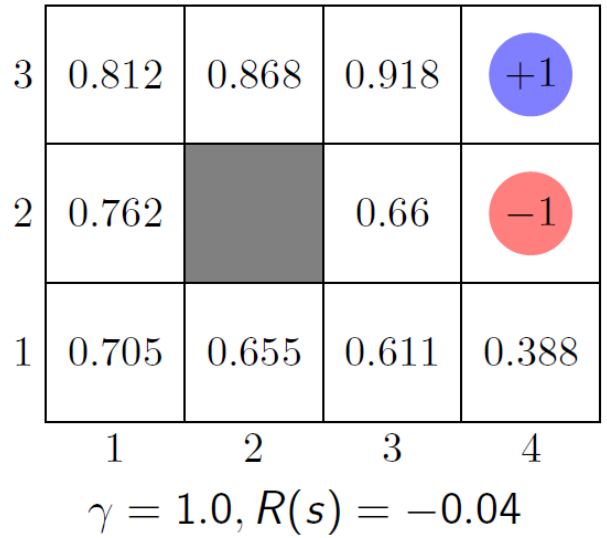
MDP - UTILITIES

- State utilities allow us to rank policies by expected rewards

$$\pi_s^* = \underset{\pi}{\operatorname{argmax}}\, U^\pi(s)$$

- Optimal policy independent of start state
  - Thus, the only really useful utility:

$$U(s) = U^{\pi^*}(s)$$



- Allows us to select actions by maximising expected utility:

$$\pi^*(s) = \underset{a \in A(s)}{\operatorname{argmax}} \sum_{s'} p(s'|s,a)\, U(s')$$

MDP – BELLMAN EQUATIONS

- The utility of a state is the immediate reward for that state plus the expected discounted utility of the next state assuming optimal actions:

$$U(s) = R(S) + \gamma \max_{a \in A(s)} \sum_{s'} p(s' \mid s, a)\, U(s')$$

  - If there are n states: n equations with n unknowns
  - Non-linear equations (max operator is non-linear): hard to solve
  - Solution: iterative algorithm