An Introductory Introduction to RL

Francisco S. Melo

Instituto Superior Técnico and INESC-ID

May 5, 2021

TÉCNICO LISBOA

Outline

What will this presentation be all about?

- What is RL?
- How to solve RL problems - the shallow view
- How to solve RL problems - the deep view
- Shallow waters…
- Fun stuff

Outline

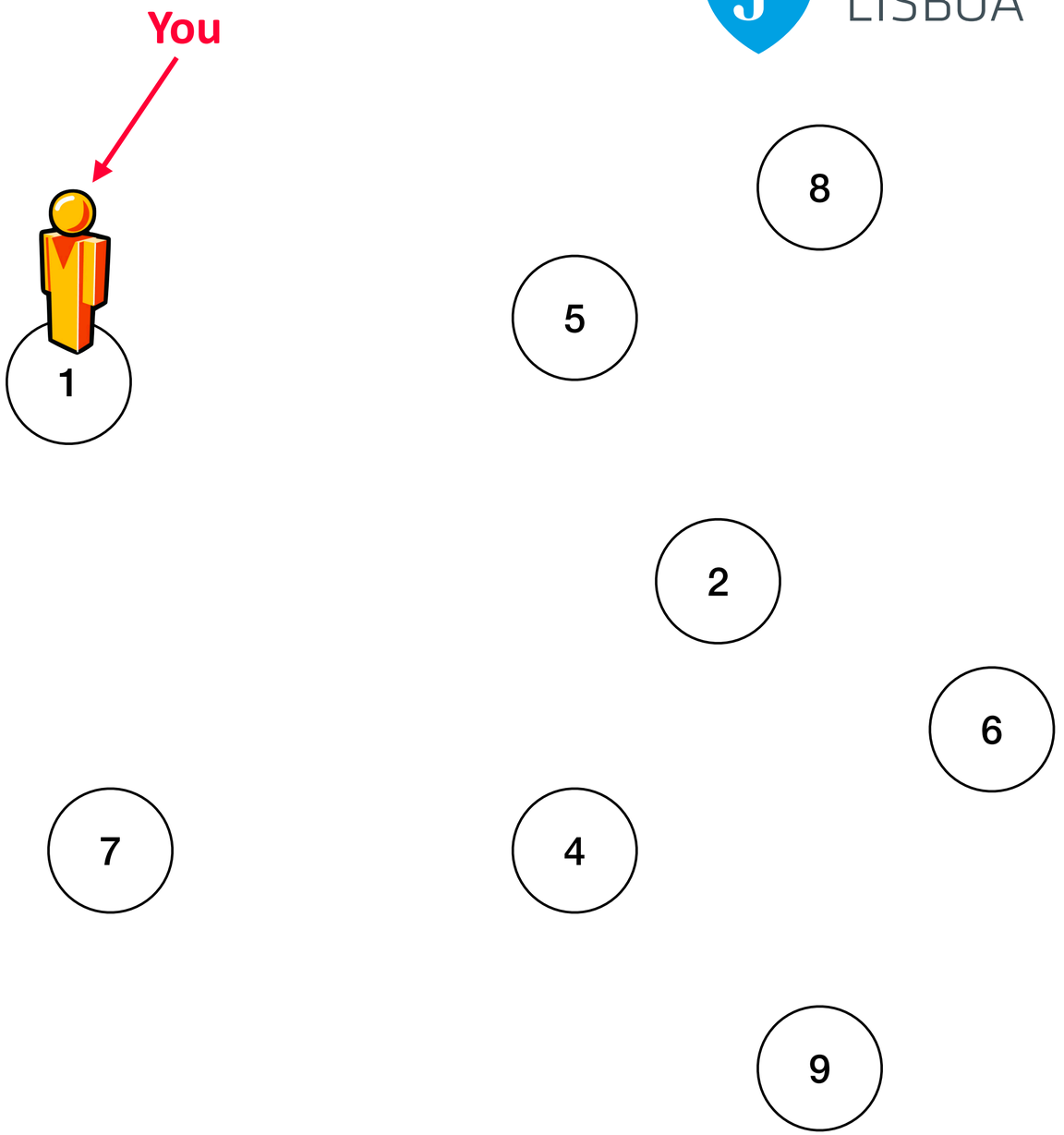
What will this presentation be all about?

- **What is RL?**
- How to solve RL problems - the shallow view
- How to solve RL problems - the deep view
- Shallow waters…
- Fun stuff

Let’s start with a puzzle…

- You are sitting in place n. 1
- You have something to do, but you do not know what
- We have 5 actions available:

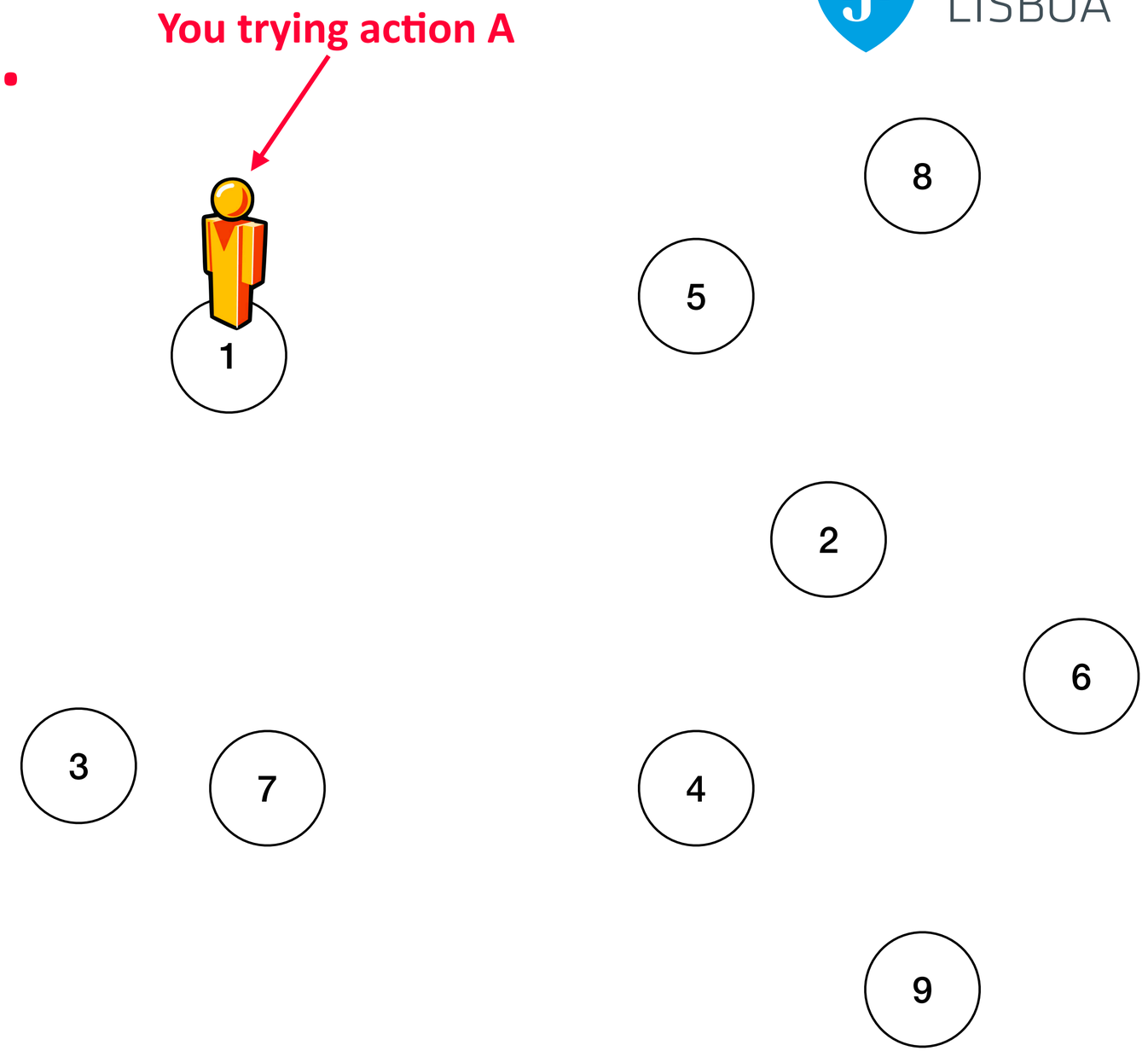
  A, B, C, D, E



Let’s start with a puzzle…

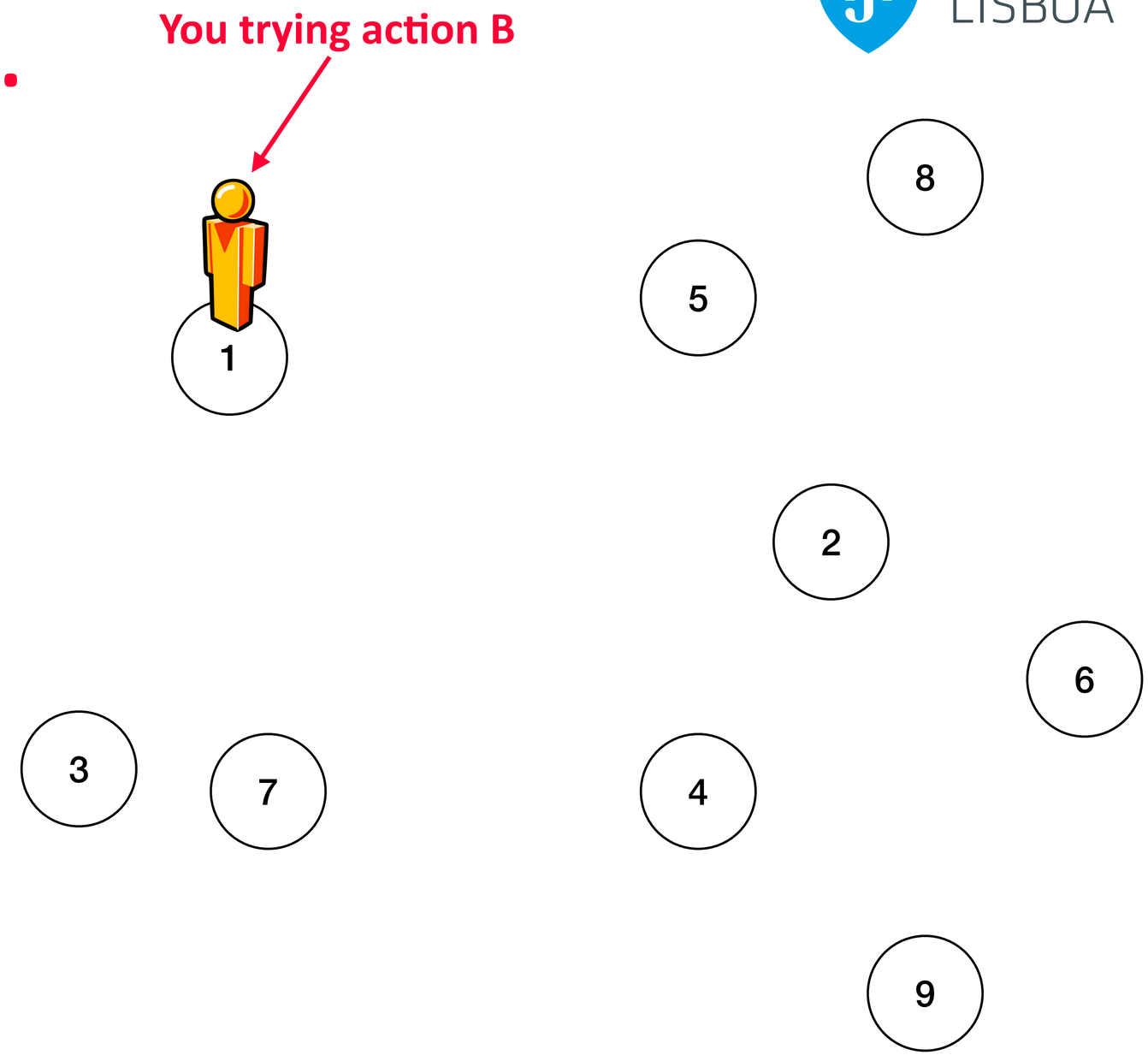
- You try action A…

  … nothing happens.



Let’s start with a puzzle…

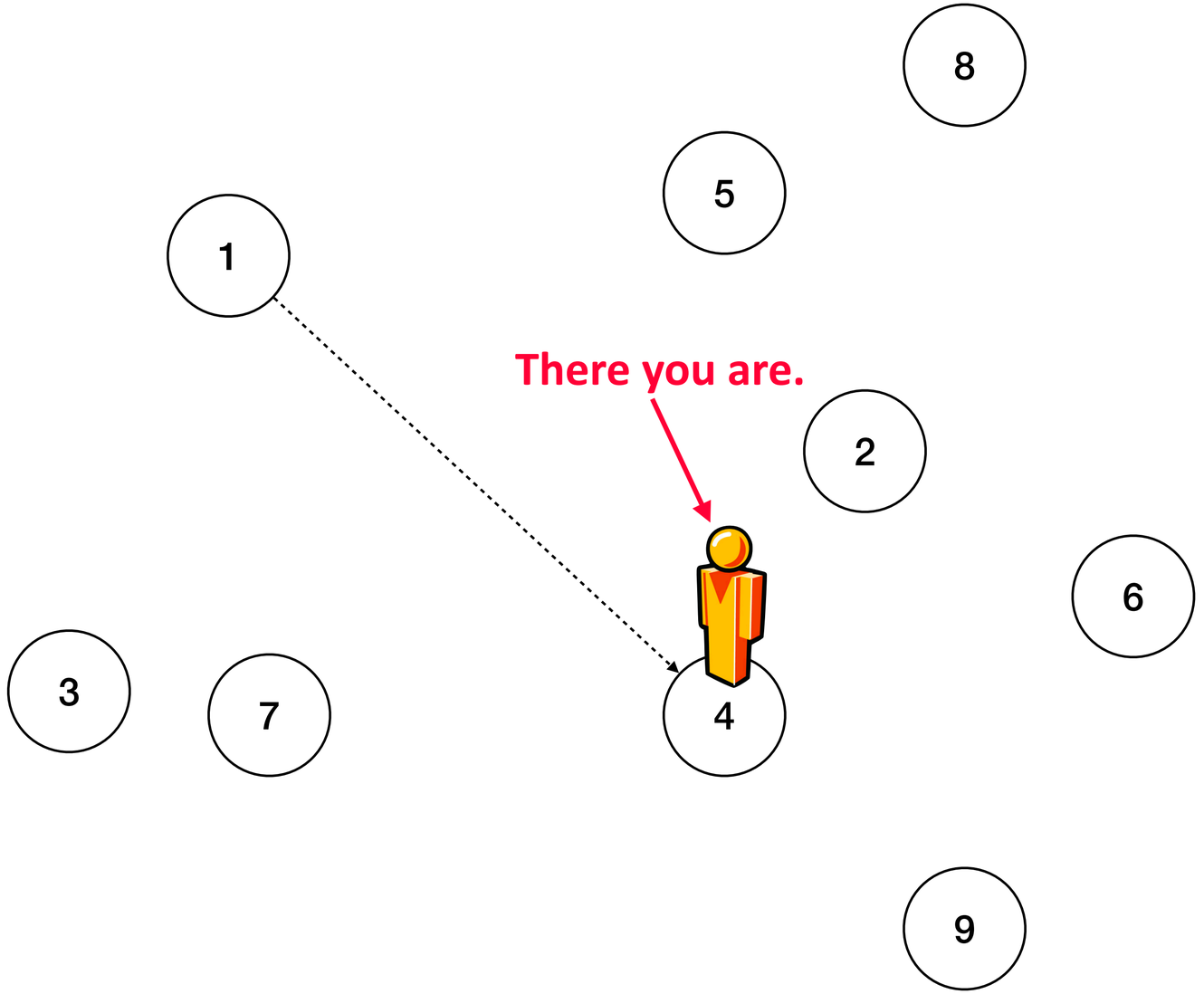
- So you try action B…



Let’s start with a puzzle…

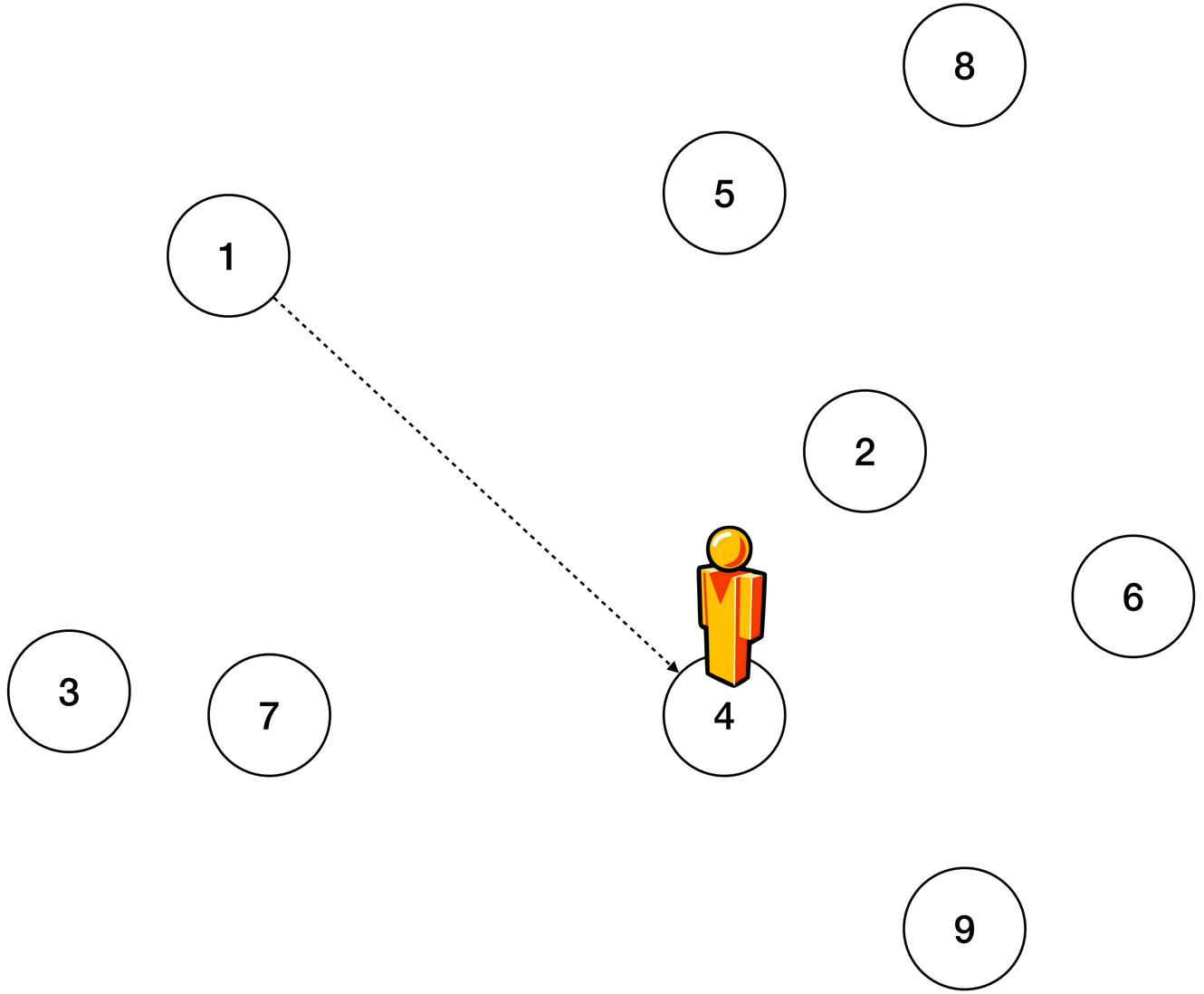
- So you try action B…

  … you end up in place n. 4



Let’s start with a puzzle…

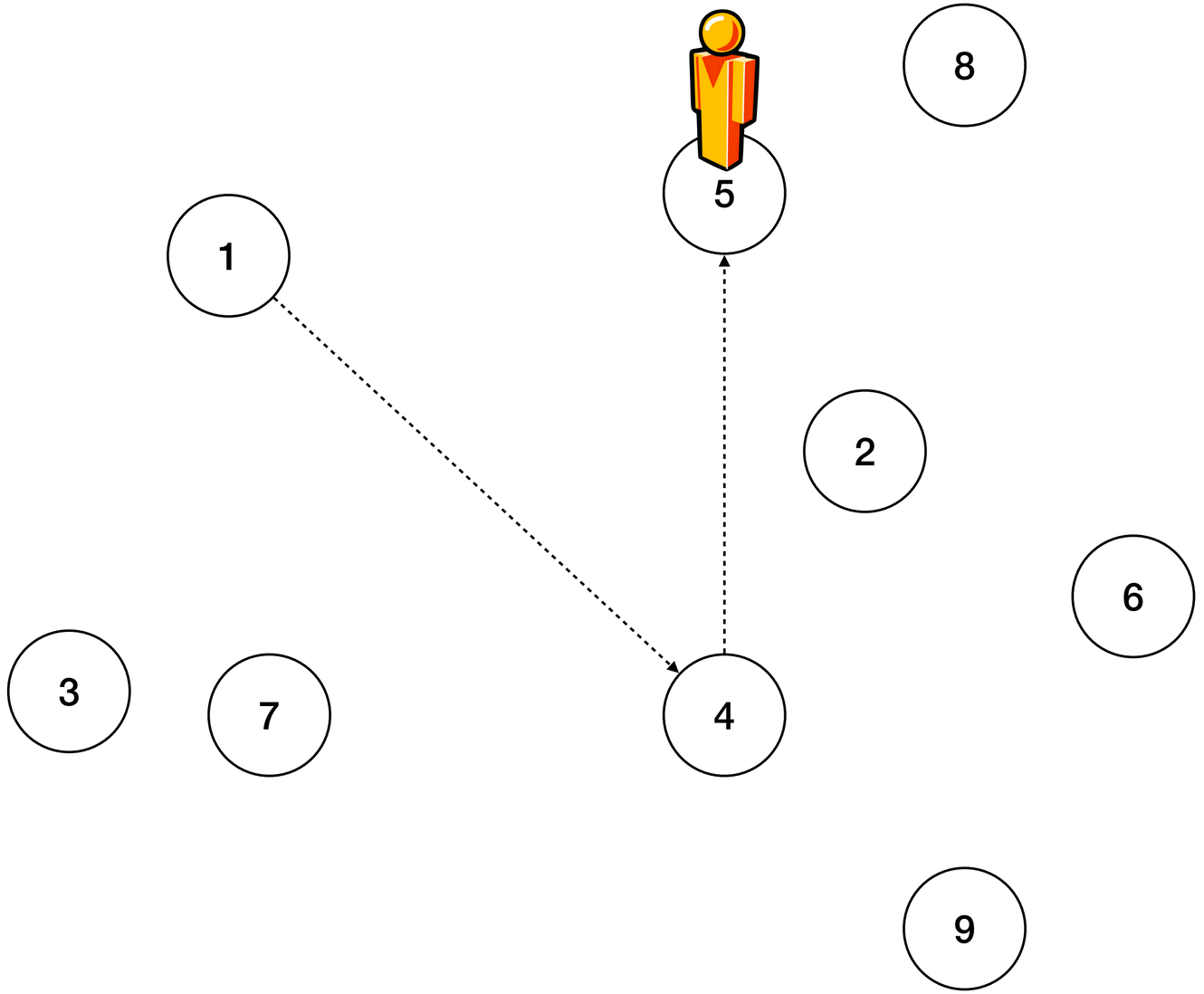
- Next you try action C…



Let’s start with a puzzle…

- Next you try action C…

  … you end up in place n. 5.



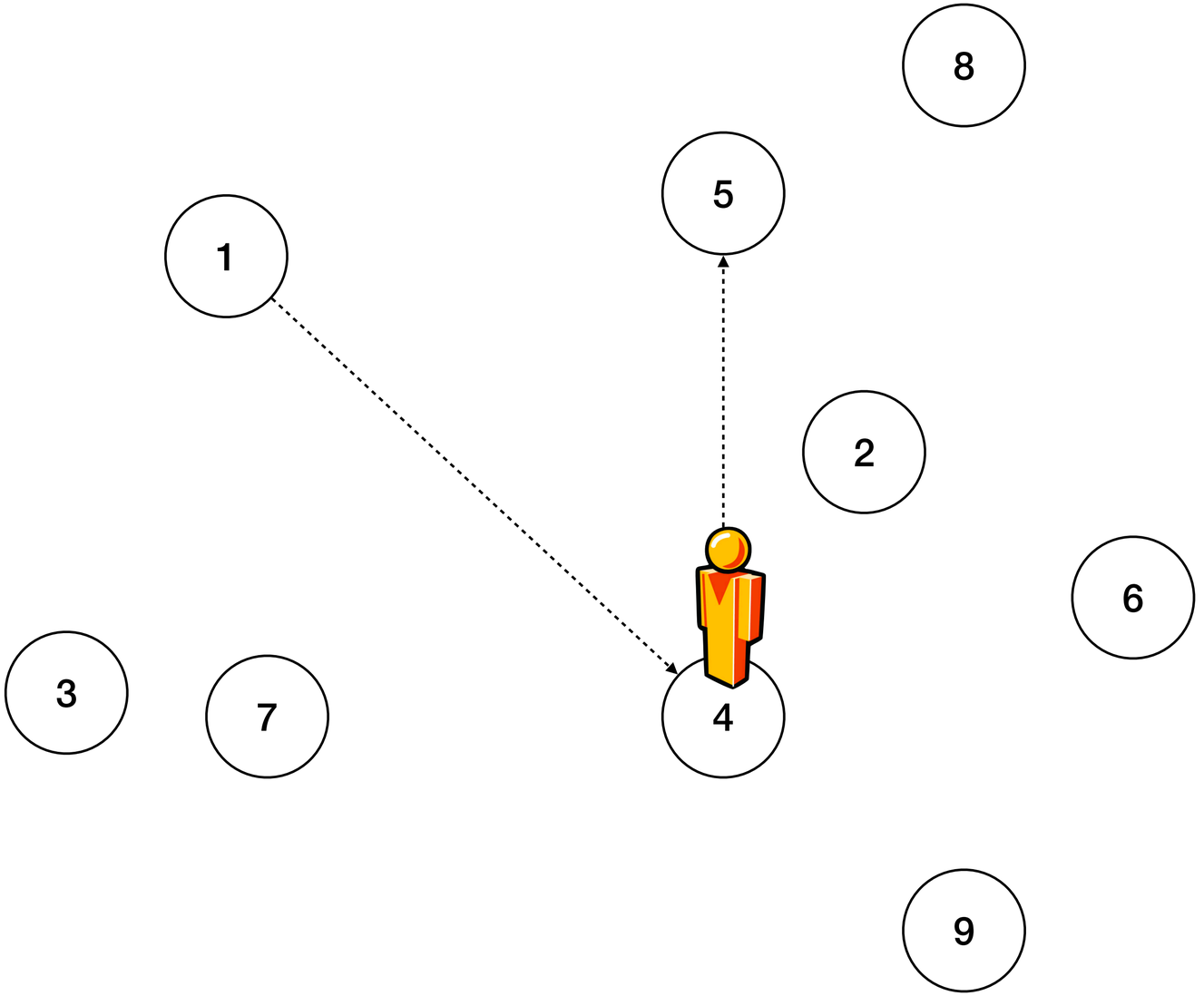
Let’s start with a puzzle…

- You try action D…

  … and end up in place n. 4.

- You try action E…

  … nothing happens.



Let’s start with a puzzle…

- You try action D…

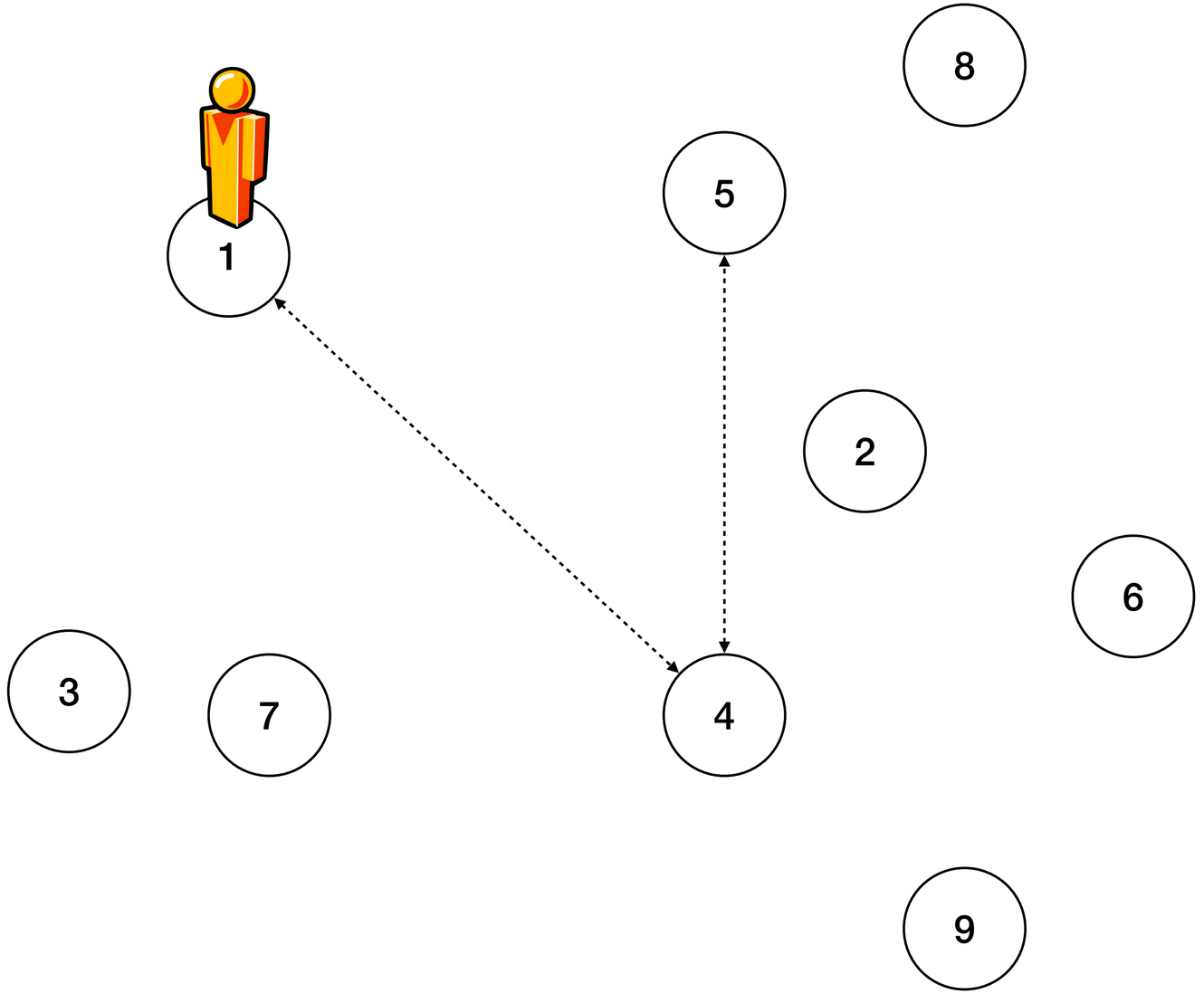
  … and end up in place n. 4.

- You try action E…

  … nothing happens.

- You try A again…

  … you go back to place n. 1.



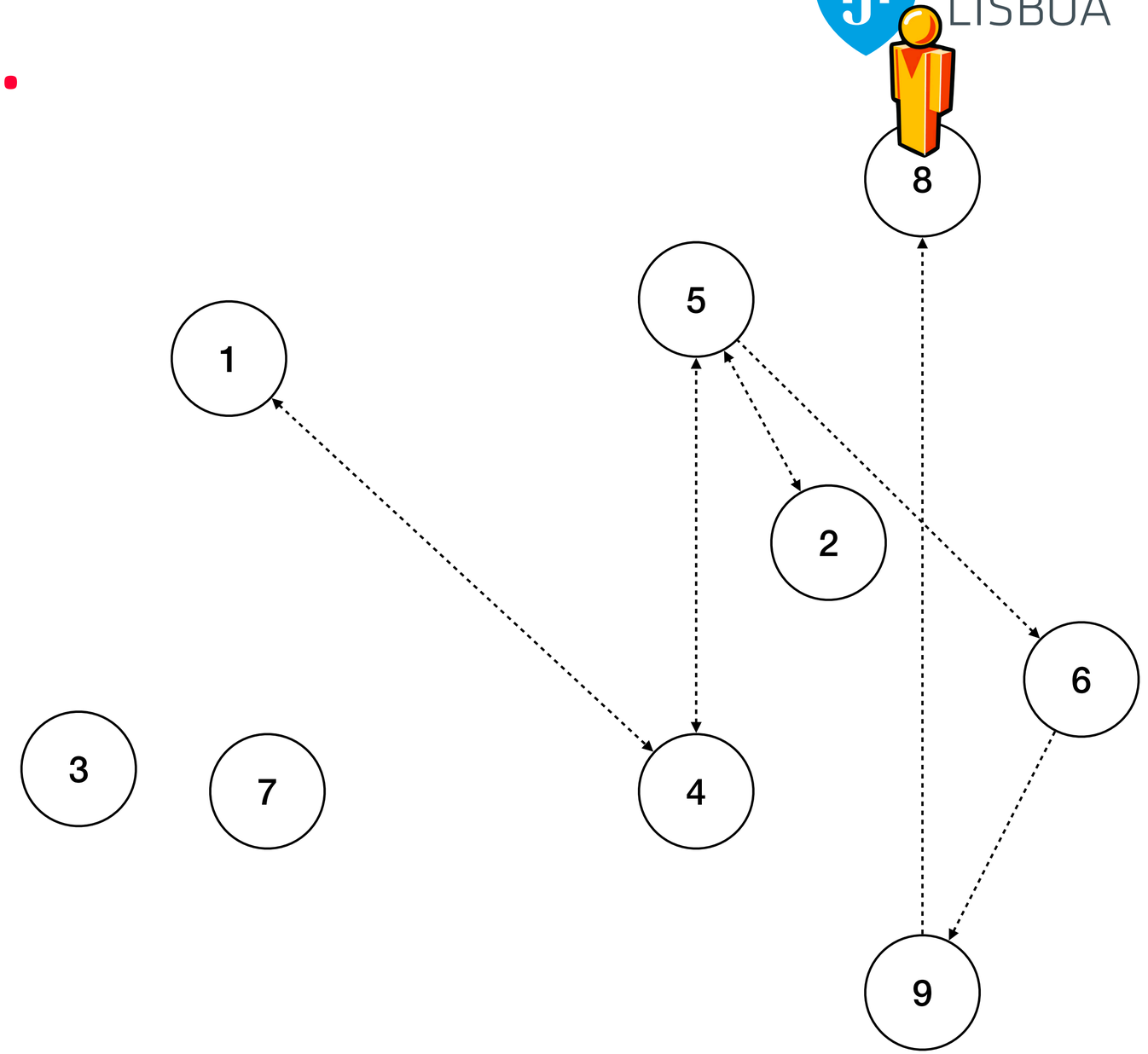
Let’s start with a puzzle…

- After a while…

  … you start to get a grasp of how

  things work.

- Then you try action D in place n. 8…



Let’s start with a puzzle…

- After a while…

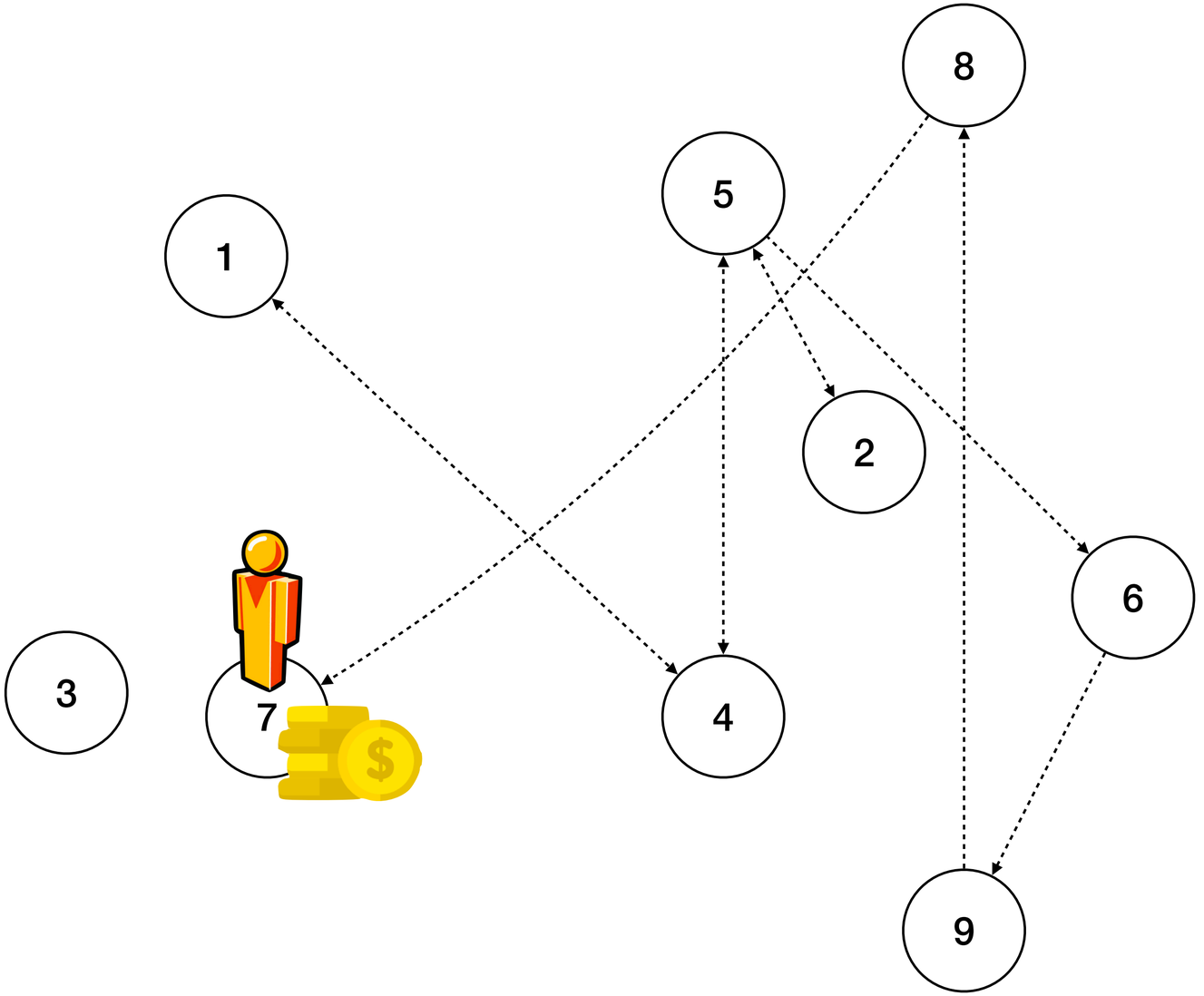
  … you start to get a grasp of how

  things work.

- Then you try action D in place n. 8…

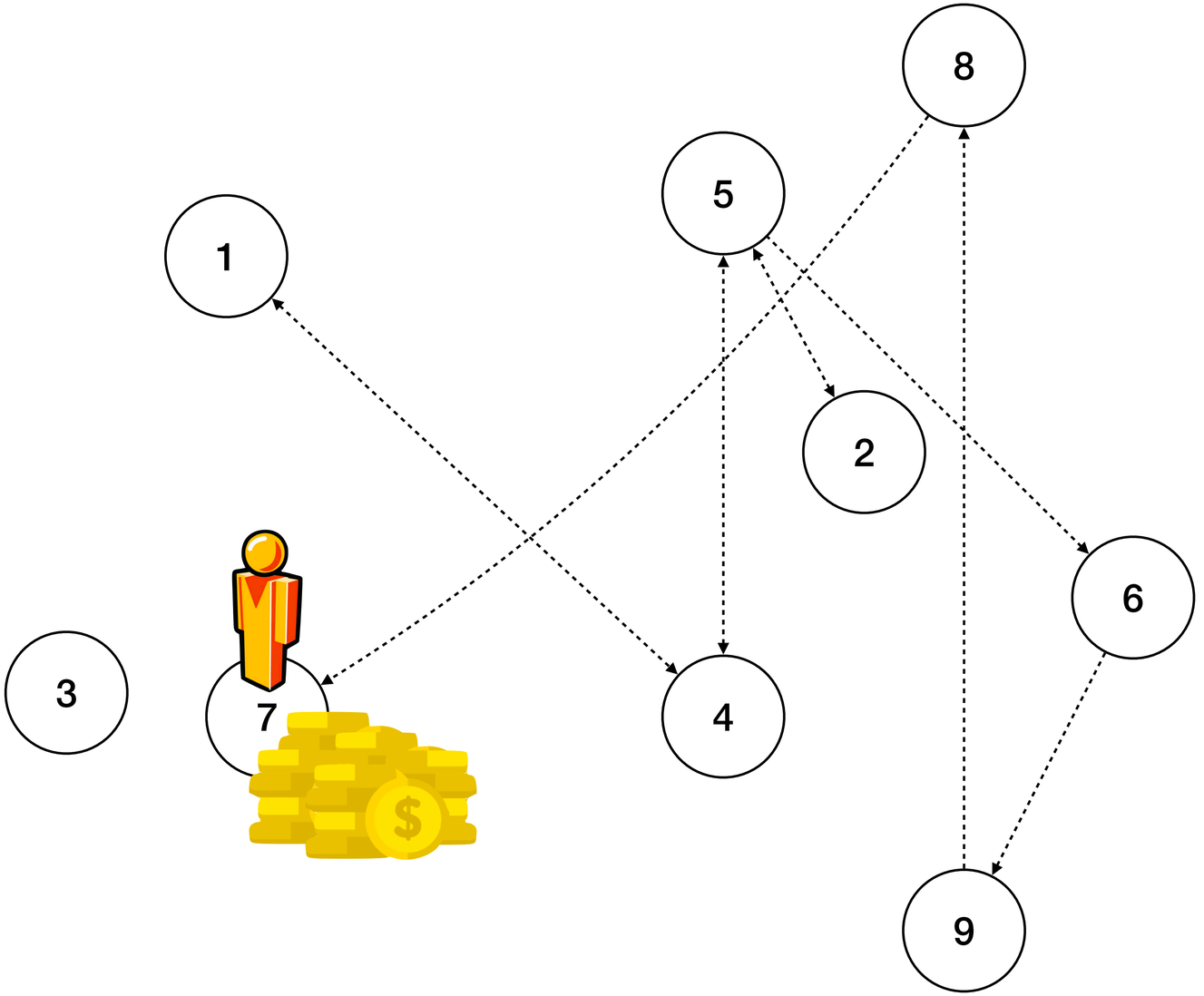
  … you end up in place n. 7

  … you get a prize of 5 EUR!



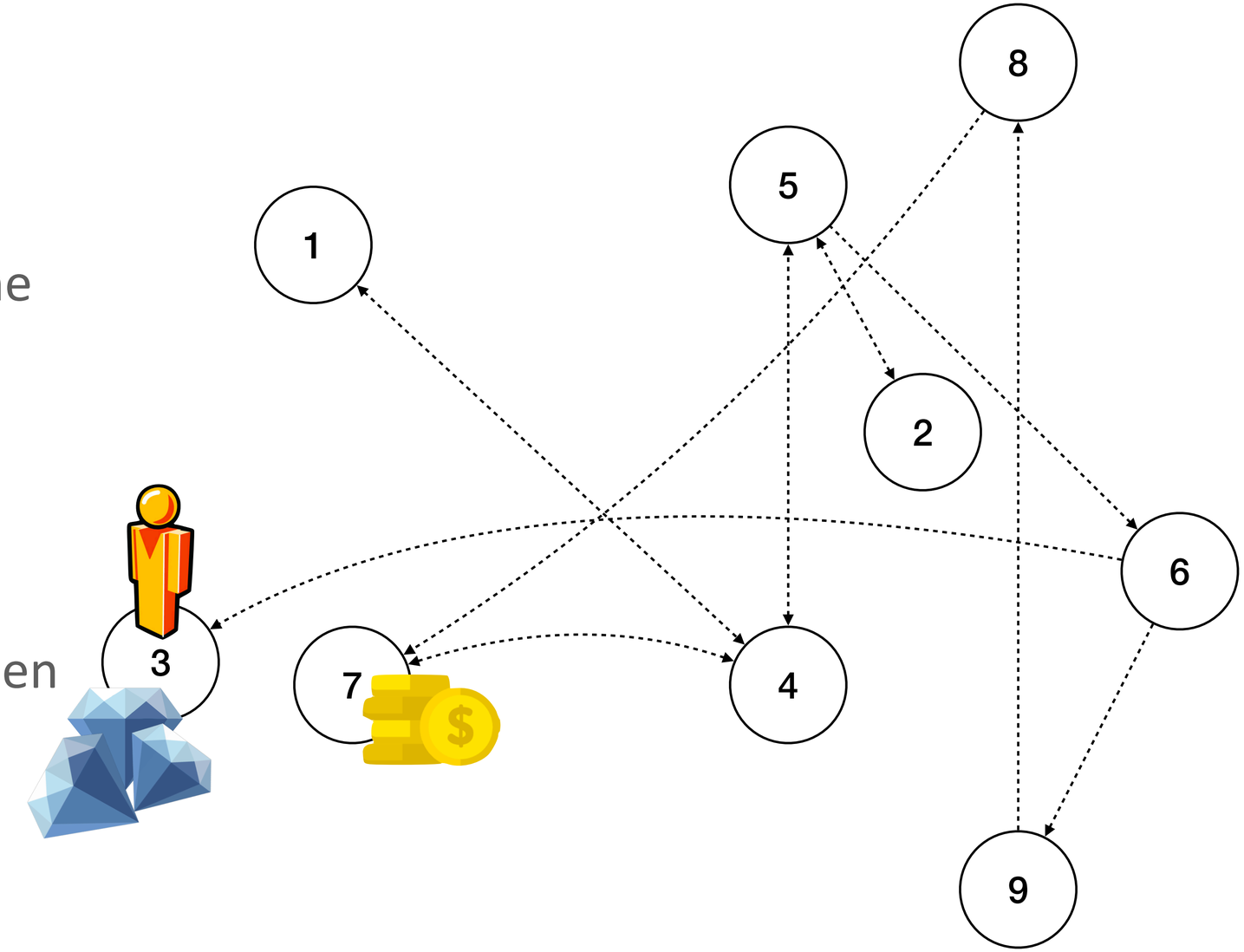
Let’s start with a puzzle…

- What’s next?
  - You may want to stay in 7, in the hope that you will get more money



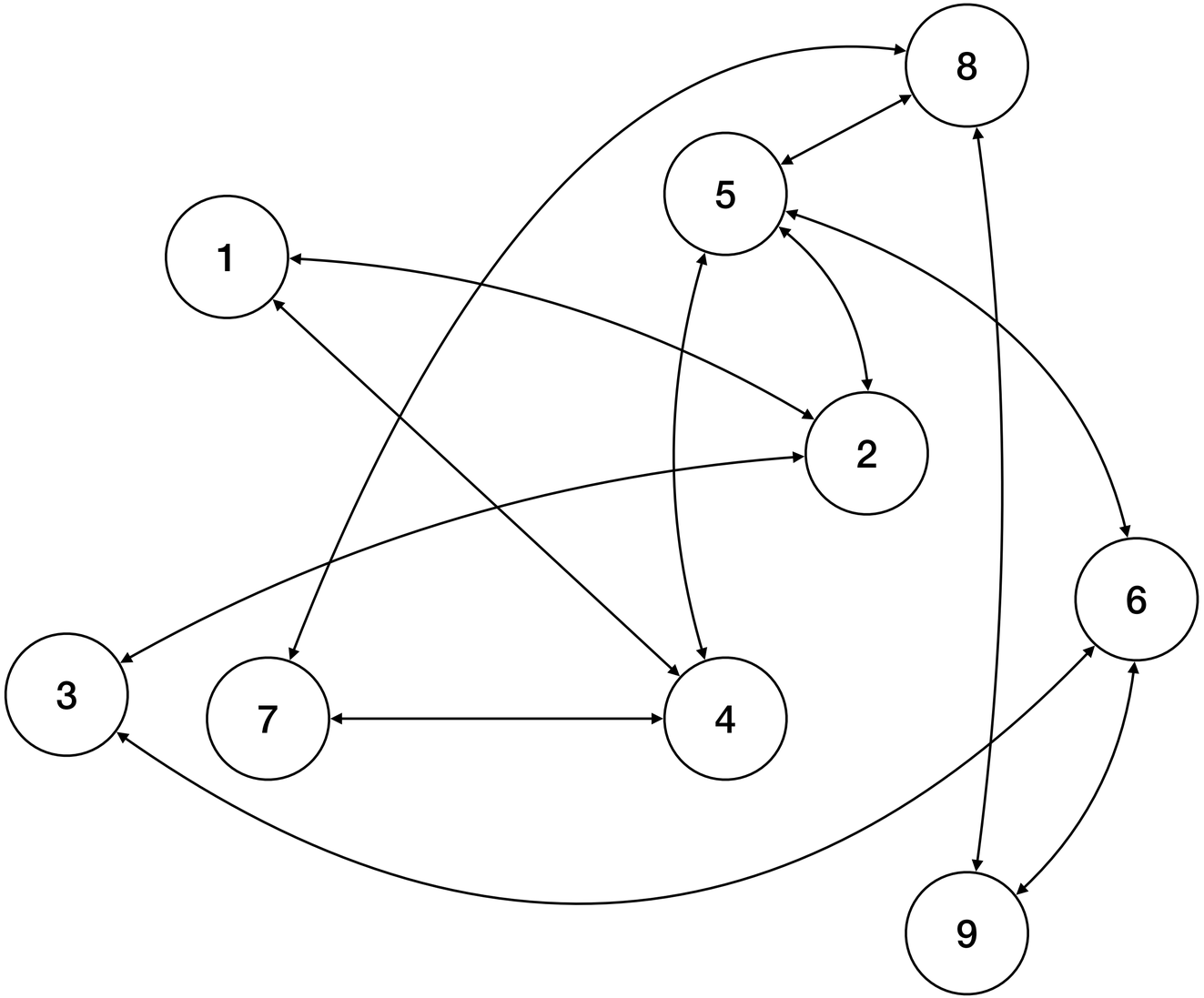
Let’s start with a puzzle…

- What’s next?
  - You may want to stay in 7, in the hope that you will get more money
  - You may also want to explore a little more, see if there’s a hidden treasure somewhere else…



So what's the point?

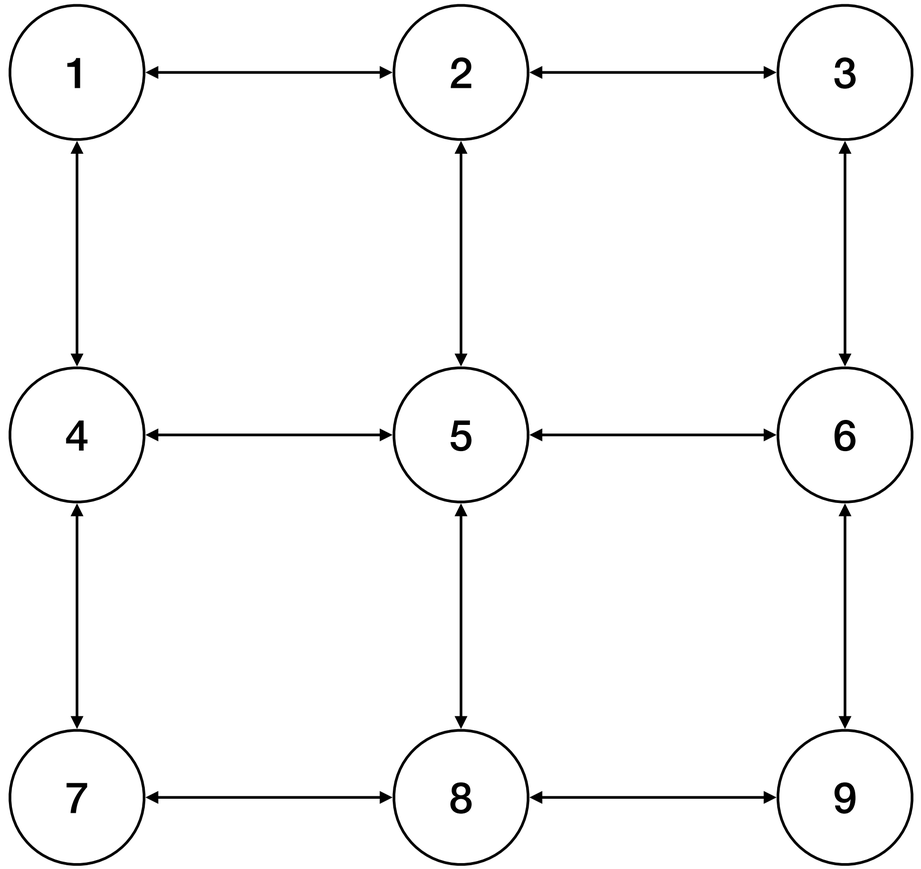
- First take away:
  - Everything would have been much easier if you knew this:



So what's the point?

- First take away:
  - Everything would have been much easier if you knew this:

A: Up
B: Down
C: Right
D: Left
E: Stay

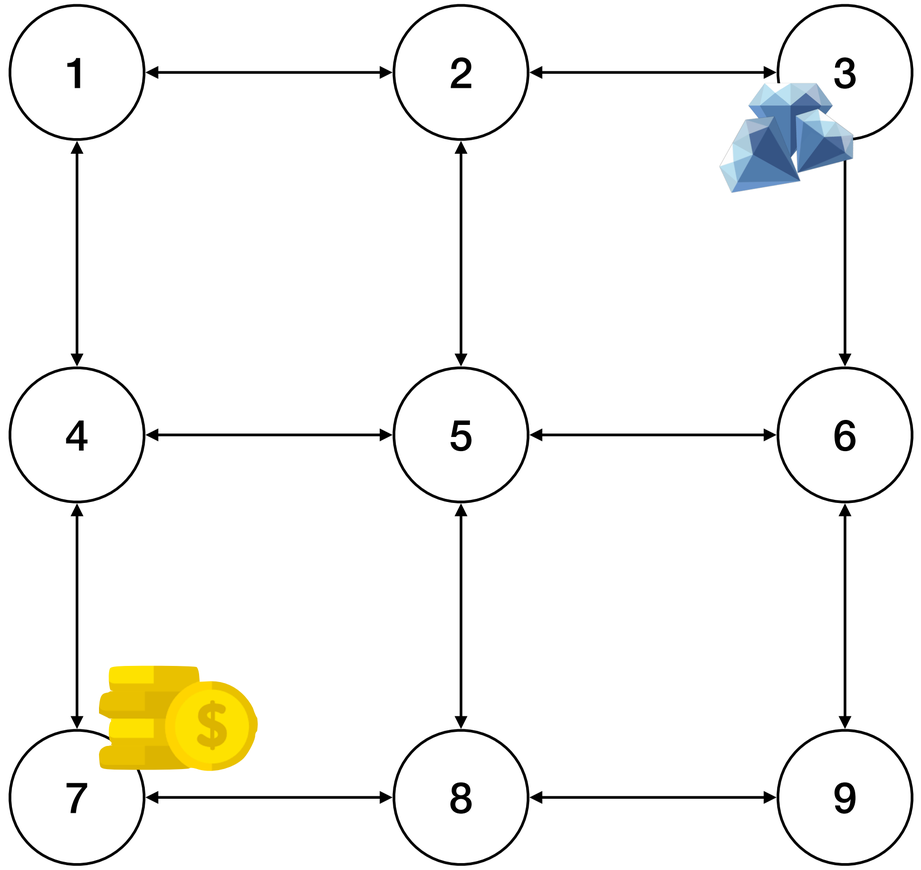


In other words: WHAT ACTIONS DO

So what's the point?

- Second take away:
  - Everything would have been even easier if you also knew this:

A: Up
B: Down
C: Right
D: Left
E: Stay



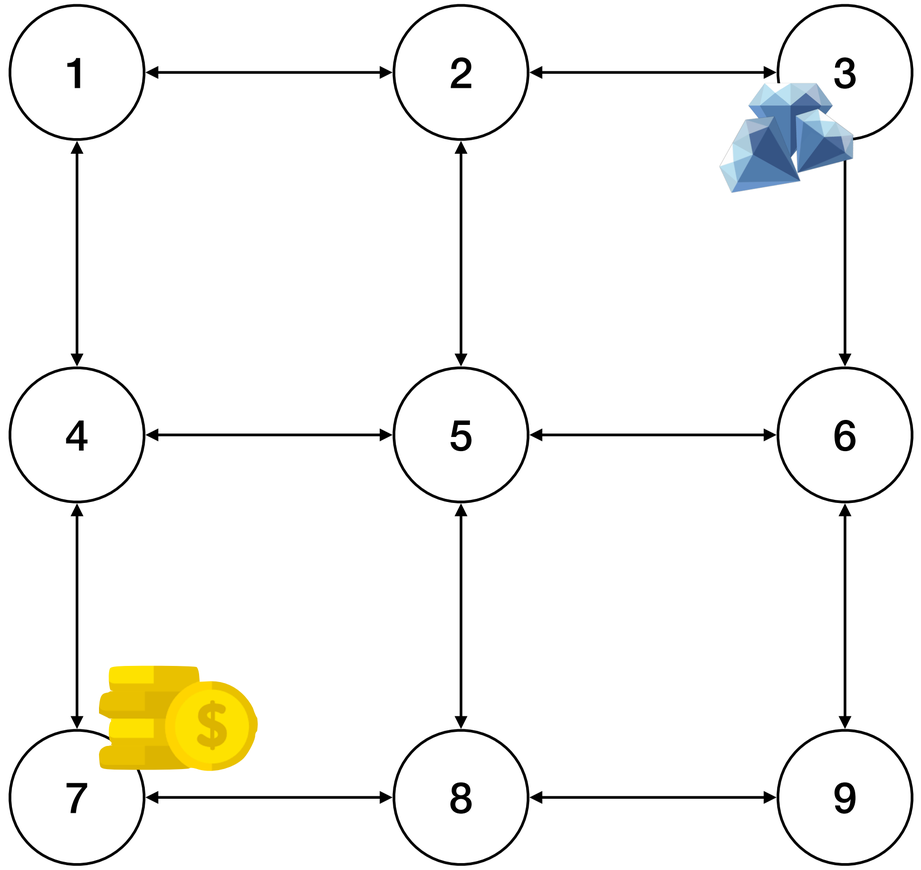
In other words: WHERE THE "MONEY" IS

So what's the point?

- Third take away:
  - If you had enough time, with some book-keeping, you'd eventually figure everything out

In other words: YOU WOULD LEARN

A: Up
B: Down
C: Right
D: Left
E: Stay



This is reinforcement learning

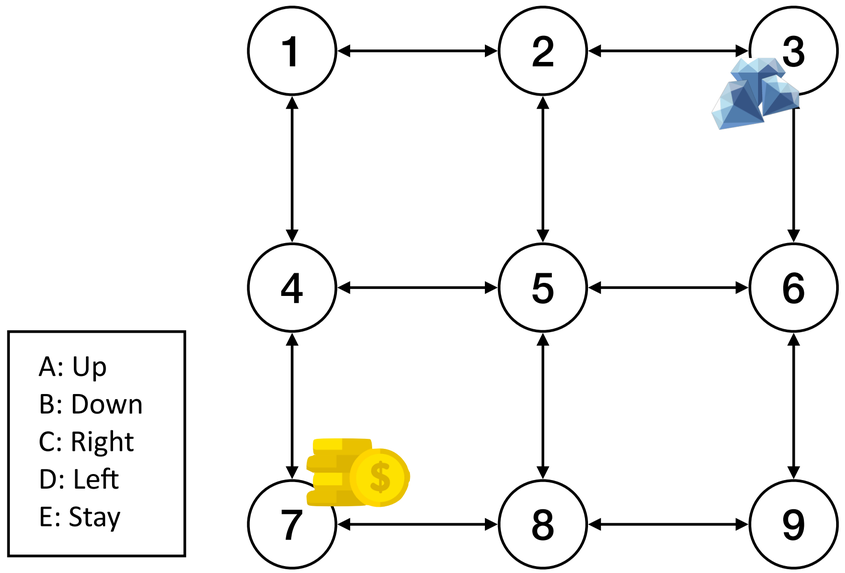
- Figuring out, through trial-and-error, …
  - … how to solve a problem…
  - … that takes multiple steps to be solved…
  - … without knowing what your actions do…
  - … without knowing what the goal is.

Key ingredients

- A reinforcement learning problem is described by…
  - The states ⟵ **What you need to know, in each moment, to choose your action**

Key ingredients

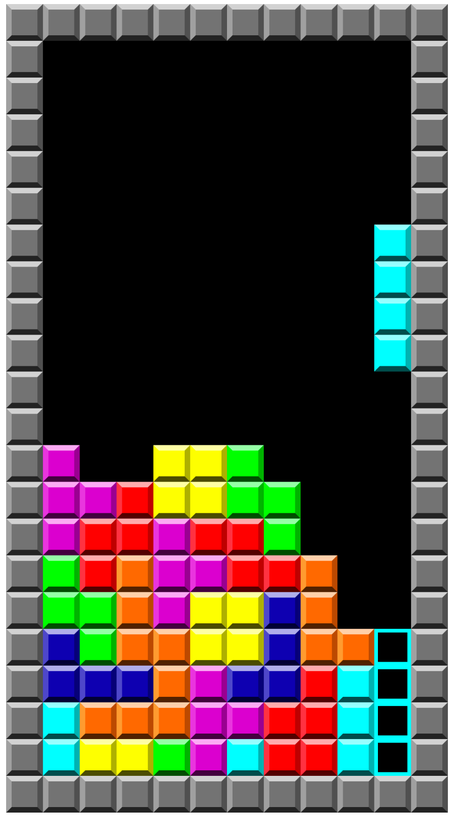
- A reinforcement learning problem is described by…
  - The states
  - E.g.:



What do you think the states are, here?

Key ingredients

- A reinforcement learning problem is described by…
  - The states
  - E.g.:



What do you think the states are, here?

Key ingredients

- A reinforcement learning problem is described by…
  - The states
  - E.g.:

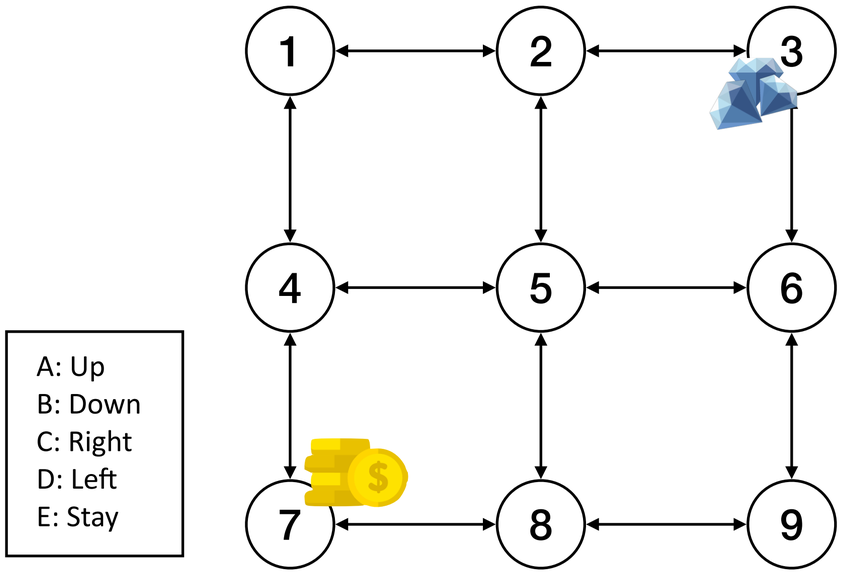
What do you think the states are, here?

Key ingredients

- A reinforcement learning problem is described by…
  - The states
  - The actions ⟵ **What you can do, in each moment**

Key ingredients

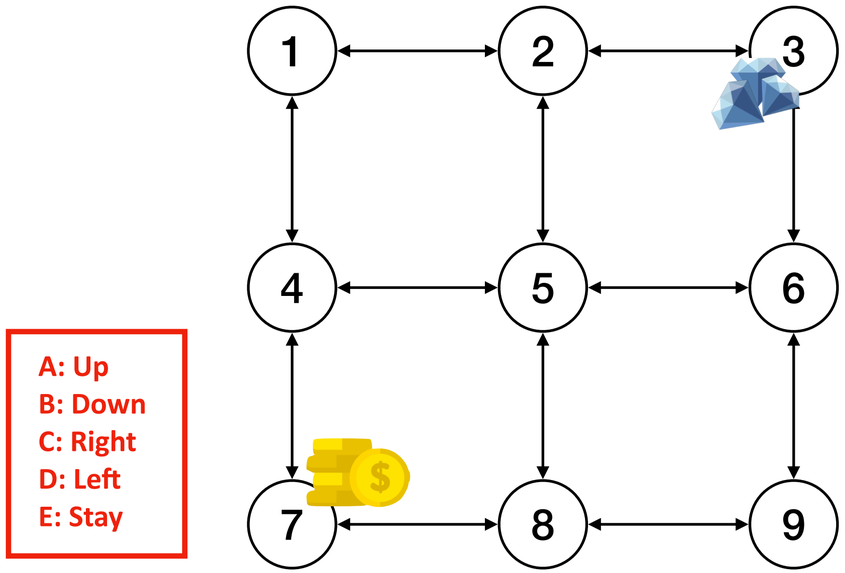
- A reinforcement learning problem is described by…
  - The states
  - The actions
  - E.g.:



What do you think the actions are, here?

Key ingredients

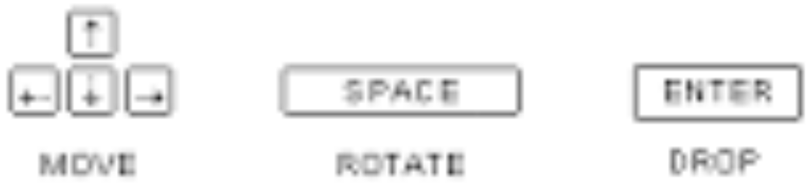
- A reinforcement learning problem is described by…
  - The states
  - The actions
  - E.g.:

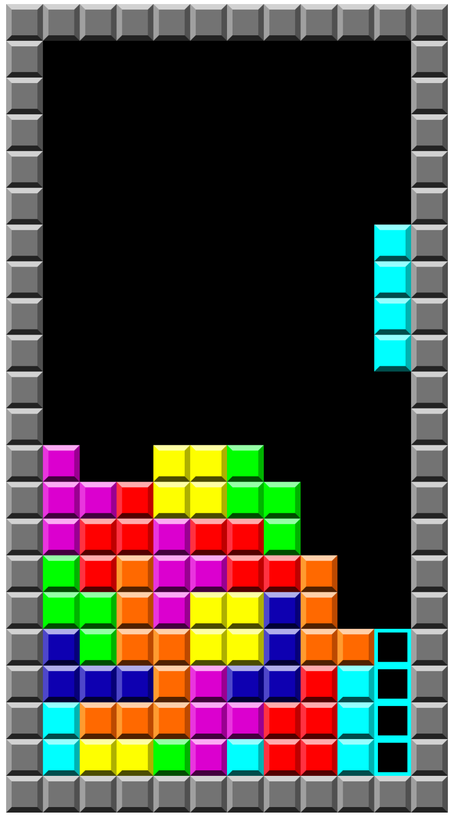


What do you think the actions are, here?

Key ingredients

- A reinforcement learning problem is described by…
  - The states
  - The actions
  - E.g.:





What do you think the actions are, here?

Key ingredients

- A reinforcement learning problem is described by…
  - The states
  - The actions
  - E.g.:

What do you think the actions are, here?

Key ingredients

- A reinforcement learning problem is described by…
  - The states
  - The actions
  - The **dynamics** ⟵ **What your actions do (their effects)**

Key ingredients

- A reinforcement learning problem is described by…
  - The states
  - The actions
  - The **dynamics** (we usually don’t know this)

Key ingredients

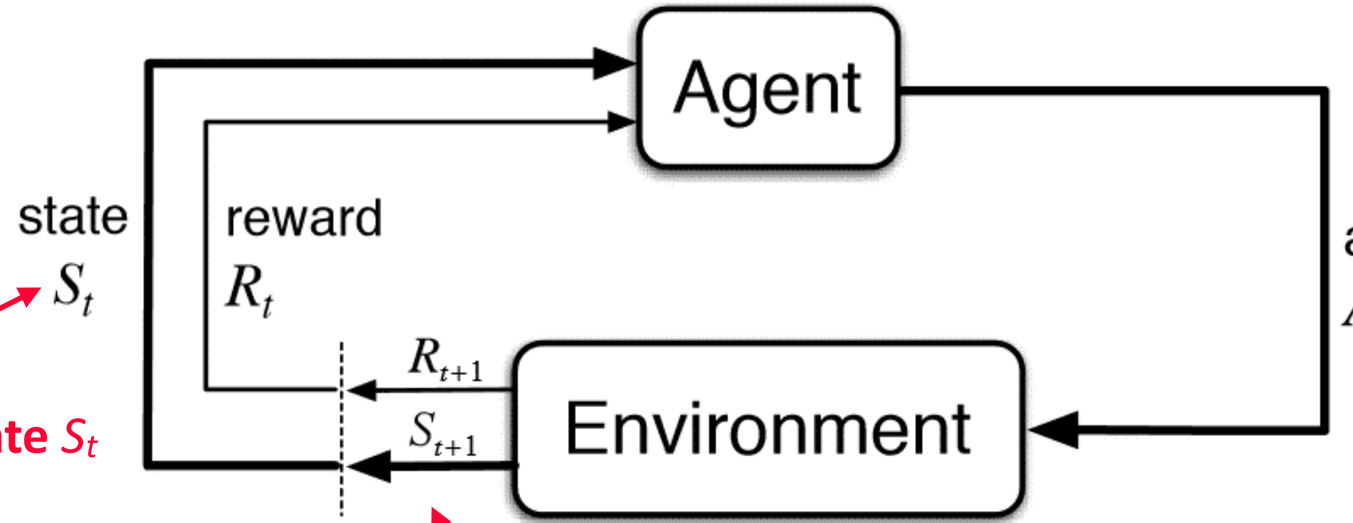
- A reinforcement learning problem is described by…
  - The states
  - The actions
  - The **dynamics**
  - The **goal** (usually we also don’t know this)

Key ingredients

- A reinforcement learning problem is described by…
  - The states
  - The actions
  - The **dynamics**
  - The **goal**
  - Some way to measure our **performance** (e.g., the total score in the game, or the total money collected, considering inflation)

The RL interaction

- At each step t



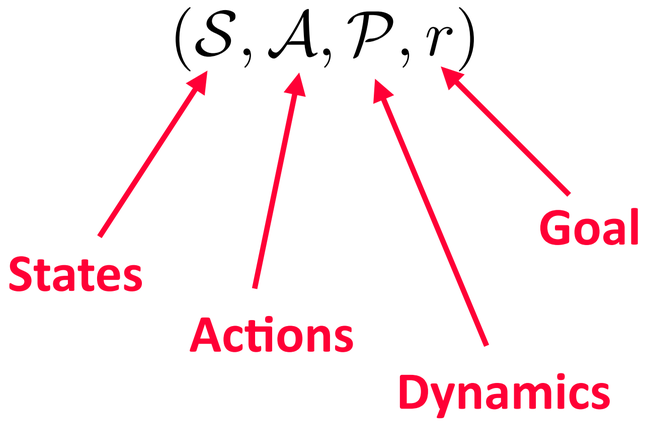
Agent observes the state S_t

It then selects an action A_t

State evolves according to the dynamics
Agent receives an instantaneous reward

Markov decision process

- The interaction between agent and environment is described using a Markov decision process

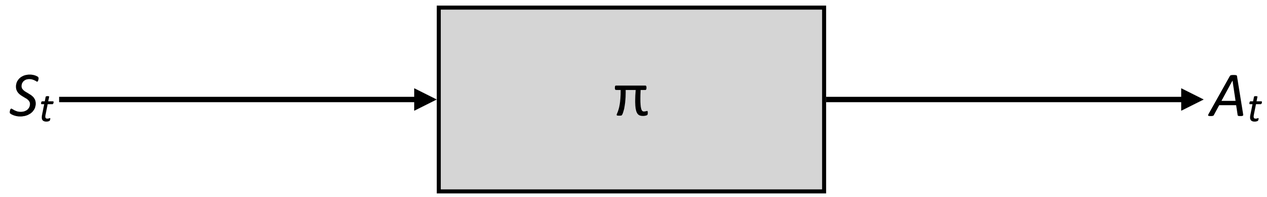


What about performance?

- The agent wants to figure out a way to select actions — a policy

What about performance?

- The agent wants to figure out a way to select actions — a policy



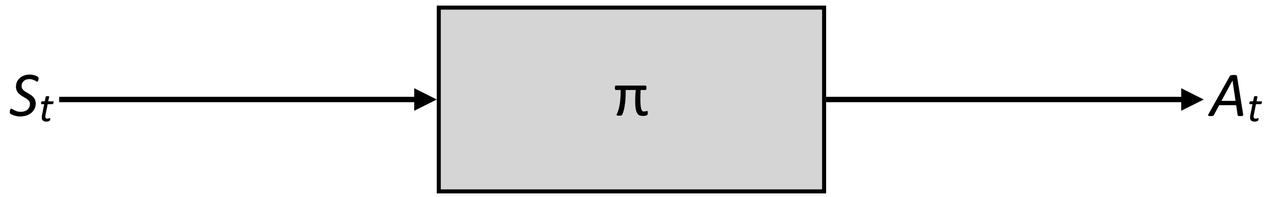
- … to get as much reward as possible…

$$\sum_{t=0}^{\infty} \gamma^t R_{t+1}$$

Inflation (discount)

What about performance?

- The agent wants to figure out a way to select actions — a policy



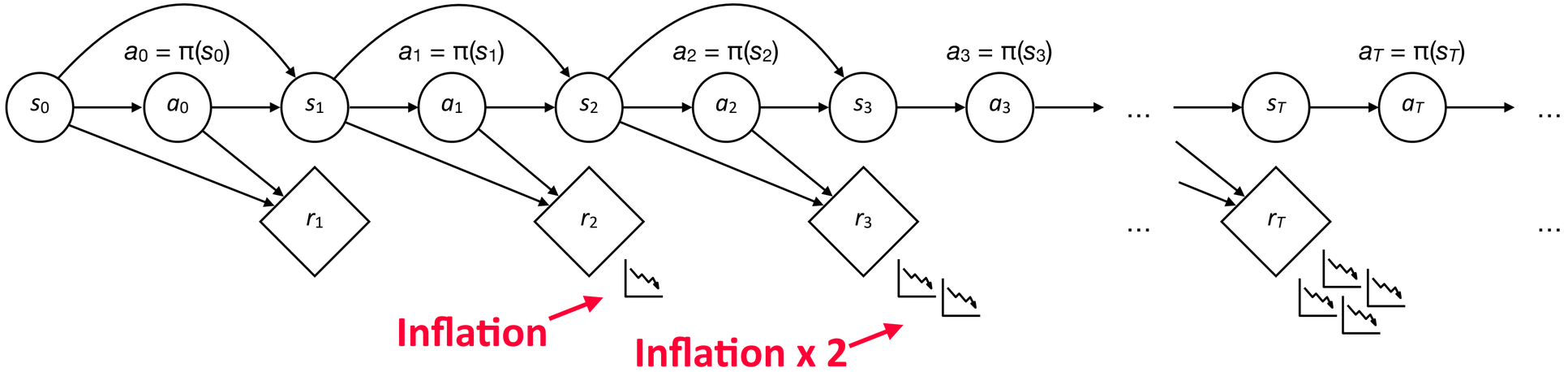
- … to get as much reward as possible…

$$\mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t R_{t+1} \right]$$

Because of uncertainty…

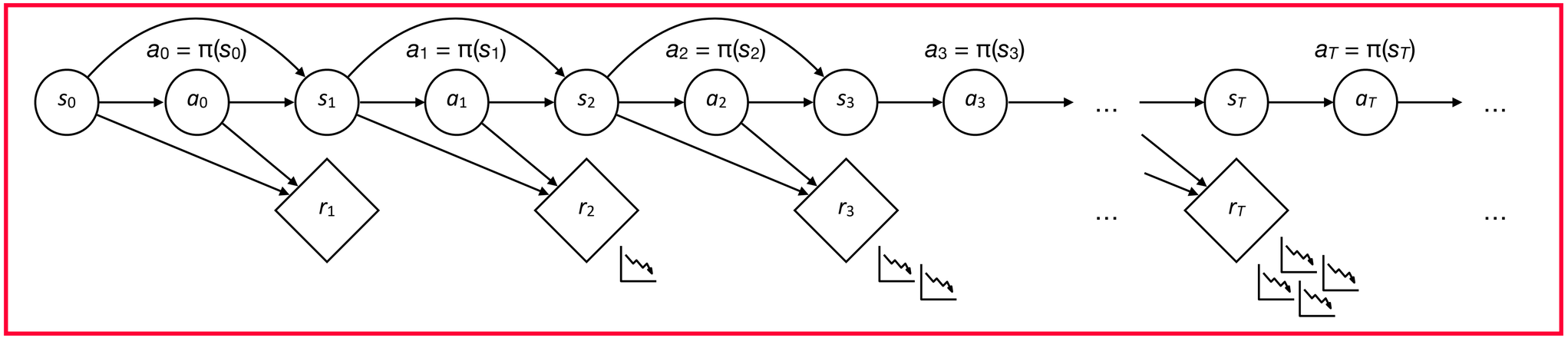
What does this mean?

- Fix a policy π



What does this mean?

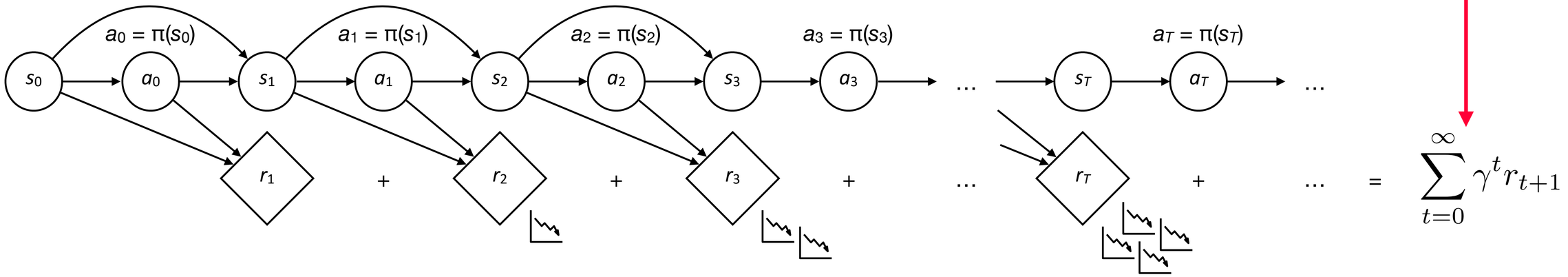
- Fix a policy π



Rollout

What does this mean?

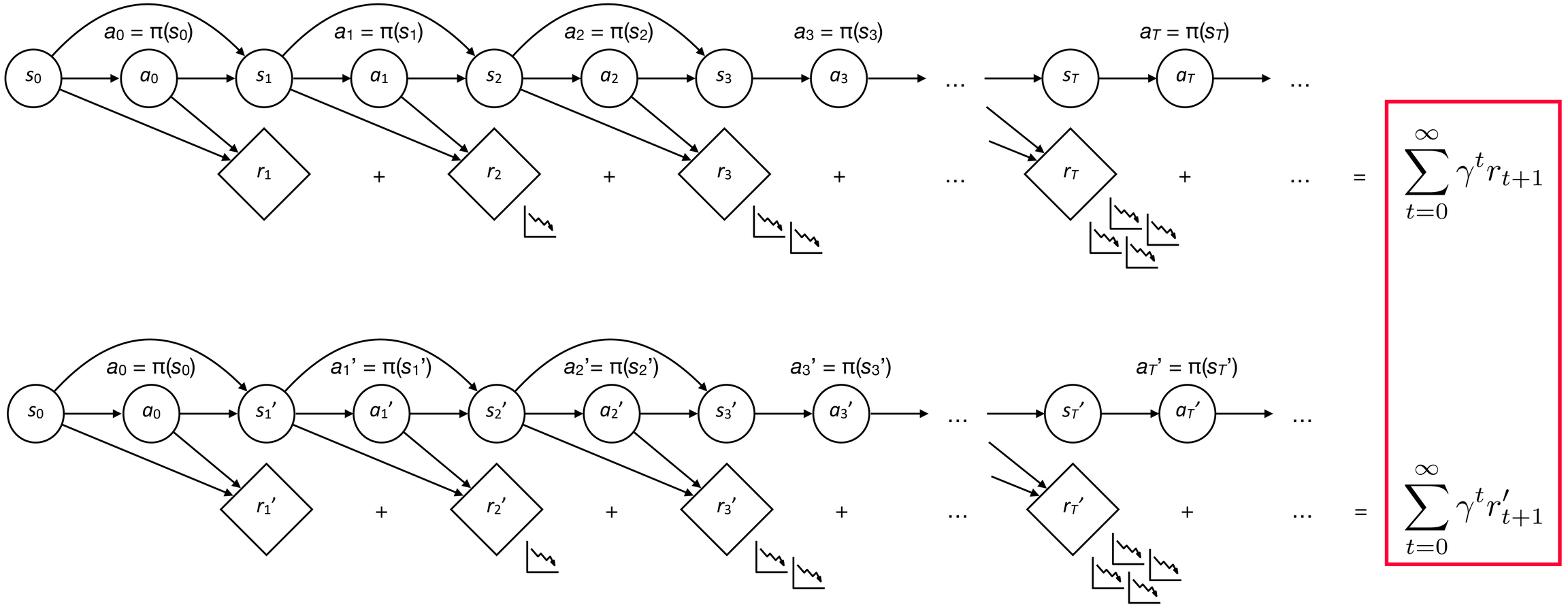
- Fix a policy π



Performance (return)

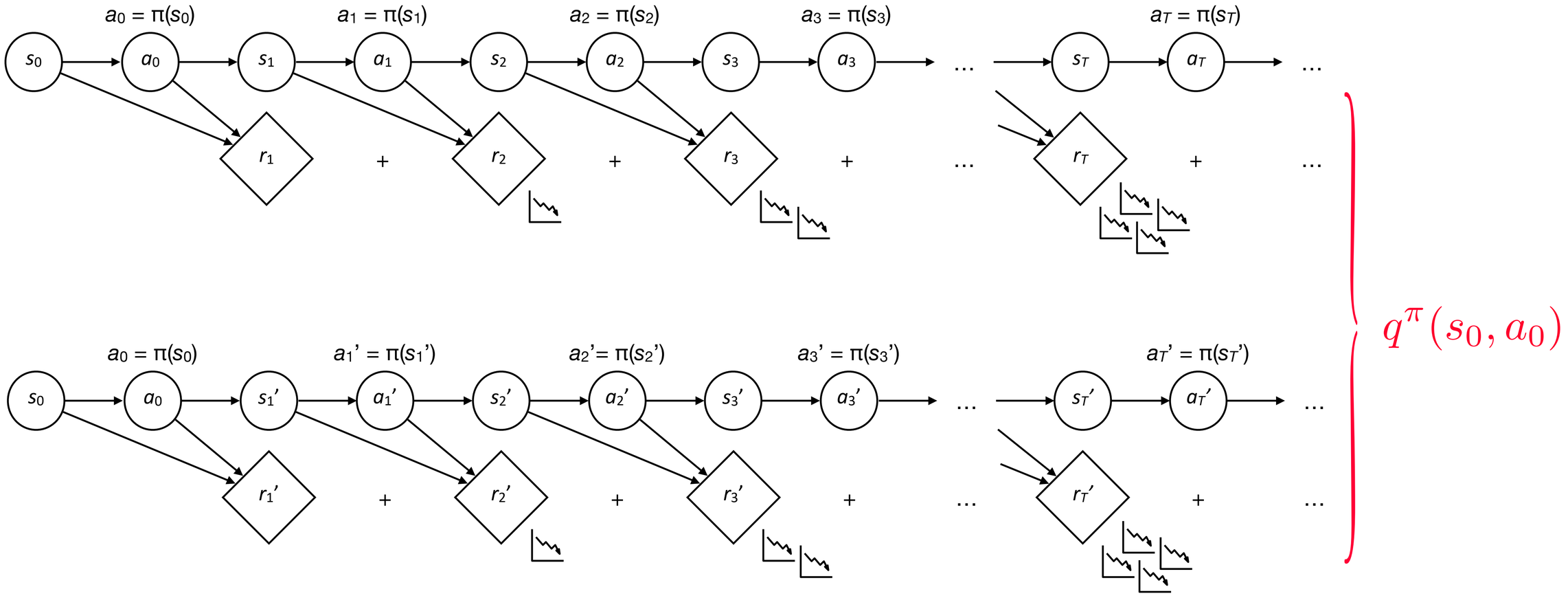
What does this mean?

- Fix a policy π



What does this mean?

- Fix a policy π



What is this “q”?

- From what we’ve seen,

$$q^{\pi}(s,a) = \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t R_{t+1} \mid S_0 = s, A_0 = a\right]$$

- It’s how much we expect to gain (considering inflation) if:
  - We start in *s*
  - Our first action is *a*
  - After that, we follow π

“Quality” of each action in each state

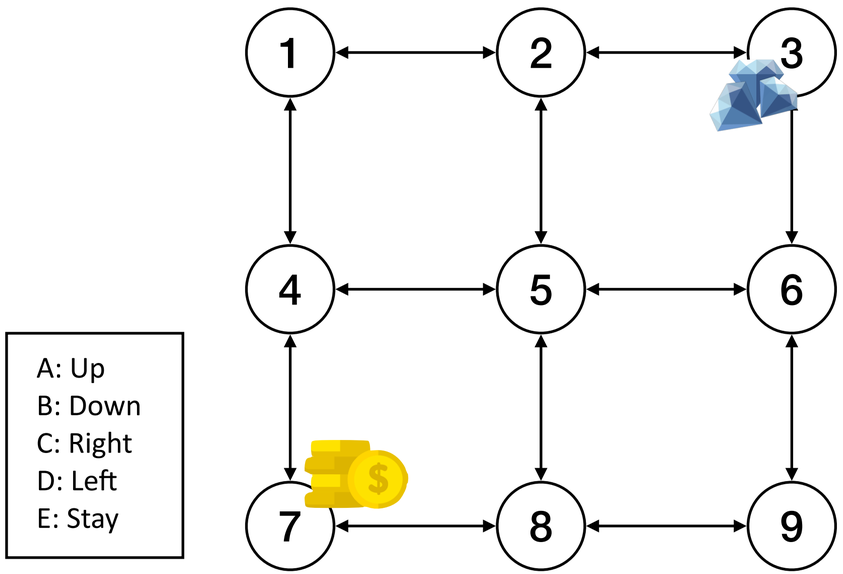
What is this “q”?

- Why should we care?
  - q provides with a long-term evaluation of the actions
  - We can use it to figure out the best actions to take!
- It verifies a nice relation:

$$q^{\pi}(s,a) = \mathbb{E}\left[R_t + \gamma q^{\pi}(S_{t+1}, \pi(S_{t+1})) \mid S_t = s, A_t = a\right]$$

What is this “q”?

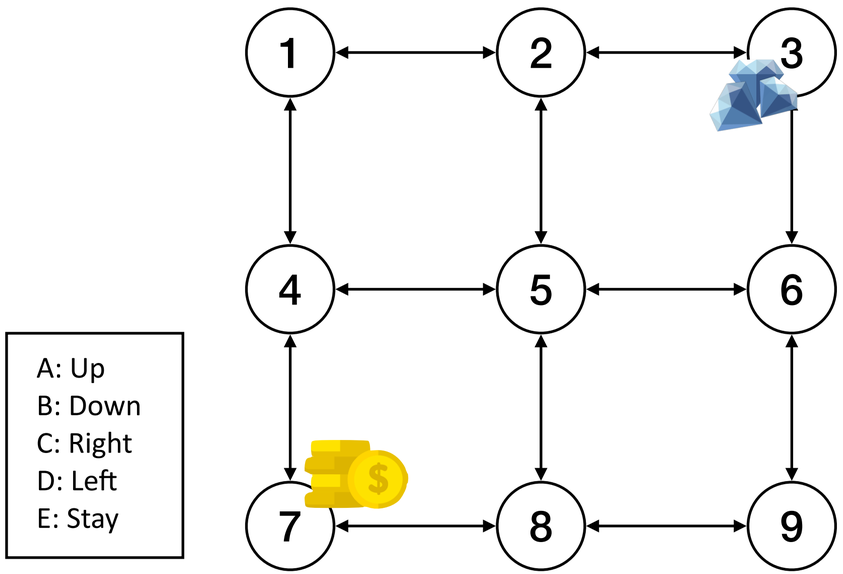
- For example, if we select actions randomly…



	Up	Down	Left	Right	Stay
1	3.1	2.9	3.1	5.6	3.3
2	5.9	3.9	3.4	14.0	6.0
3	19.9	11.8	11.8	19.9	20.0
4	3.1	4.0	2.9	3.6	3.0
5	5.7	3.0	3.0	5.7	3.8
6	14.0	3.4	3.9	5.9	6.0
7	4.0	5.1	5.1	4.0	5.2
8	3.6	2.9	4.0	3.1	3.0
9	5.6	3.1	2.9	3.2	3.3

What is this “q”?

- For example, if we select actions in the best possible way…



	Up	Down	Left	Right	Stay
1	35.4	31.8	35.4	39.4	35.5
2	39.9	35.8	35.8	44.4	40.0
3	49.9	45.4	45.4	49.9	50.0
4	35.0	29.1	31.4	35.0	31.5
5	39.4	31.8	31.8	39.4	35.5
6	44.4	35.8	35.8	39.9	40.0
7	32.1	29.8	29.8	32.1	29.9
8	35.0	31.4	29.1	35.0	31.5
9	39.4	35.4	31.8	35.4	35.5

TÉCNICO LISBOA

Outline

What will this presentation be all about?

- What is RL?
- **How to solve RL problems - the shallow view**
- How to solve RL problems - the deep view
- Shallow waters…
- Fun stuff

Our proto-RL algorithm

- We start with some policy π
- We compute $q^{\pi}(s, a)$ for all pairs (s, a)
  - How?

    …
- We improve our policy
  - How?

    $$\pi_{\text{improved}}(s) = \arg\max_{a} q^{\pi}(s, a)$$
- We repeat

How can we compute “q”?

- Open question: how do we compute $q^{\pi}(s, a)$?
- Simple idea — we do it “*a la* supervised learning”:

$$\hat{q} = \arg\min_{q} \sum_{n=1}^{N} \|q^{\pi}(s_n, a_n) - q(s_n, a_n)\|^2$$

- Using SGD, this would yield an algorithm looking like this:

$$\hat{q}(s_n, a_n) \leftarrow \hat{q}(s_n, a_n) + \alpha(q^{\pi}(s_n, a_n) - \hat{q}(s_n, a_n))$$

How can we compute “q”?

- Open question: how do we compute $q^{\pi}(s, a)$ for all (s, a)?
- Simple idea — we do it “*a la* supervised learning”:

$$\hat{q} = \arg\min_{q} \sum_{n=1}^{N} \| q^{\pi}(s_n, a_n) - q(s_n, a_n) \|^2$$

- Using SGD, this would yield an algorithm looking like this:

$$\hat{q}(s_n, a_n) \leftarrow \hat{q}(s_n, a_n) + \alpha(q^{\pi}(s_n, a_n) - \hat{q}(s_n, a_n))$$

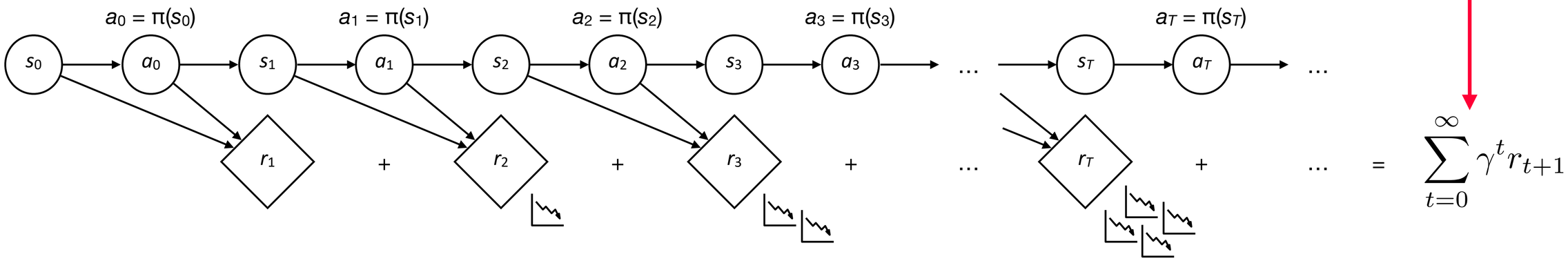
We don’t have these “target” values…

Idea n. 1: We use rollouts

- For a fixed policy π,

$$q^{\pi}(s,a) = \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t R_{t+1} \mid S_0 = s, A_0 = a\right]$$

Estimate (sample) of $q^{\pi}(s_0, a_0)$



- We use the returns as “targets”

Monte-Carlo RL

- We start with some policy π
- We run some rollouts using π for different initial states and actions and set

$$\hat{q}(s_n, a_n) \leftarrow \hat{q}(s_n, a_n) + \alpha \left(\sum_{t=0}^{T} \gamma^t r_{t+1} - \hat{q}(s_n, a_n) \right)$$

- We improve our policy

$$\pi_{\text{improved}}(s) = \arg\max_a \hat{q}(s, a)$$

- We repeat

Monte-Carlo RL

- Monte Carlo RL suffers from large variance:

$$\hat{q}(s_n, a_n) \leftarrow \hat{q}(s_n, a_n) + \alpha \left(\sum_{t=0}^{T} \gamma^t r_{t+1} - \hat{q}(s_n, a_n) \right)$$

For long trajectories the target may take very different values

Idea n. 2: Temporal difference

- In Monte-Carlo RL, we used the fact that

$$q^{\pi}(s,a) = \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty}\gamma^{t}R_{t+1} \mid S_0 = s, A_0 = a\right]$$

We used samples of this as target

- However, it is also true that

$$q^{\pi}(s,a) = \mathbb{E}\left[R_t + \gamma q^{\pi}(S_{t+1}, \pi(S_{t+1})) \mid S_t = s, A_t = a\right]$$

What if we use samples of this as target?

Idea n. 2: Temporal difference

- In Monte-Carlo RL, we used the fact that

$$q^\pi(s,a) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t R_{t+1} \mid S_0 = s, A_0 = a \right]$$

We used samples of this as target

- However, it is also true that

$$q^\pi(s,a) = \mathbb{E}\left[R_t + \gamma q^\pi(S_{t+1}, \pi(S_{t+1})) \mid S_t = s, A_t = a \right]$$

What if we use samples of this as target? → We don't have this term… → We bootstrap!

SARSA

- We start with some policy π
- At each step *t*, we interact with the environment for some time using π and compute

$$\hat{q}(s_t, a_t) \leftarrow \hat{q}(s_t, a_t) + \alpha \left(r_{t+1} + \gamma \hat{q}(s_{t+1}, a_{t+1}) - \hat{q}(s_t, a_t) \right)$$

- We improve our policy

$$\pi_{\text{improved}}(s) = \arg\max_a \hat{q}(s, a)$$

- We repeat

SARSA

- The quantity

  $$\delta_t = r_{t+1} + \gamma \hat{q}(s_{t+1}, a_{t+1}) - \hat{q}(s_t, a_t)$$

  is called a temporal difference

- The action a_{t+1} is selected according to π

What if we merge the policy improvement with the above update?

Q-learning

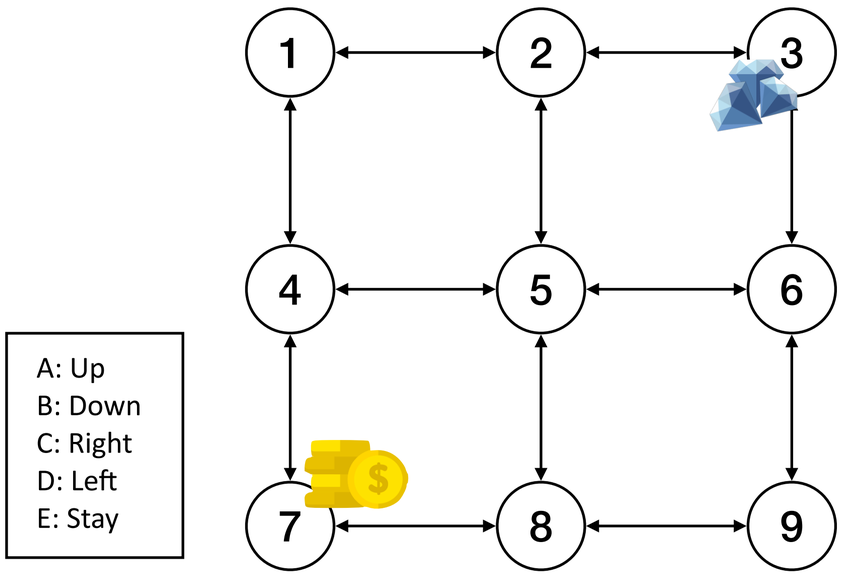
- We start with some policy π
- At each step *t*, we interact with the environment using π and compute

$$\hat{q}(s_t, a_t) \leftarrow \hat{q}(s_t, a_t) + \alpha \left(r_{t+1} + \gamma \max_{a'} \hat{q}(s_{t+1}, a') - \hat{q}(s_t, a_t) \right)$$

- Very similar to SARSA, but uses a different action in the next step term
  - It does not depend on the policy π (it is off-policy)
  - It converges to the best q (i.e., the q for the best policy)

Running Q-learning on our example

- After 5,000 steps, we get:



	Up	Down	Left	Right	Stay
1	35.8	31.8	31.5	39.2	34.6
2	39.0	35.4	36.1	44.9	39.2
3	49.9	45.6	45.3	49.9	50.0
4	31.2	28.6	31.4	35.7	31.8
5	39.8	31	32	40	35.6
6	44.9	34.4	35.9	39.2	39.4
7	27.2	19.9	25	32.7	25.1
8	35.1	31.1	30	31	31.1
9	39.9	29.5	29.9	29.3	27.4

Outline

What will this presentation be all about?

- What is RL?
- How to solve RL problems - the shallow view
- **How to solve RL problems - the deep view**
- Shallow waters…
- Fun stuff

Let us revisit the computation of q^π

- How do we compute $q^\pi(s, a)$?
- We departed from a "supervised learning" perspective:

$$\hat{q} = \arg\min_q \sum_{n=1}^{N} \|\text{target} - q(s_n, a_n)\|^2$$

- All our algorithms look like this:

$$\hat{q}(s_n, a_n) \leftarrow \hat{q}(s_n, a_n) + \alpha(\text{target} - \hat{q}(s_n, a_n))$$

Monte Carlo RL target:

$$\sum_{t=0}^{T} \gamma^t r_{t+1}$$

SARSA target:

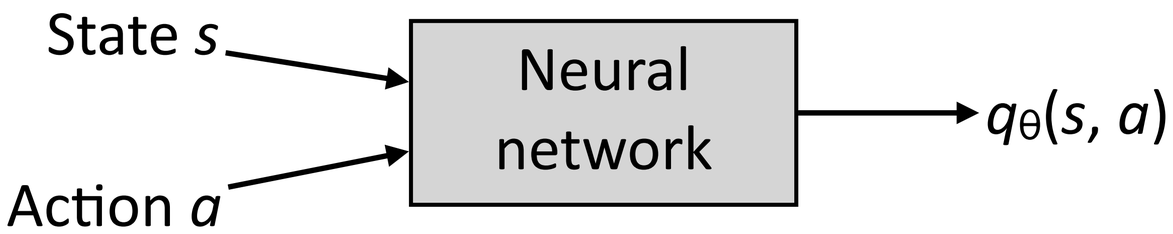
$$r_{t+1} + \gamma \hat{q}(s_{t+1}, a_{t+1})$$

Q-learning target:

$$r_{t+1} + \gamma \max_{a'} \hat{q}(s_{t+1}, a')$$

Deep RL

- We can replace our estimate by a neural network

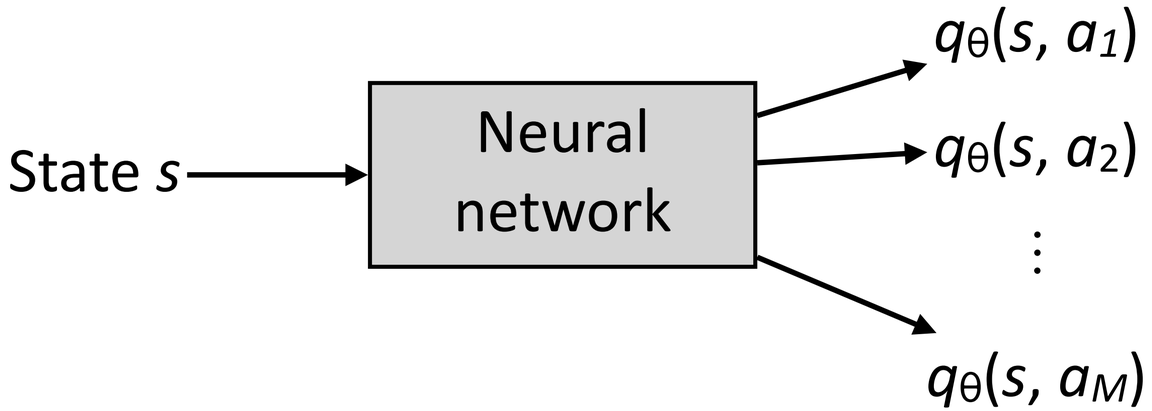


- Our algorithms will now look like this:

$$\theta \leftarrow \theta + \alpha \nabla_\theta q_\theta(s_n, a_n)(\text{target} - q_\theta(s_n, a_n))$$

Deep RL

- We can replace our estimate by a neural network



- Our algorithms will now look like this:

$$\theta \leftarrow \theta + \alpha \nabla_\theta q_\theta(s_n, a_n)(\text{target} - q_\theta(s_n, a_n))$$

Monte-Carlo RL

- We start with some policy π
- We run some rollouts using π for different initial states and update

$$\theta \leftarrow \theta + \alpha \nabla_\theta q_\theta(s_0, a_0) \left(\sum_{t=0}^{T} \gamma^t r_{t+1} - q_\theta(s_0, a_0) \right)$$

Monte Carlo target

- We improve our policy

$$\pi_{\text{improved}}(s) = \arg\max_a q_\theta(s, a)$$

- We repeat

SARSA

- We start with some policy π
- At each step *t*, we interact with the environment for some time using π and compute

$$\theta \leftarrow \theta + \alpha \nabla_\theta q_\theta(s_t, a_t) \left(r_{t+1} + \gamma q_\theta(s_{t+1}, a_{t+1}) - q_\theta(s_t, a_t) \right)$$

- We improve our policy

$$\pi_{\text{improved}}(s) = \arg\max_a \hat{q}(s, a)$$

- We repeat

SARSA target

Q-learning

- We start with some policy π
- At each step *t*, we interact with the environment using π and compute

$$\theta \leftarrow \theta + \alpha \nabla_\theta q_\theta(s_t, a_t) \left(r_{t+1} + \gamma \max_{a'} q_\theta(s_{t+1}, a') - q_\theta(s_t, a_t) \right)$$

Q-learning target

Q-learning

- We start with some policy π
- At each step *t*, we interact with the environment using π and compute

$$\theta \leftarrow \theta + \alpha \nabla_\theta q_\theta(s_t, a_t) \left(r_{t+1} + \gamma \max_{a'} q_\theta(s_{t+1}, a') - q_\theta(s_t, a_t) \right)$$

Q-learning may diverge when used with approximations

Q-learning

- We start with some policy π
- At each step *t*, we interact with the environment using π and compute

$$\theta \leftarrow \theta + \alpha \nabla_\theta q_\theta(s_t, a_t) \left(r_{t+1} + \gamma \max_{a'} q_\theta(s_{t+1}, a') - q_\theta(s_t, a_t) \right)$$

  - Q-learning falls prey to the so-called “deadly triad”:
    - Bootstrapping
    - Off-policy learning
    - Function approximation

Q-learning

- We use a second network to compute the targets and thus “avoid” bootstrapping
  - The second network (let’s call it “target network”) is a copy of the main network
  - The target network is not updated (often)
- The resulting algorithm is known as DQN and looks like:

$$\theta \leftarrow \theta + \alpha \nabla_\theta q_\theta(s_t, a_t) \left(r_{t+1} + \gamma \max_{a'} q_{\text{target}}(s_{t+1}, a') - q_\theta(s_t, a_t) \right)$$

We use the “target network” to compute the targets

TÉCNICO LISBOA

… however…

Going back to our proto-RL algorithm

- We start with some policy π
- We compute/approximate $q^{\pi}(s, a)$ for all pairs (s, a)
- We improve our policy

$$\pi_{\text{improved}}(s) = \arg\max_a q^{\pi}(s, a)$$

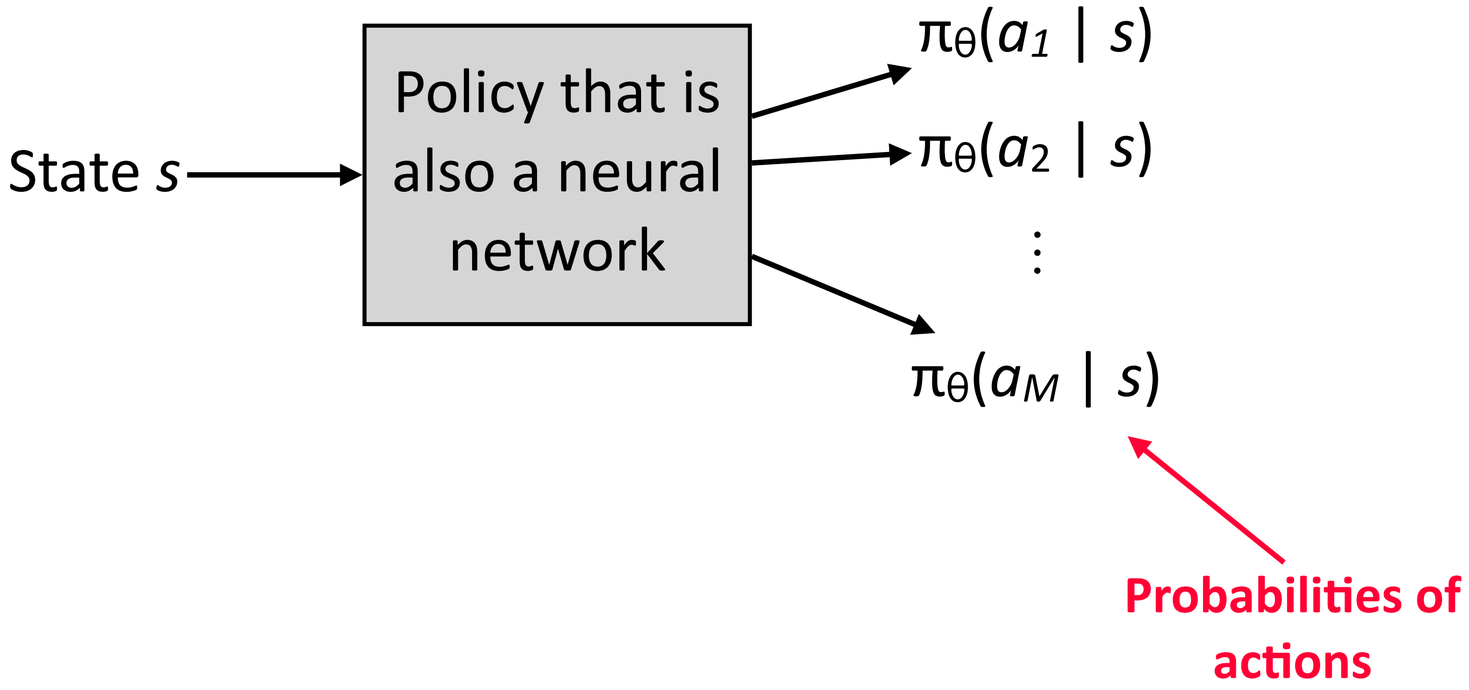
- We repeat

Objection n. 1: What if the actions are continuous?

Objection n. 2: Why do we need q^{π}? Why not optimize π directly?

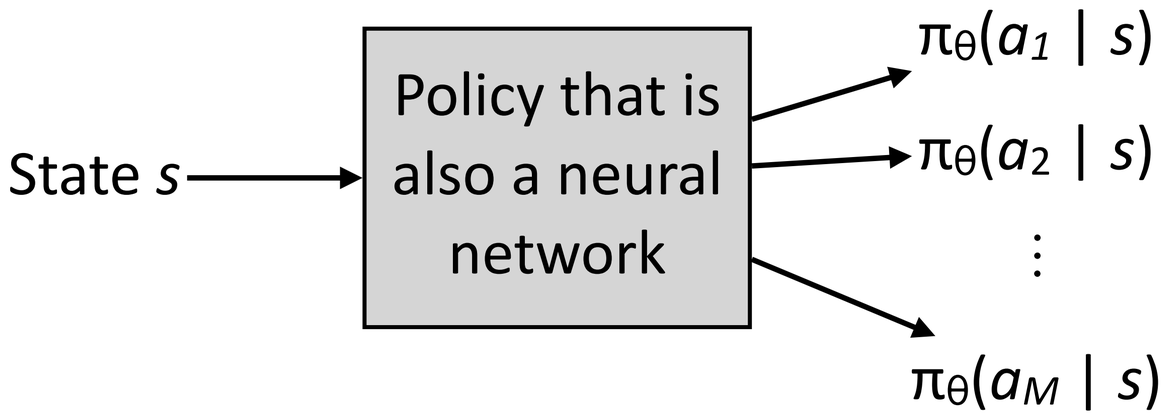
Idea n. 3: Optimize π directly

- We start with a policy represented, for example, using a neural network:



Idea n. 3: Optimize π directly

- We start with a policy represented, for example, using a neural network:



- Optimize the network to yield large sums of rewards!

But how do we compute

$$\nabla_\theta \mathbb{E}_{\pi_\theta} \left[\sum_{t=0}^{\infty} \gamma^t R_{t+1} \right] ?$$

Policy gradient

- It turns out that the gradient that we need is given by:

$$\mathbf{grad} = \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{\infty} \gamma^t \nabla_\theta \log \pi_\theta(A_t \mid S_t) q_{\pi_\theta}(S_t, A_t)\right]$$

- This looks rather ugly
- However, we can compute it using rollouts!
- This method of improving the policy using gradient ascent is called REINFORCE

REINFORCE

- We start with some policy π_θ
- We run some rollouts using π_θ for different initial states and actions
- We improve our policy

$$\theta \leftarrow \theta + \alpha \sum_{t=0}^{T} \gamma^t \nabla_\theta \log \pi_\theta(s_t, a_t) \hat{q}(s_t, a_t)$$

- We repeat

REINFORCE

- We start with some policy π_θ
- We run some rollouts using π_θ for different initial states and actions
- We improve our policy

$$\theta \leftarrow \theta + \alpha \sum_{t=0}^{T} \gamma^t \nabla_\theta \log \pi_\theta(s_t, a_t) \sum_{\tau=t}^{T} \gamma^{\tau-t} r_{\tau+1}$$

- We repeat

Other variations

- In the update

$$\theta \leftarrow \theta + \alpha \sum_{t=0}^{T} \gamma^t \nabla_\theta \log \pi_\theta(s_t, a_t) \hat{q}(s_t, a_t)$$

  we can use other ways of estimating q^{π_θ}

- For example, we can use a SARSA update

Other variations

- We start with some policy π_θ
- At each step *t*, we interact with the environment for some time using π and compute

$$\mathbf{w} \leftarrow \mathbf{w} + \alpha \nabla_{\mathbf{w}} q_{\mathbf{w}}(s_t, a_t) \left(r_{t+1} + \gamma q_{\mathbf{w}}(s_{t+1}, a_{t+1}) - q_{\mathbf{w}}(s_t, a_t)\right)$$

SARSA update

- We improve our policy

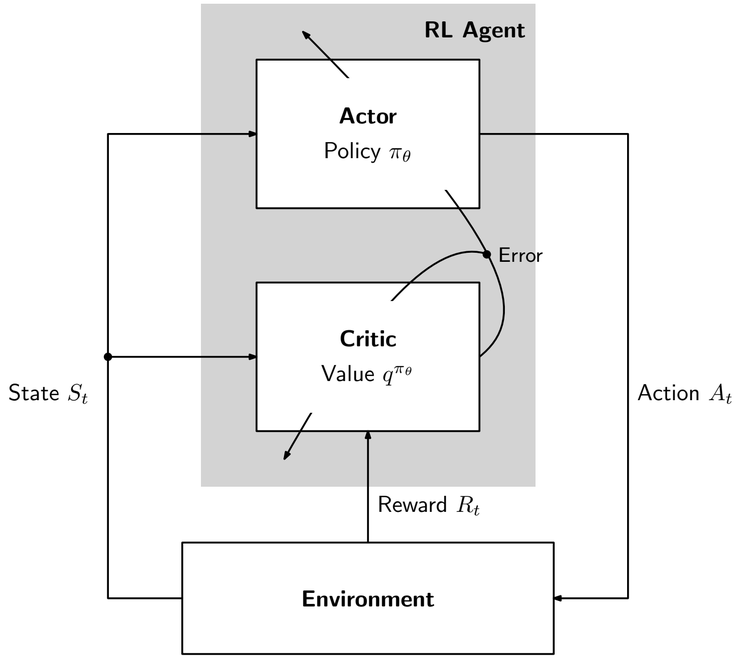
$$\theta \leftarrow \theta + \alpha \sum_{t=0}^{T} \gamma^t \nabla_\theta \log \pi_\theta(s_t, a_t) q_{\mathbf{w}}(s_t, a_t)$$

Policy update

- We repeat

Actor-critic architecture

- The resulting algorithm is known as actor-critic, since its architecture looks like this:



Variations

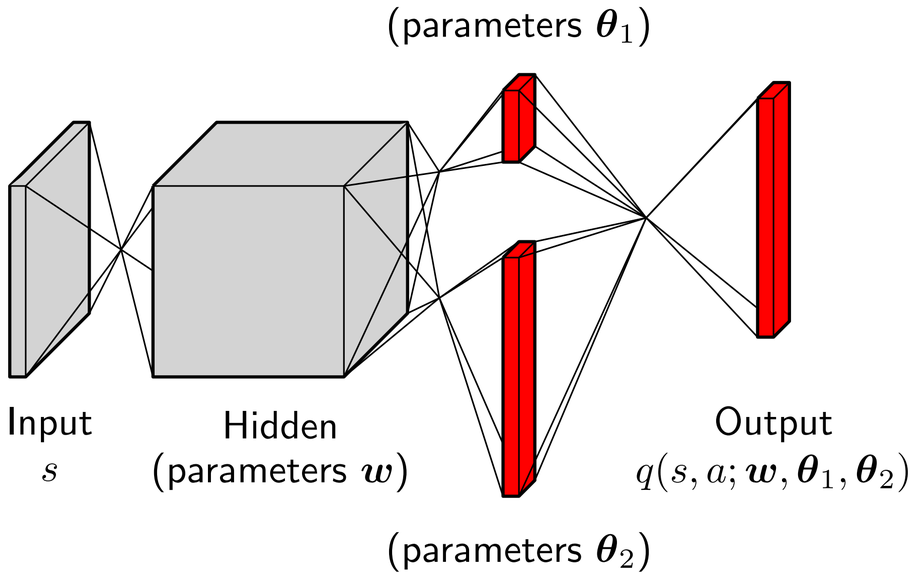
- There are many variations of these basic algorithms

Variations

- Even with the target network, DQN still has problems
- Some variations have been proposed to improve the vanilla DQN:
  - Maximizing action from original network and target value from target network (DDQN)

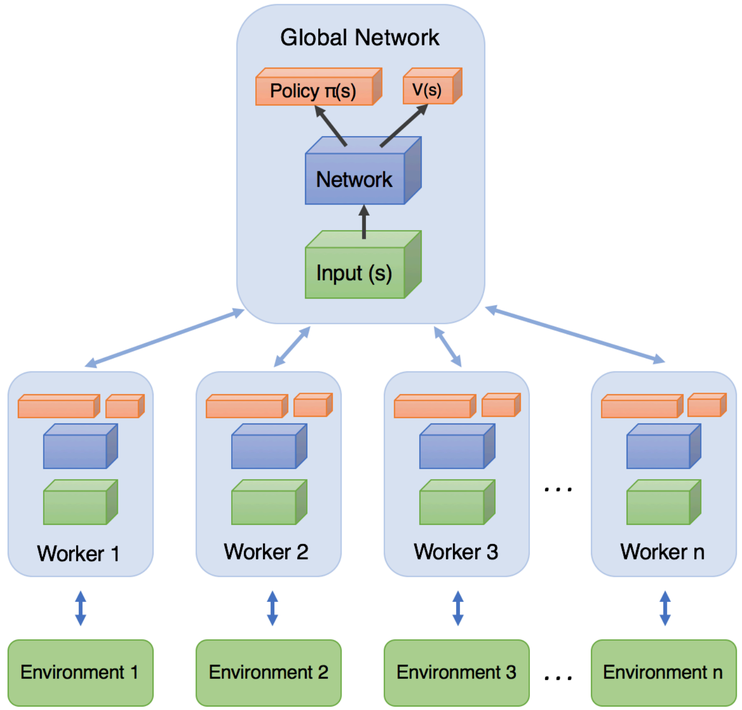
Variations

- Even with the target network, DQN still has problems
- Some variations have been proposed to improve the vanilla DQN:
  - Use a “split layer” before the output (dueling network)



Variations

- Actor critic methods can be made more efficient by distributing the computation (A3C)



Variations

- Policy-gradient methods can be made more data efficient by allowing only small variations to the original policy (TRPO, PPO)

Outline

What will this presentation be all about?

- What is RL?
- How to solve RL problems - the shallow view
- How to solve RL problems - the deep view
- **Shallow waters…**
- Fun stuff

TÉCNICO LISBOA

… actually…

Outline

What will this presentation be all about?

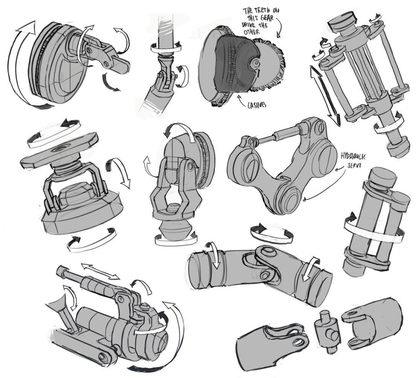
- What is RL?
- How to solve RL problems - the shallow view
- How to solve RL problems - the deep view
- **Shallow waters…**
- Fun stuff

Outline

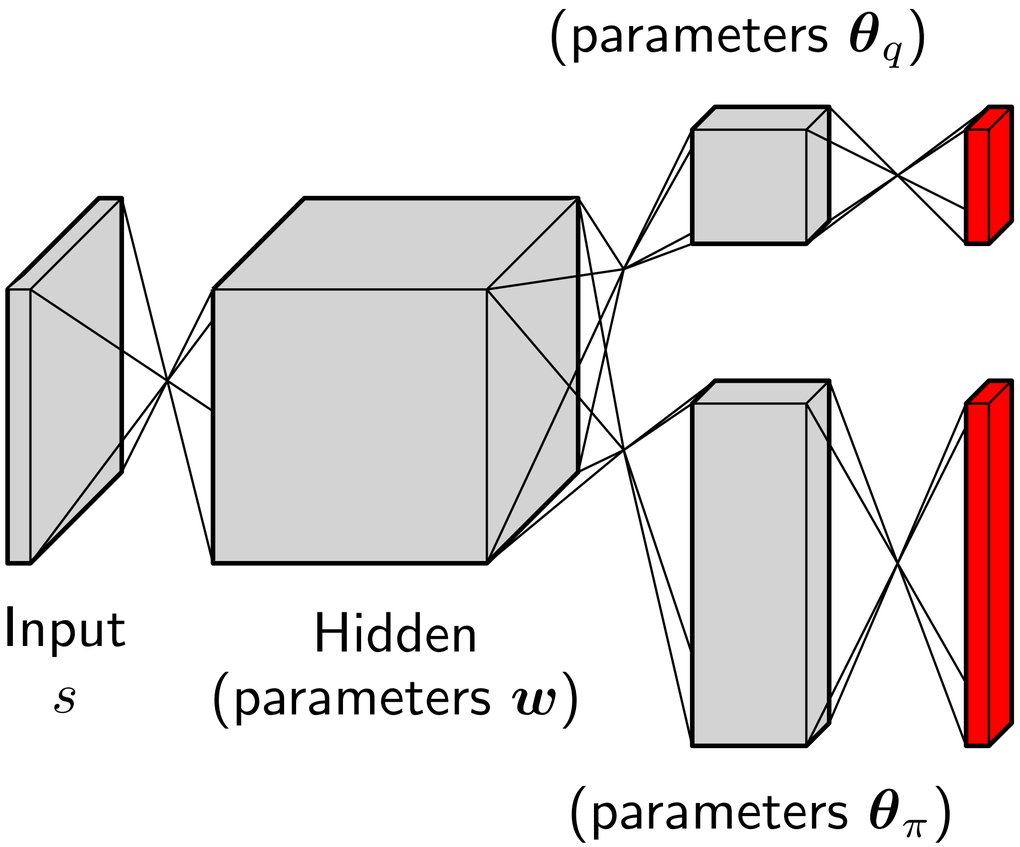
What will this presentation be all about?

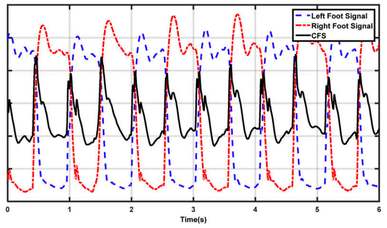
- What is RL?
- How to solve RL problems - the shallow view
- How to solve RL problems - the deep view
- **Fun stuff**
- Shallow waters…

Actor-critic controlling a humanoid



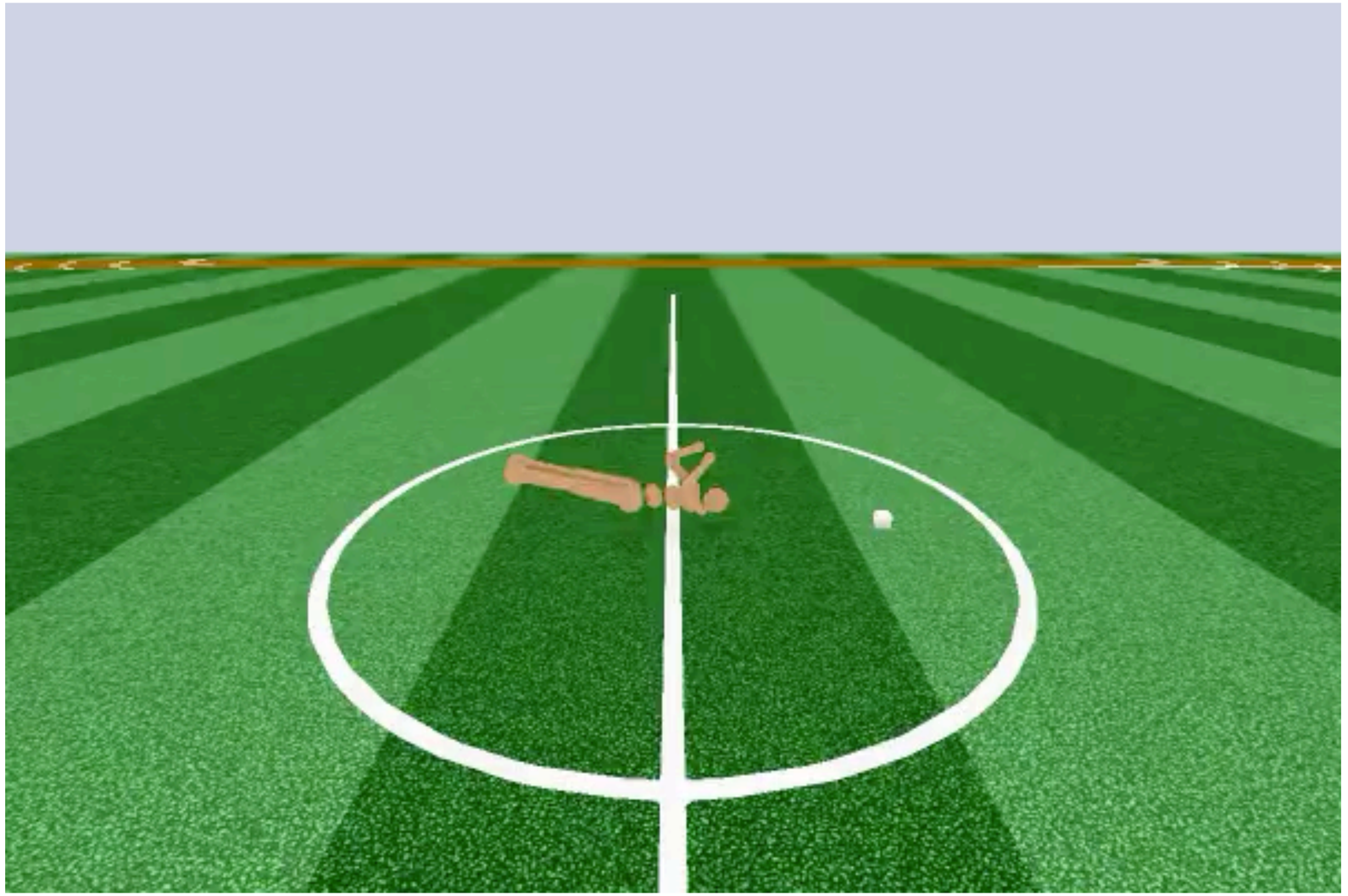
Joint angles, velocities



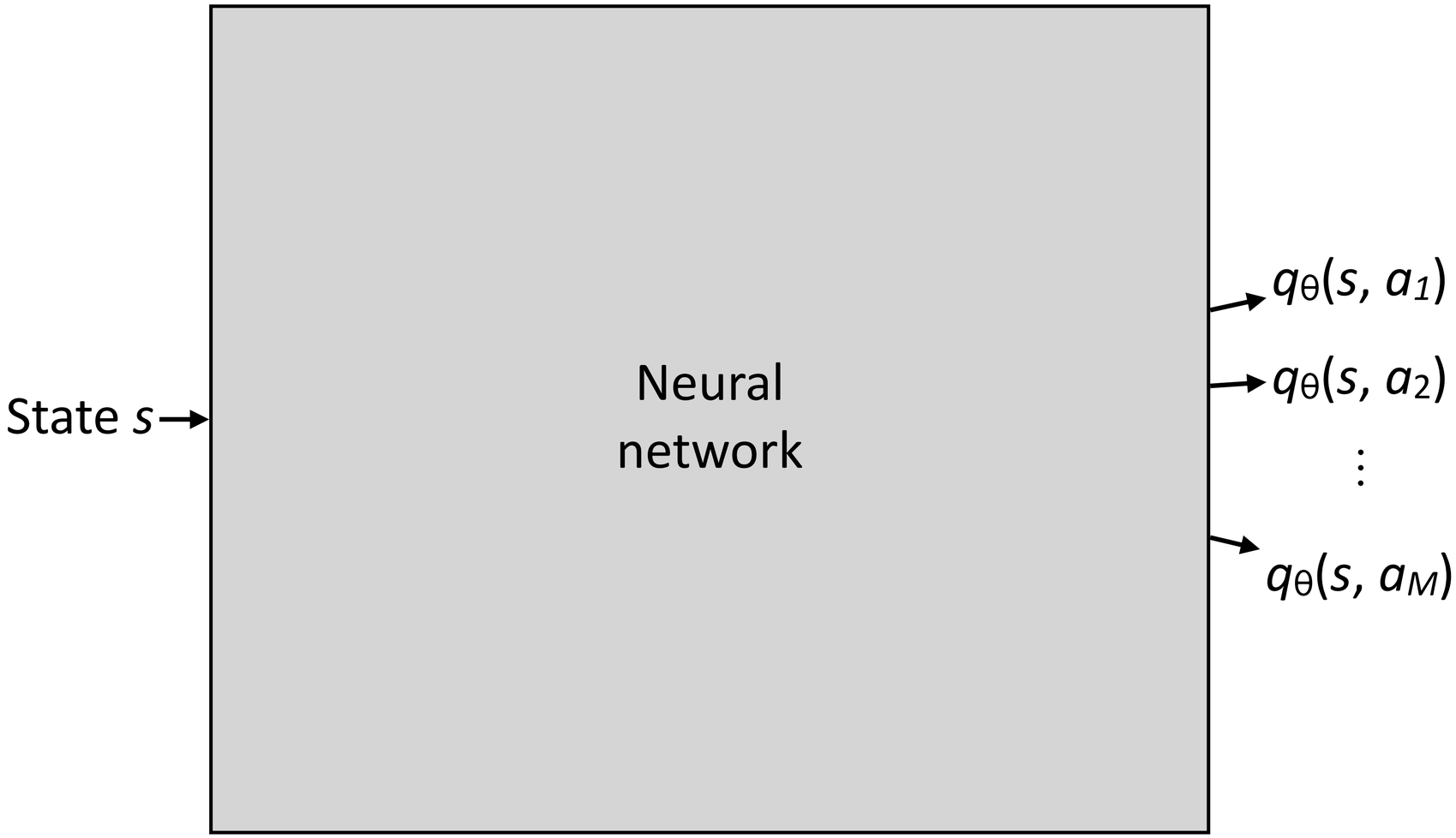


Actuator signals

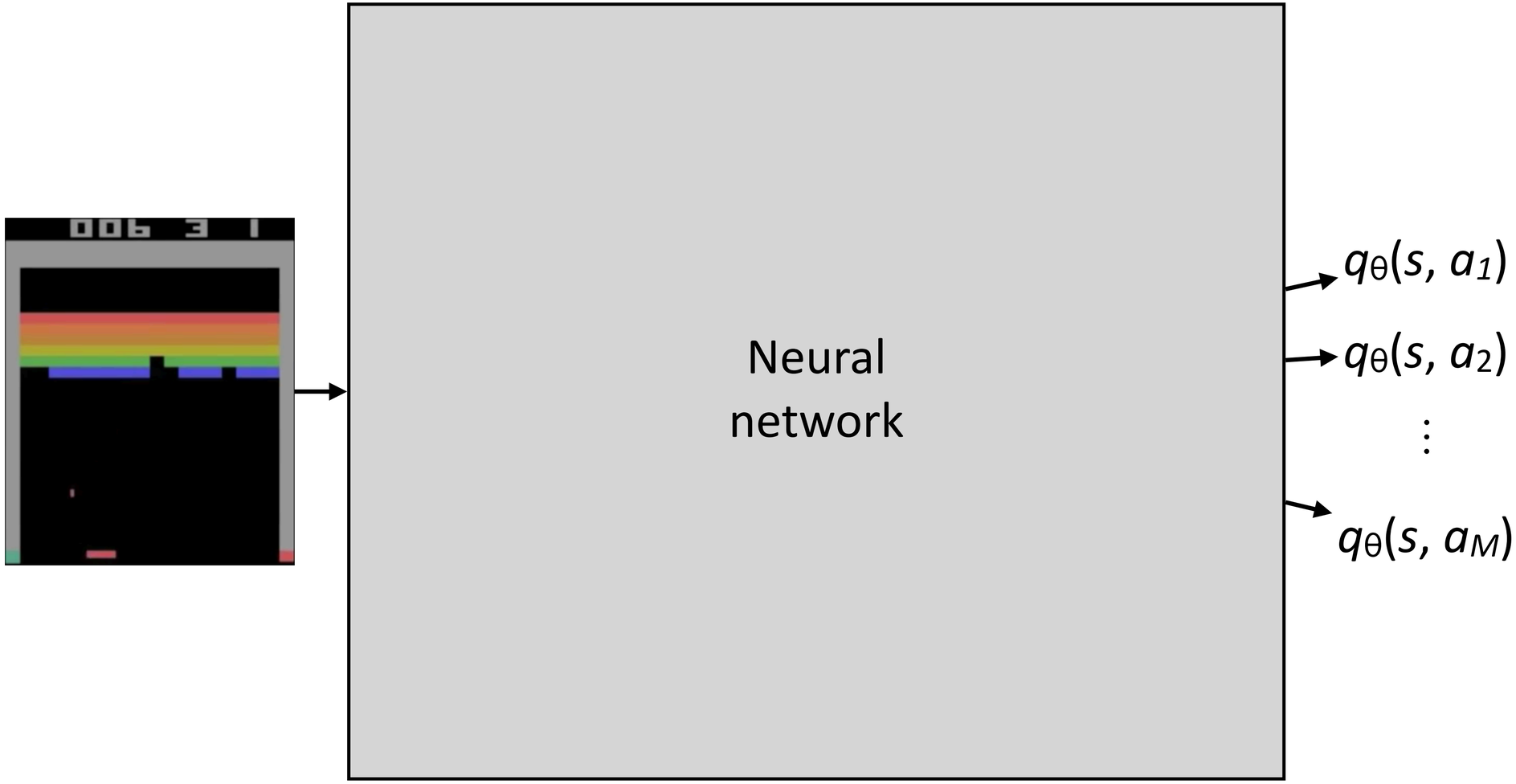
Actor-critic controlling a humanoid



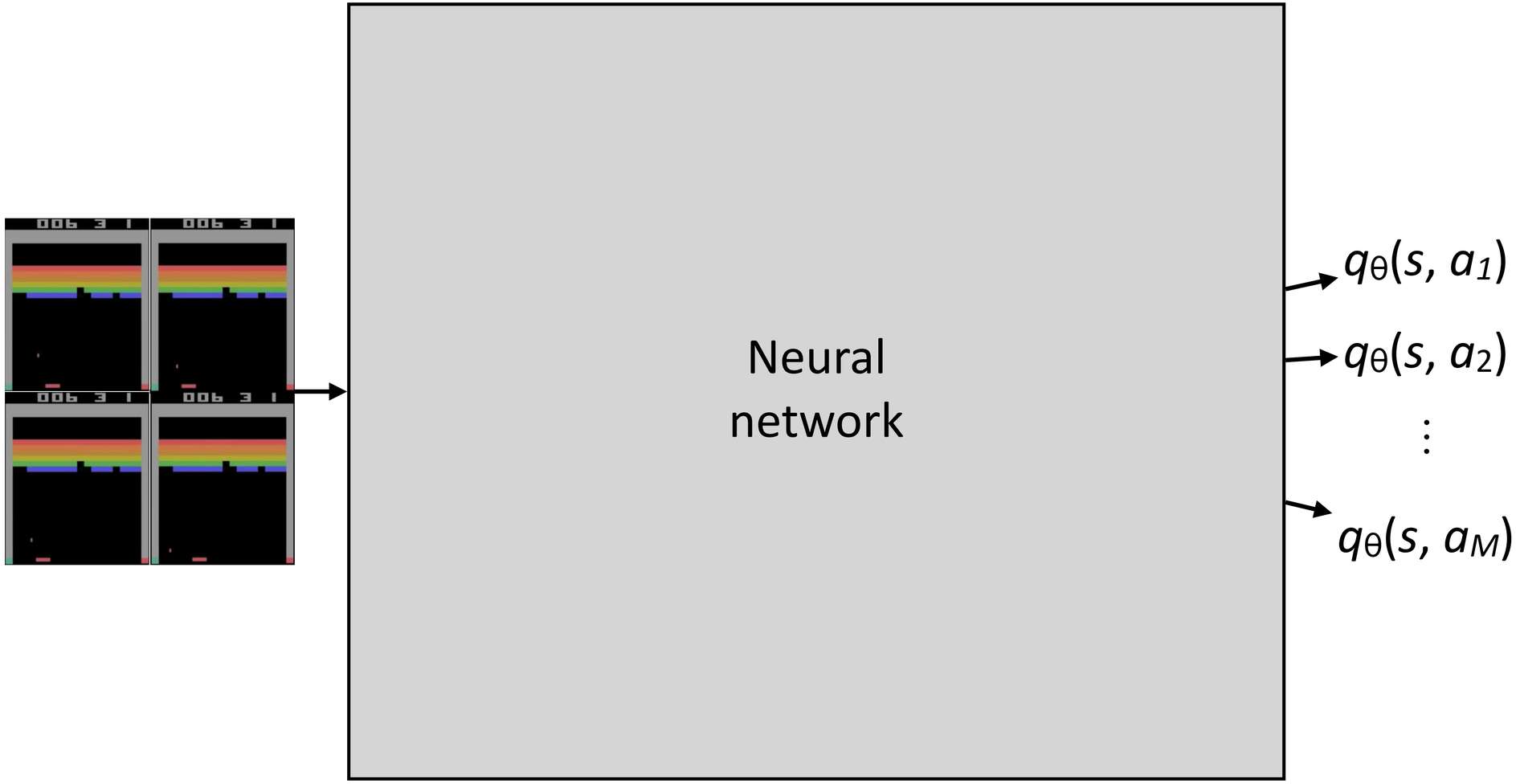
DQN playing Atari



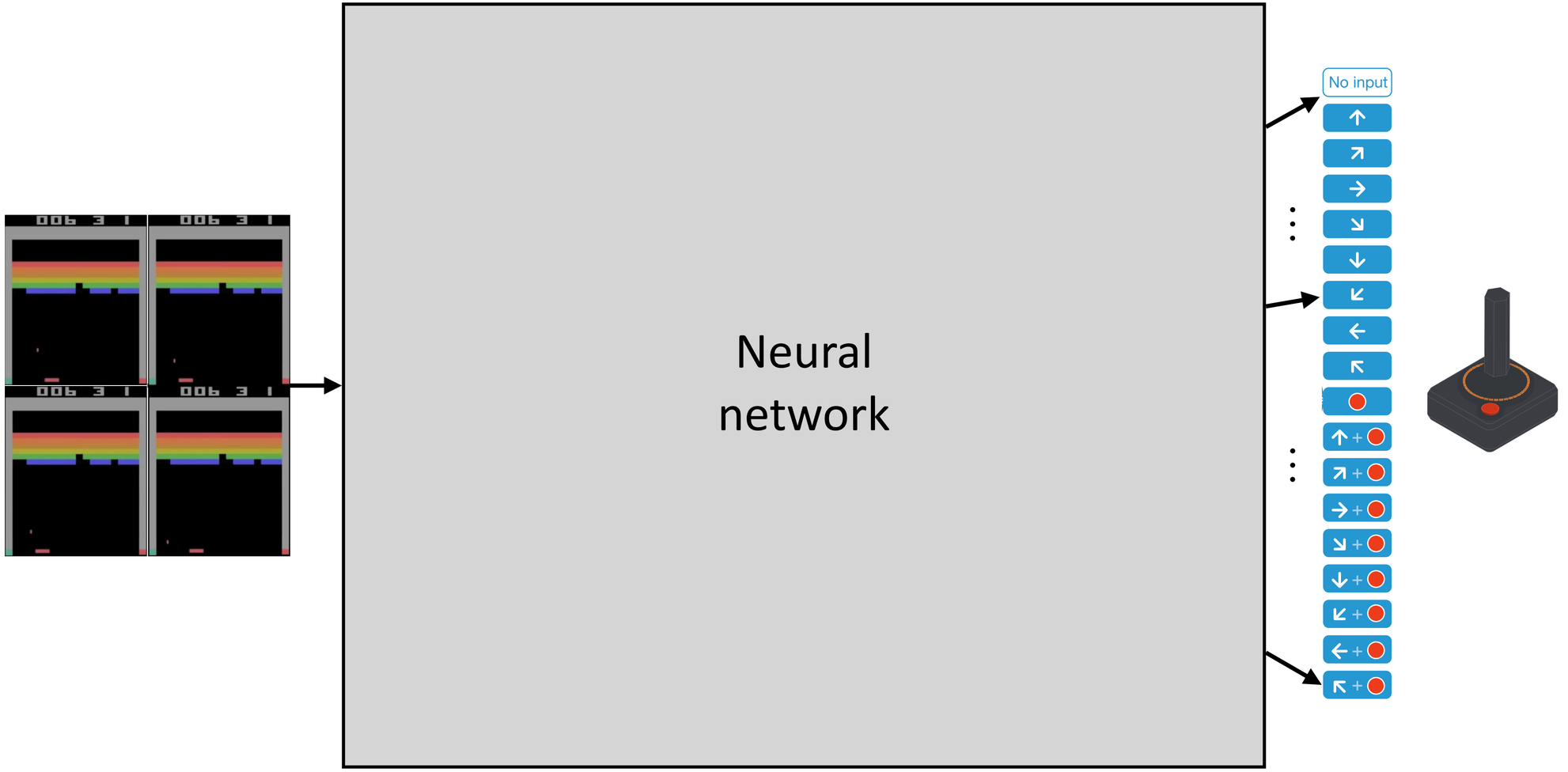
DQN playing Atari



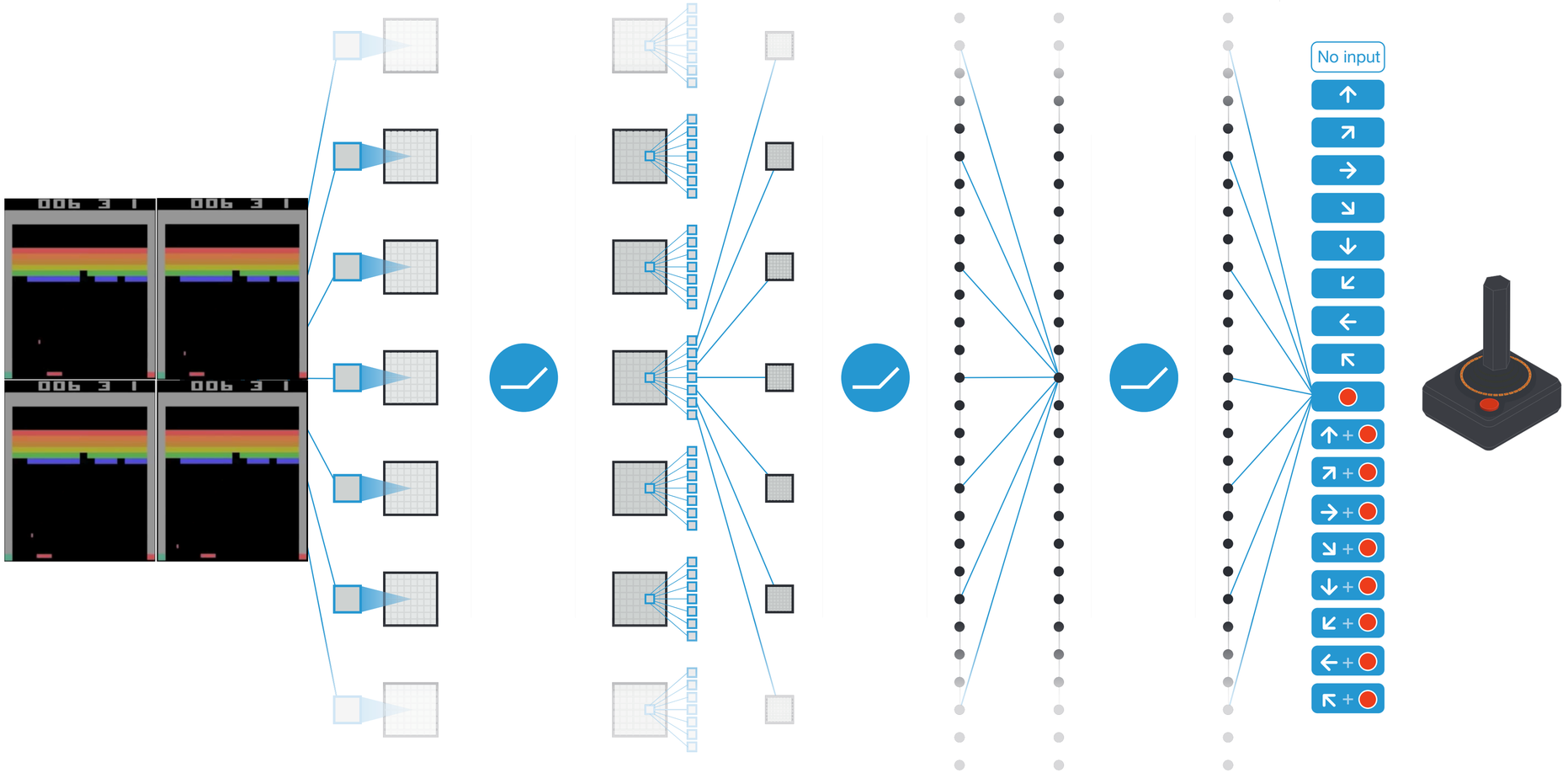
DQN playing Atari



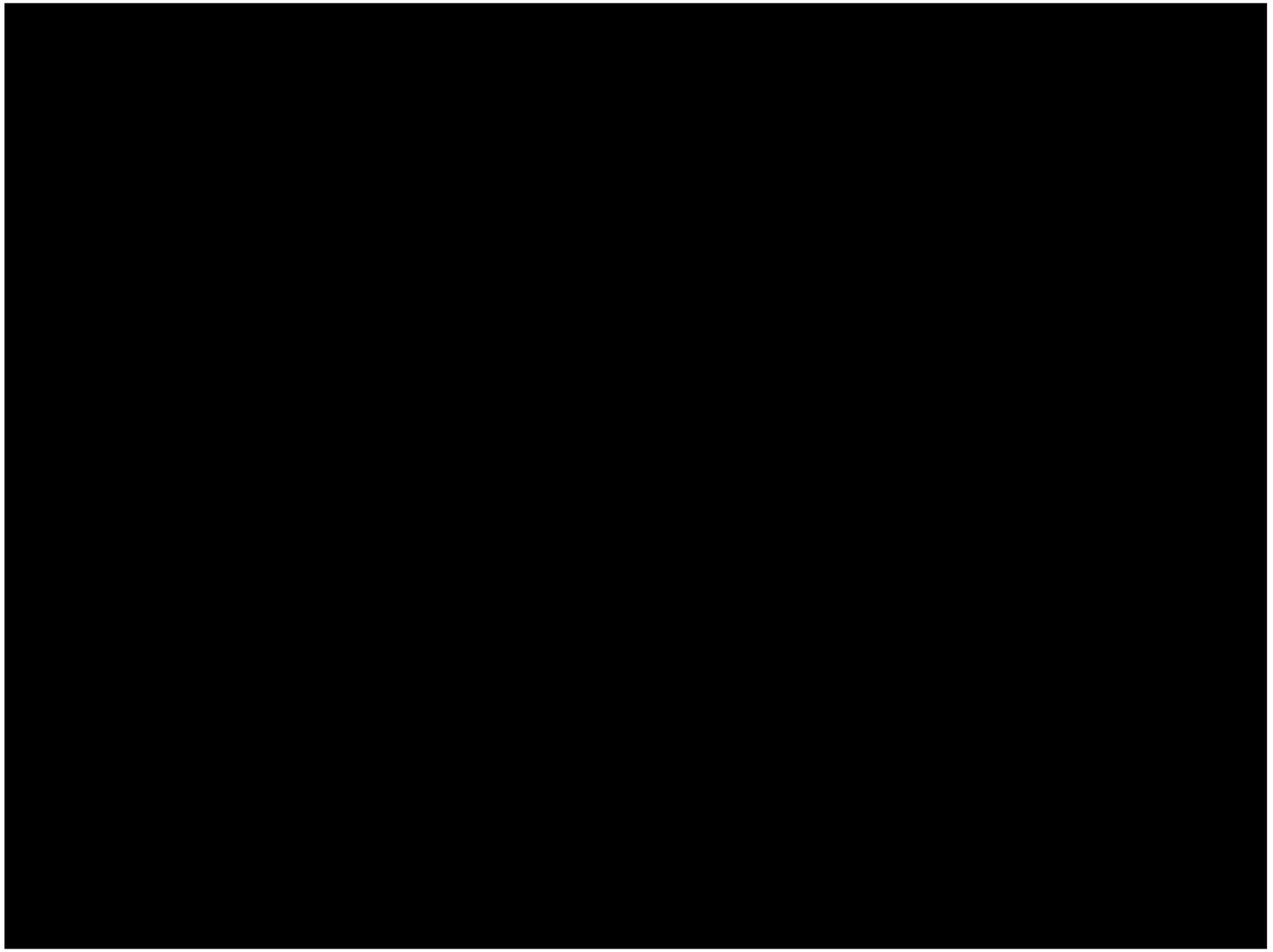
DQN playing Atari



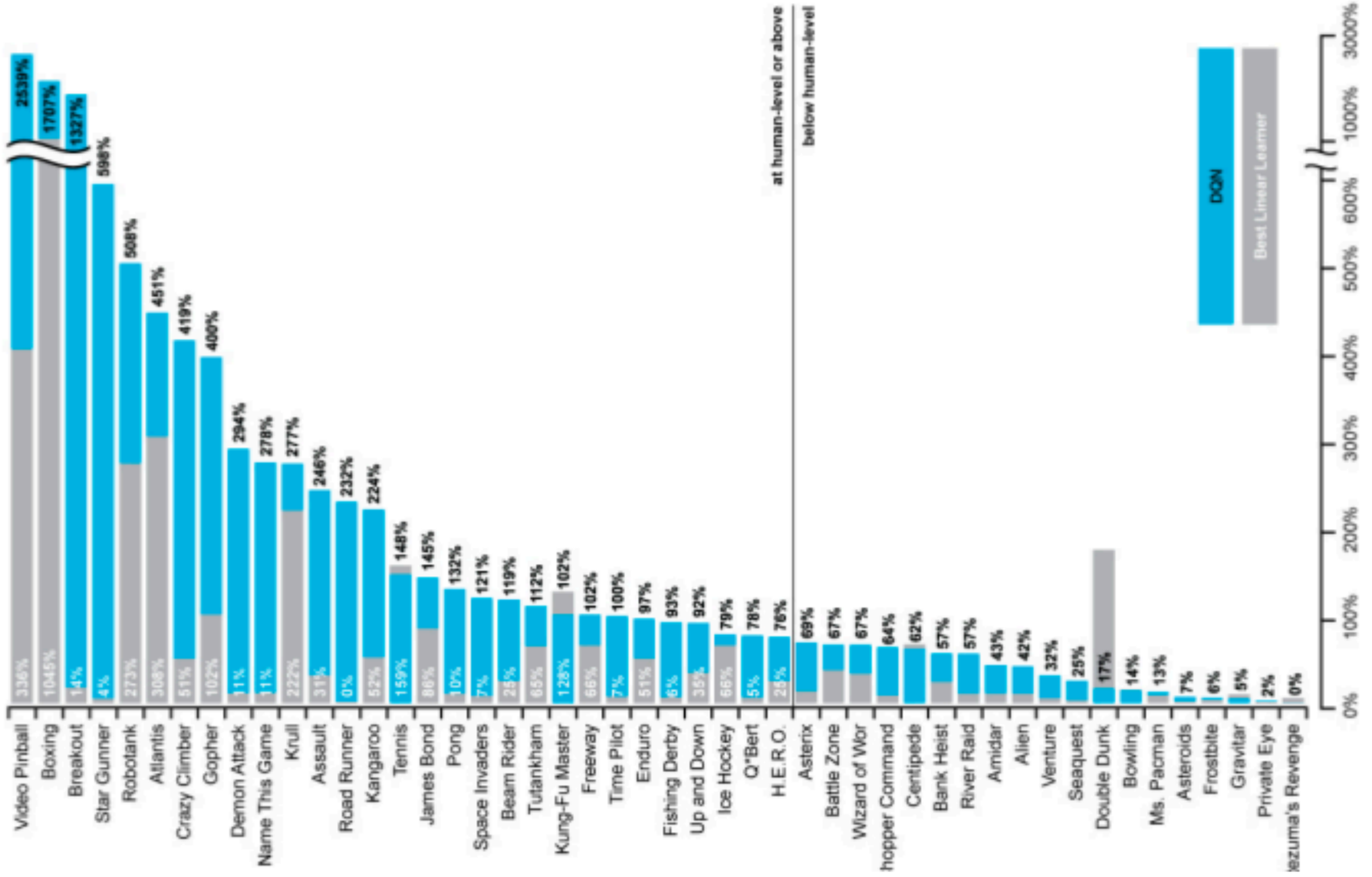
DQN playing Atari



DQN playing Atari

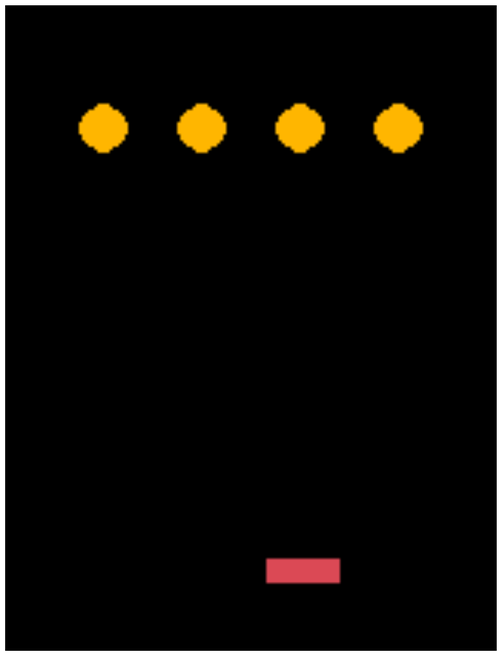


DQN playing Atari



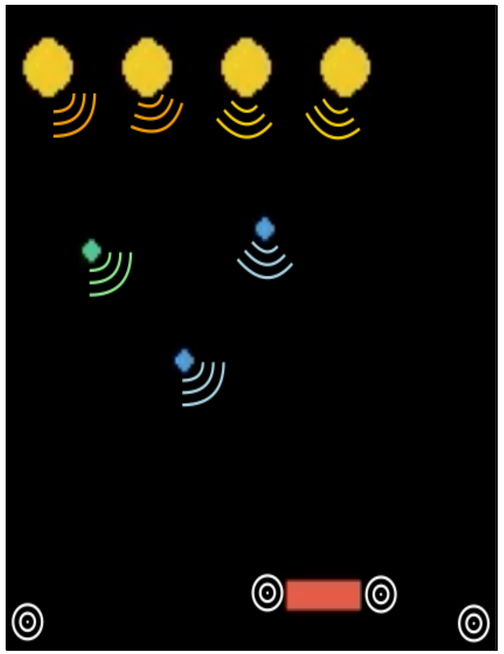
Playing a game using only sound

- Simplified version of Space Invaders game



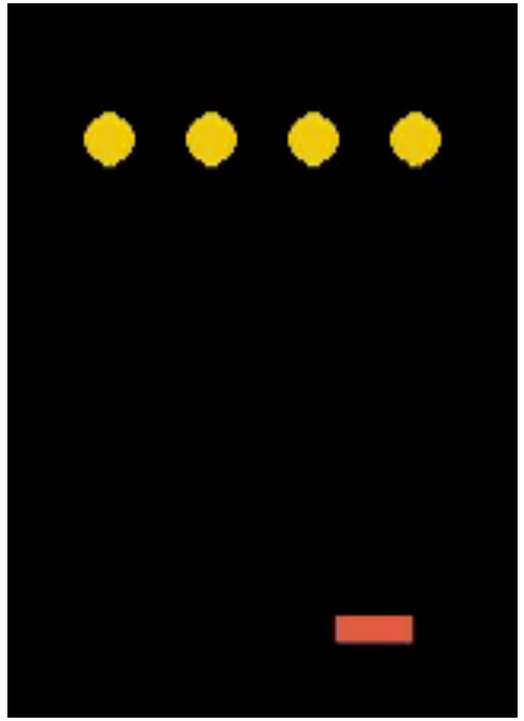
Playing a game using sound

- Simplified version of Space Invaders game
- Bullets and enemies emit sound
- There are microphones at the bottom and on the pad
- Learned a joint model of sound and image



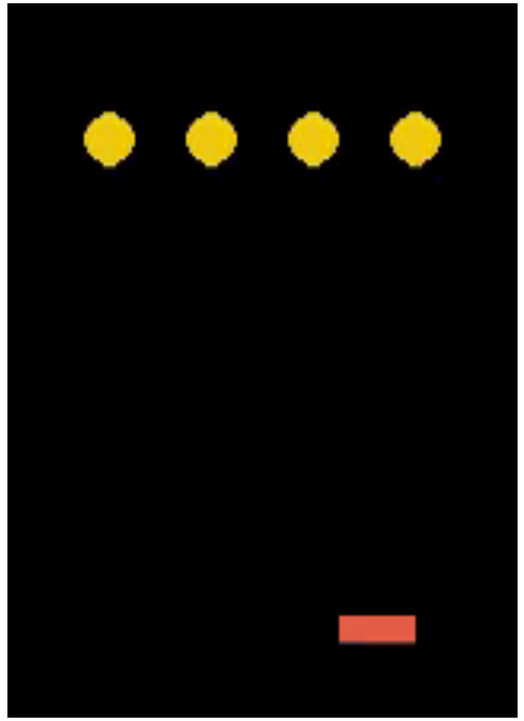
Playing a game using sound

- Policy trained using (encoded) images
- Tested with (encoded) images



Playing a game using sound

- Policy trained using (encoded) images
- Tested with only (encoded) sound

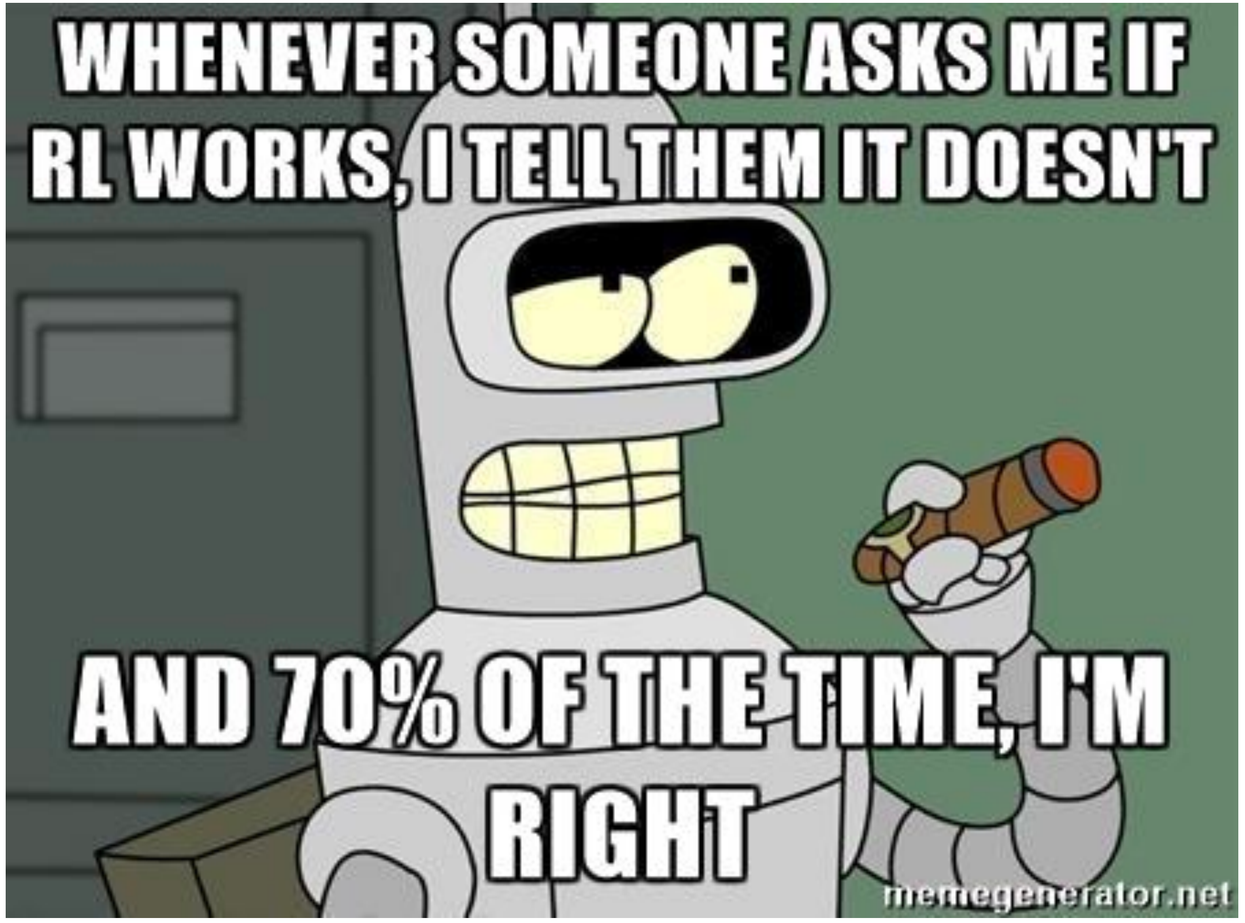


Outline

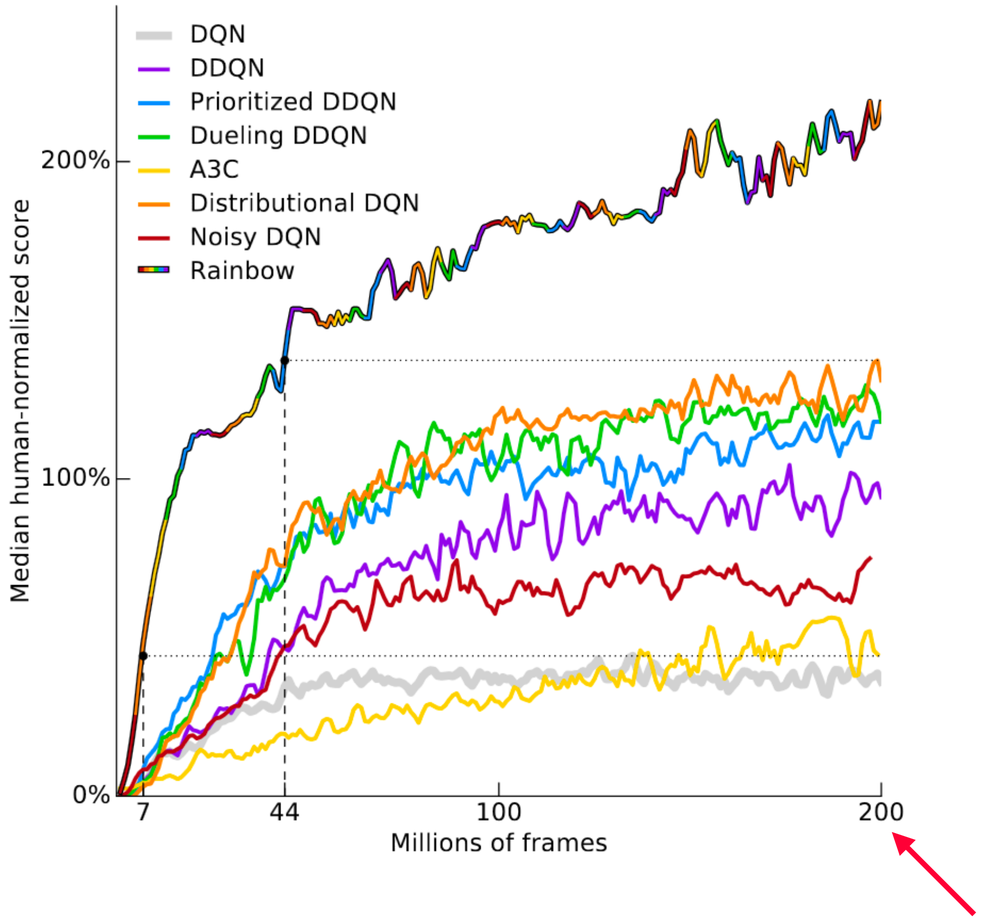
What will this presentation be all about?

- What is RL?
- How to solve RL problems - the shallow view
- How to solve RL problems - the deep view
- Fun stuff
- **Shallow waters…**

Has Deep RL solved AI?

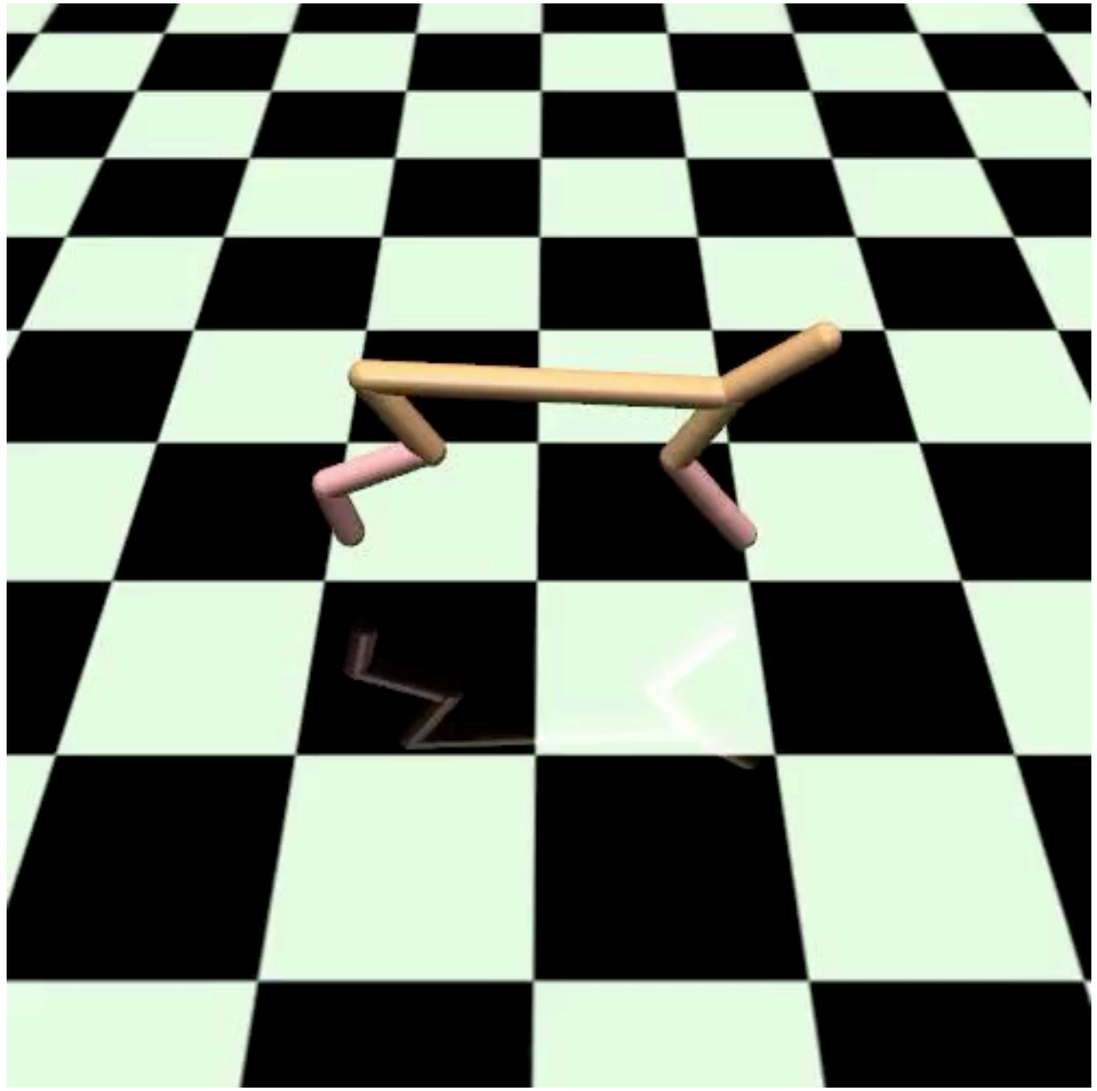


Sample (in)efficiency

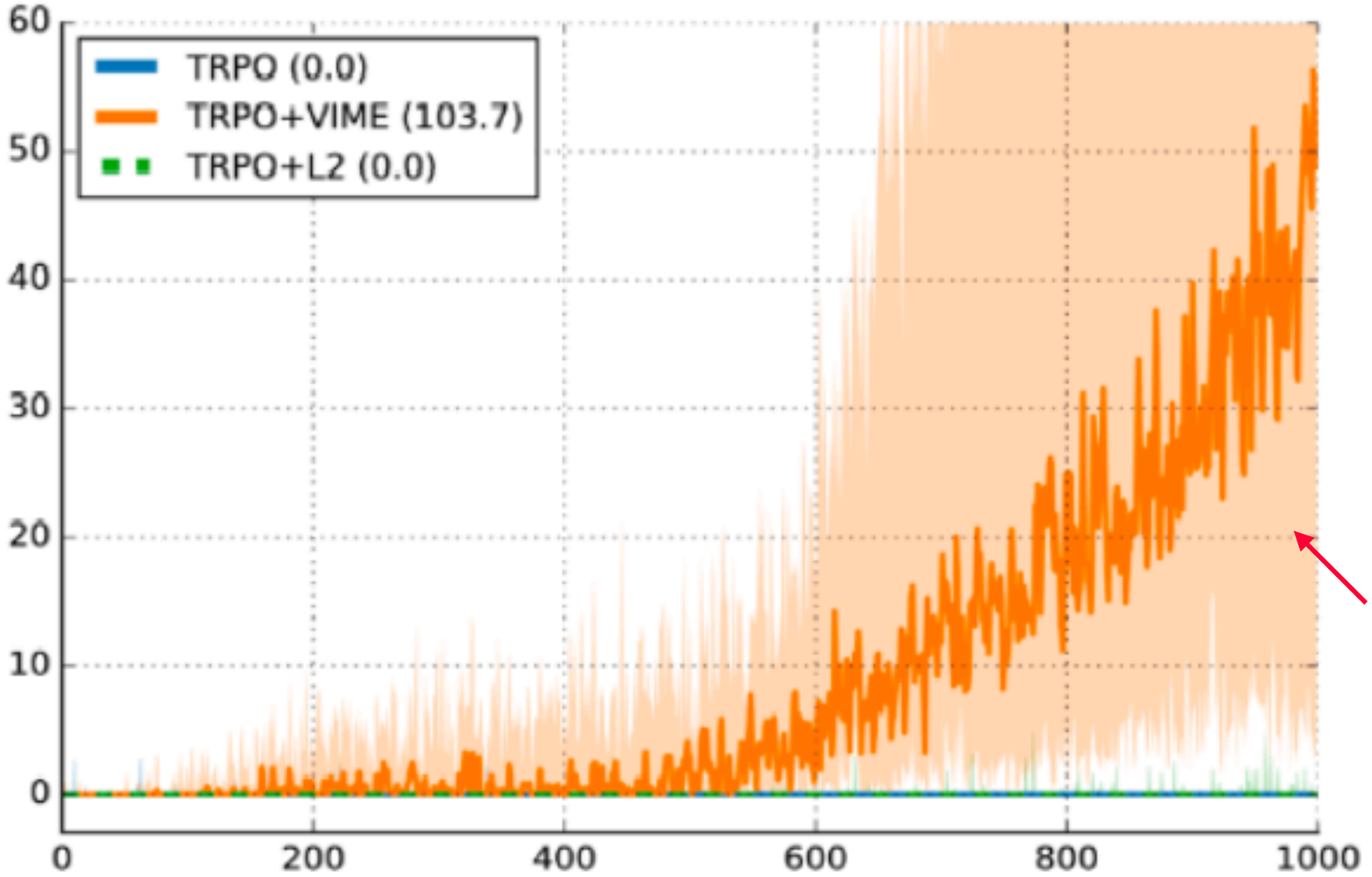


Where do rewards come from?

What about local minima?



Stability and reproducibility



Wrap-up

Reinforcement learning

- Reinforcement learning allows an agent/algorithm to learn how to solve sequential problems by trial and error
- Reinforcement learning problems are hard
- Deep RL combines ideas from deep learning with classical ideas from RL
  - Tremendous successes
- There is still a lot to be learned
- There is still a lot to be done

References

- Hasselt, H., Guez, A., and Silver, D. “Deep reinforcement learning with double Q-Learning.” In **Proc. 30th AAAI Conf. Artificial Intelligence**, pp. 2094-2100, 2016. **(DDQN)**
- Irpan, A. “Deep reinforcement learning doesn’t work yet”. Online at: https://www.alexirpan.com/2018/02/14/rl-hard.html (last visited, May 5, 2021)
- Mnih, V., Badia, A., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., and Kavukcuoglu, K. “Asynchronous methods for deep reinforcement learning.” In **Proc. 33rd Int. Conf. Machine Learning**, pp. 1928-1937, 2016. **(A3C)**
- Mnih, V., et al. “Human-level control through deep reinforcement learning.” **Nature**, 518:529-533, 2015. **(DQN)**
- Schaul, T., Quan, J., Antonoglou, I., and Silver., D. “Prioritized experience replay.” arXiv preprint arXiv:1511.05952, 2015. **(Prioritized experience replay)**
- Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. “Trust Region Policy Optimization.” In **Proc. 32nd Int. Conf. Machine Learning**, pp. 1889-1897, 2015. **(TRPO)**
- Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. “Proximal Policy Optimization Algorithms.” arXiv preprint arXiv:1707:06347, 2017. **(PPO)**
- Silva, R., Vasco, M., Melo, F., Paiva, A., and Veloso, M. “Playing games in the dark: An approach for cross-modality transfer in reinforcement learning.” In Proc. 19th Int. Conf**. Autonomous Agents and Multiagent Systems**, pp. 1260-1268, 2020.
- Sutton, R., and Barto, A. **Reinforcement Learning: An Introduction**, 2nd Edition, MIT Press, 2018.
- Wang, Z., Schaul, T., Hessel, M., Hasselt, H., Lanctot, M., and Freitas, N. “Dueling network architectures for deep reinforcement learning.” In **Proc. 33rd Int. Conf. Machine Learning**, pp. 1995-2003, 2016. **(Dueling architecture)**