StarAi: Deep Reinforcement Learning

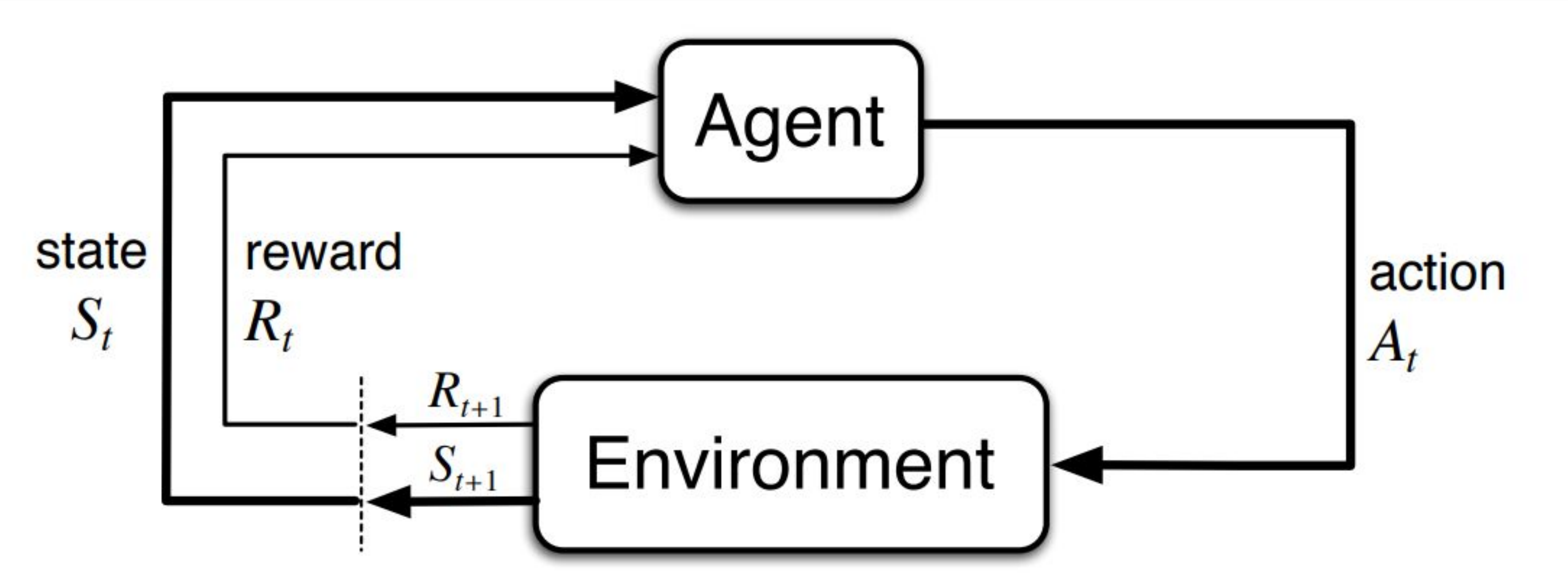
Tabular Q Learning

William Xu

Outline

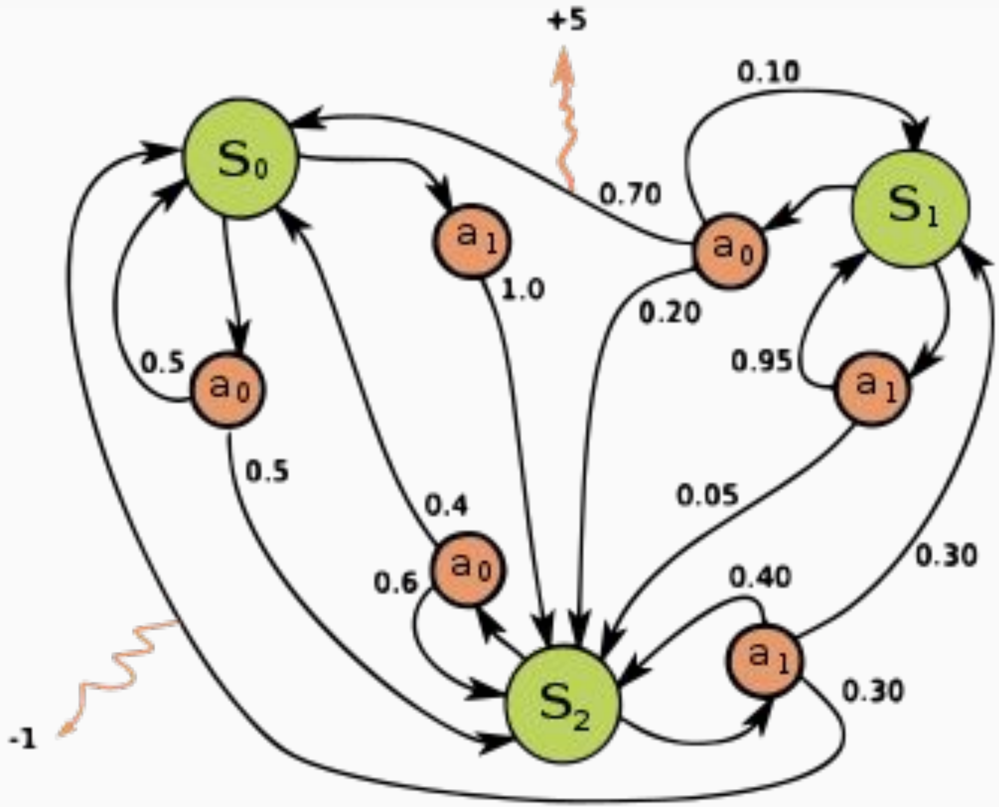
- Reflect on week 1 & 2
- Defining the problem
- Intuition for Tabular Q
- Simplify the problem and solution, do a walk through
- Exercise
- Dealing with continuous state spaces
- Homework
- Key takeaways and next week

The Reinforcement Learning Problem

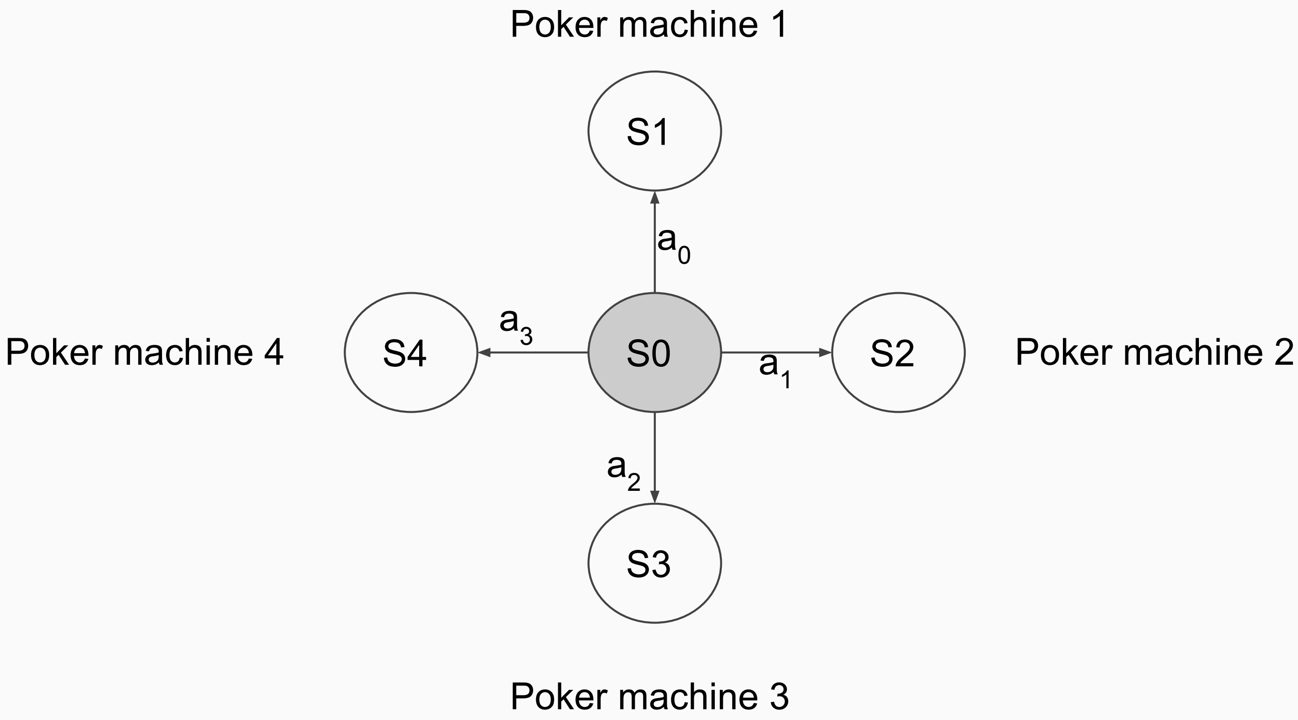


Markov Decision Process

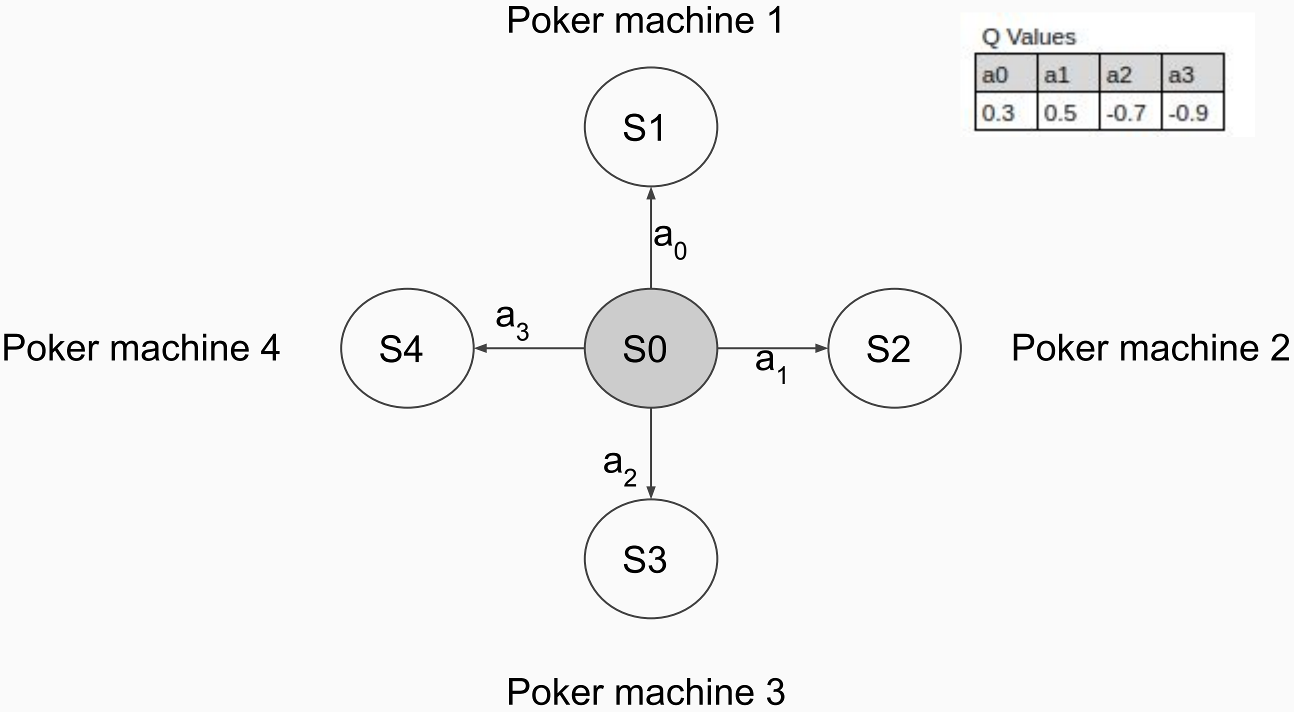
$$(S, A, P_a, R_a, \gamma)$$



Multi Armed Bandit - MDP perspective



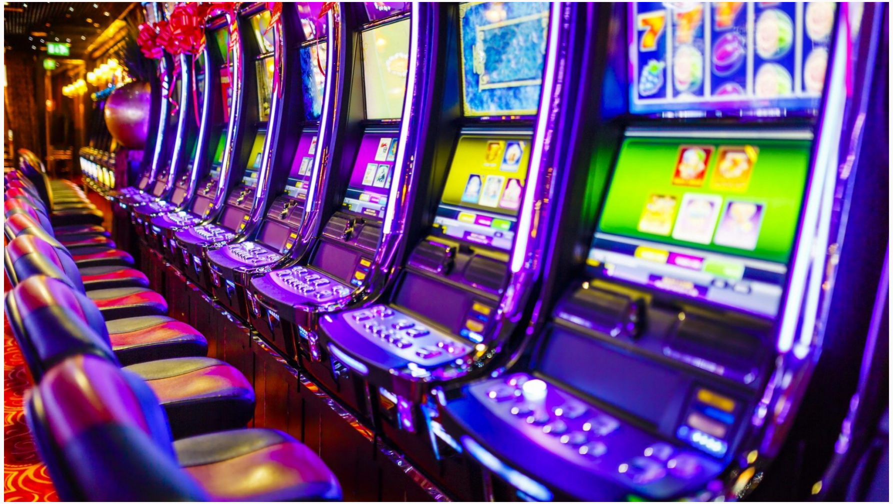
Multi Armed Bandit



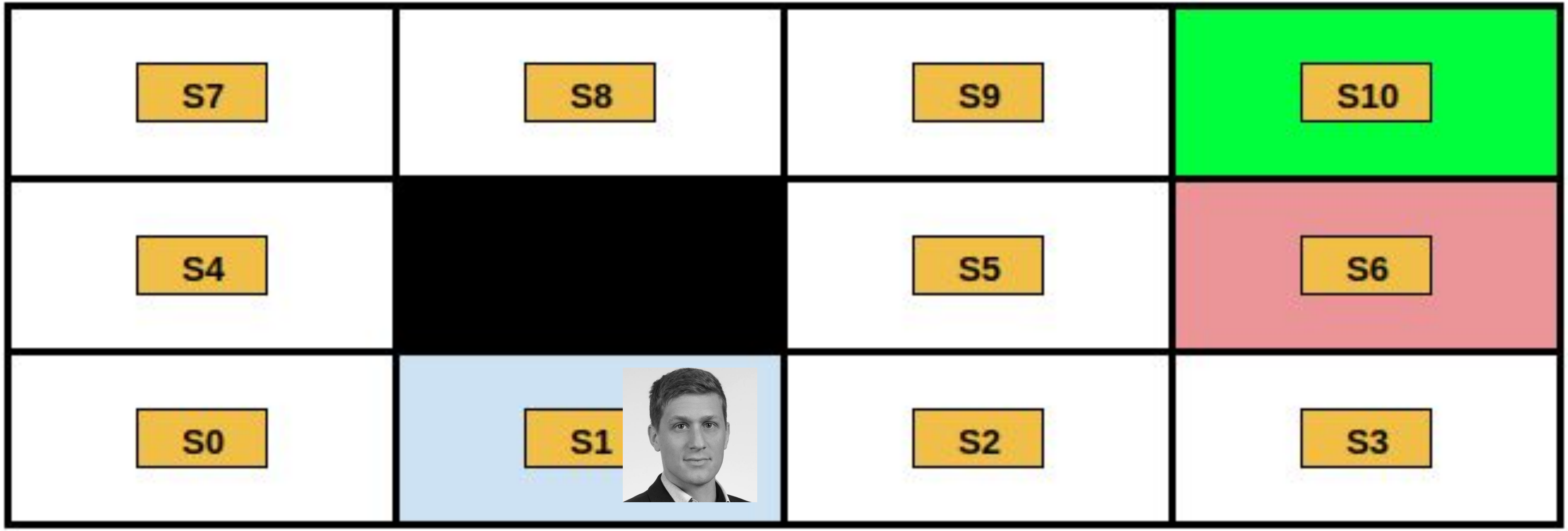
Q Values

a0	a1	a2	a3
0.3	0.5	-0.7	-0.9

Poker machine FWT - To the Casino



Paul at the Casino



Tabular Q Learning

- Learning Q(s, a)
- Temporal Difference Learning - TD(0)
  - Temporal definition: relating to time

Q Learning

- Learning Q(s, a)
- Temporal Difference Learning - TD(0)
    - Temporal definition: relating to time
- Bellman optimality equation for Q_*

$$q_*(s,a) \quad = \quad \mathbb{E}\Big[R_{t+1} + \gamma \max_{a'} q_*(S_{t+1}, a') \;\Big|\; S_t = s, A_t = a\Big]$$

Tabular Q

Q-learning (off-policy TD control) for estimating $\pi \approx \pi_*$

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(\mathit{terminal}, \cdot) = 0$

Loop for each episode:
    Initialize S
    Loop for each step of episode:
        Choose A from S using policy derived from Q (e.g., ε-greedy)
        Take action A, observe R, S'
        $Q(S, A) \leftarrow Q(S, A) + \alpha\big[R + \gamma \max_a Q(S', a) - Q(S, A)\big]$
        $S \leftarrow S'$
    until S is terminal

A Simplified Tabular Q

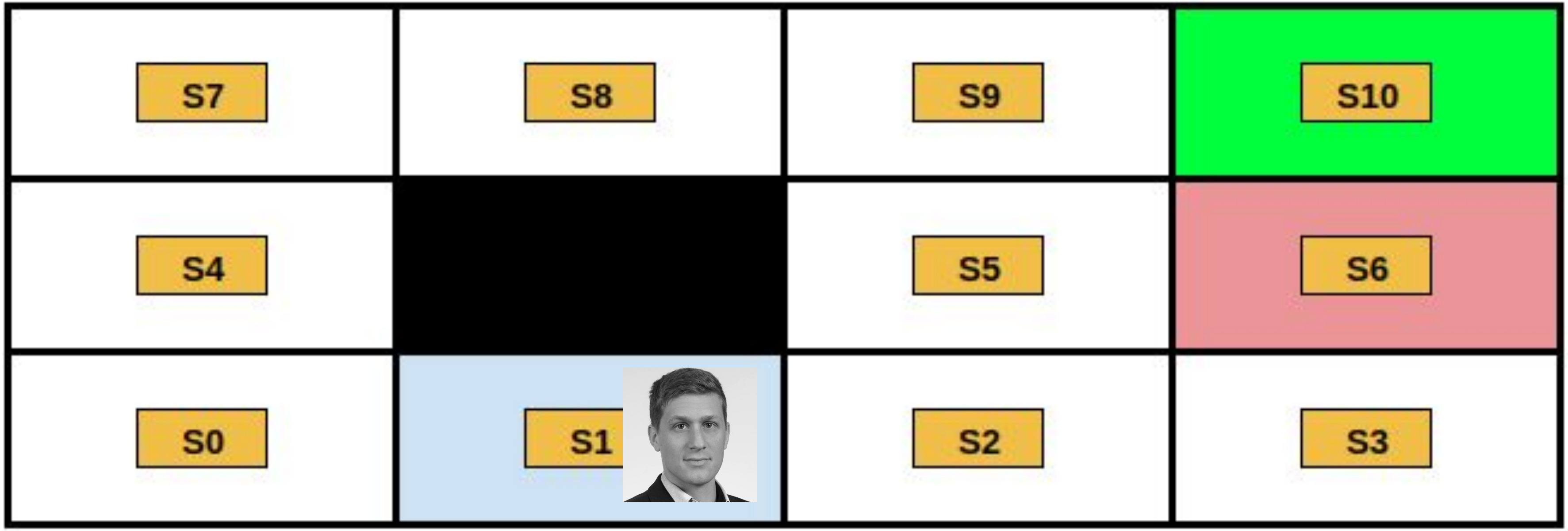
$$Q(S, A) \leftarrow Q(S, A) + \alpha \left[R + \gamma \max_a Q(S', a) - Q(S, A)\right]$$

A Simplified Tabular Q

Set alpha = 1

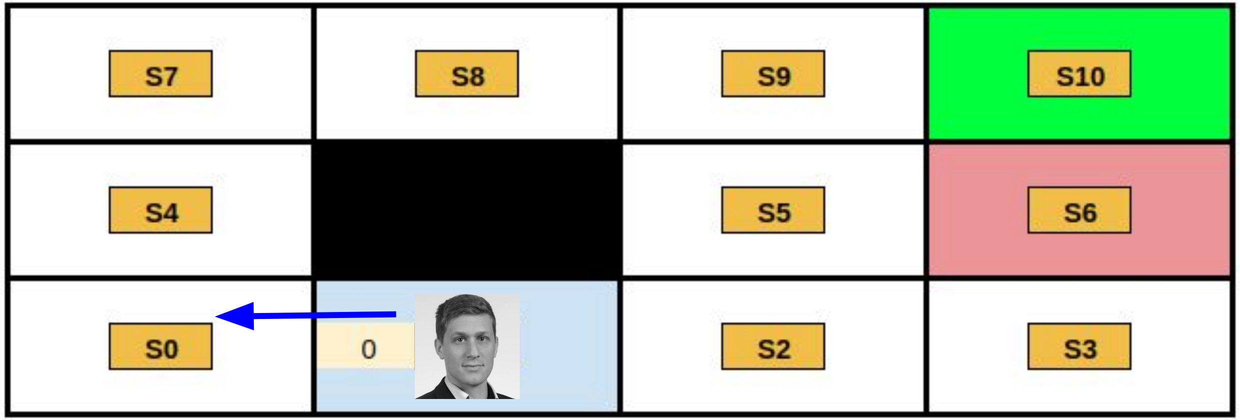
$$Q(S,A) \leftarrow \cancel{Q(S,A)} + \cancel{\alpha}\left[R + \gamma \max_a Q(S',a) - \cancel{Q(S,A)}\right]$$

A Simplified walkthrough



$$Q(S, A) \leftarrow R + \gamma \max_a Q(S', a)$$

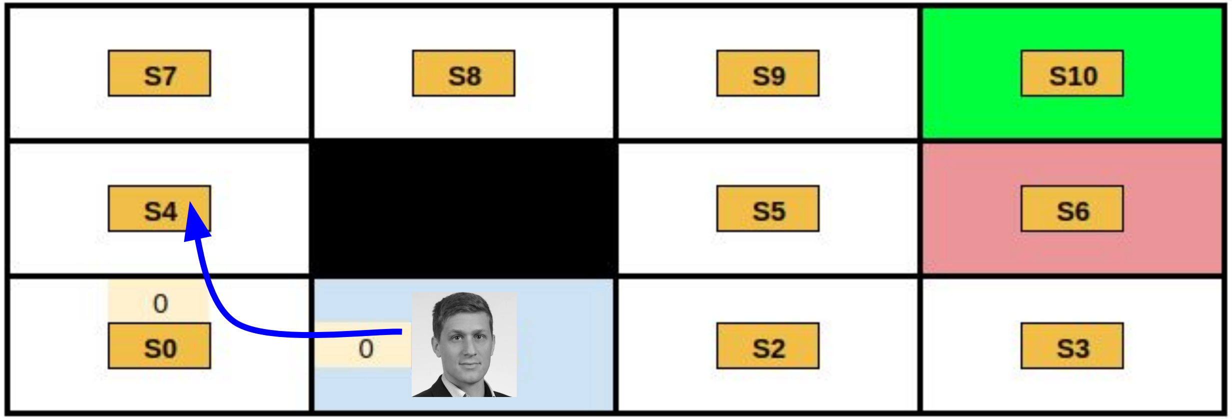
A Simplified walkthrough - Attempt 1



State	Left	Right	Up	Down
S1	0			

$$Q(S, A) \leftarrow R + \gamma \max_a Q(S', a)$$

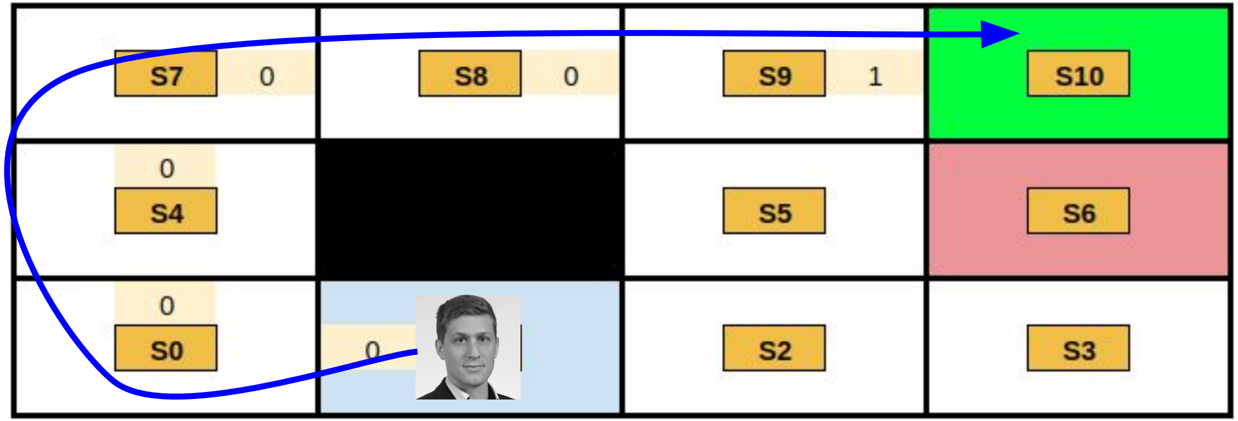
A Simplified walkthrough - Attempt 1



State	Left	Right	Up	Down
S0			0	
S1	0			

$$Q(S, A) \leftarrow R + \gamma \max_a Q(S', a)$$

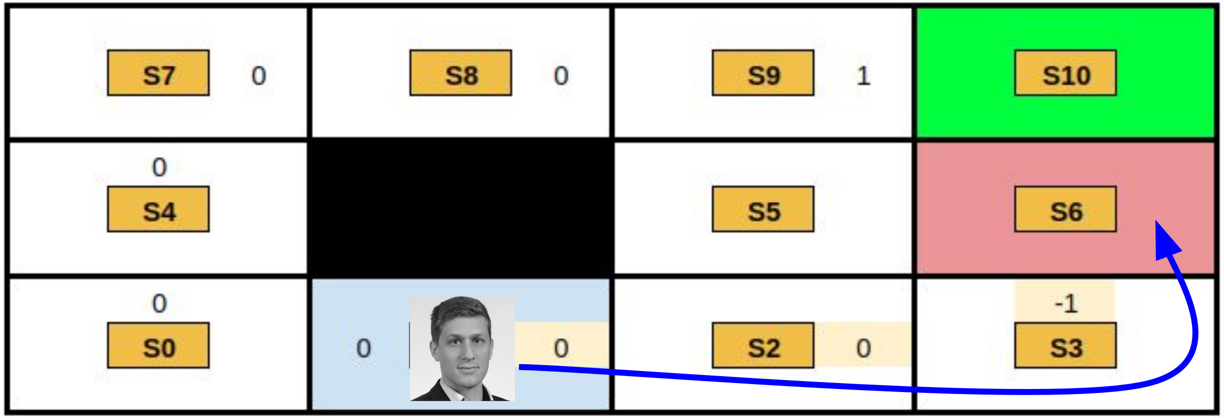
A Simplified walkthrough - Attempt 1



State	Left	Right	Up	Down
S0			0	
S1	0			
S4			0	
S7		0		
S8		0		
S9		1		
S10				

$$Q(S, A) \leftarrow R + \gamma \max_a Q(S', a)$$

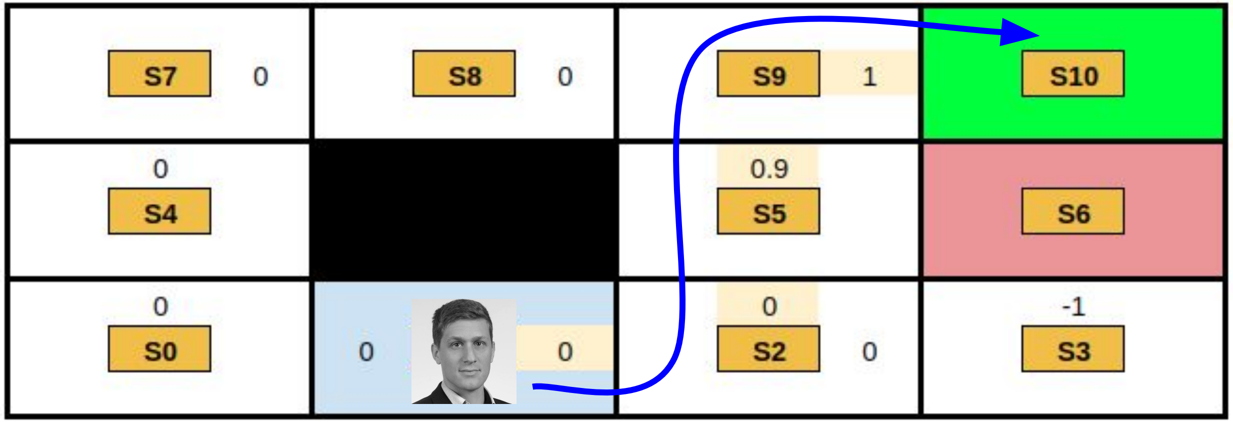
A Simplified walkthrough - Attempt 2



State	Left	Right	Up	Down
S0			0	
S1	0	0		
S2		0		
S3			-1	
S4			0	
S7		0		
S8		0		
S9		1		

$$Q(S, A) \leftarrow R + \gamma \max_a Q(S', a)$$

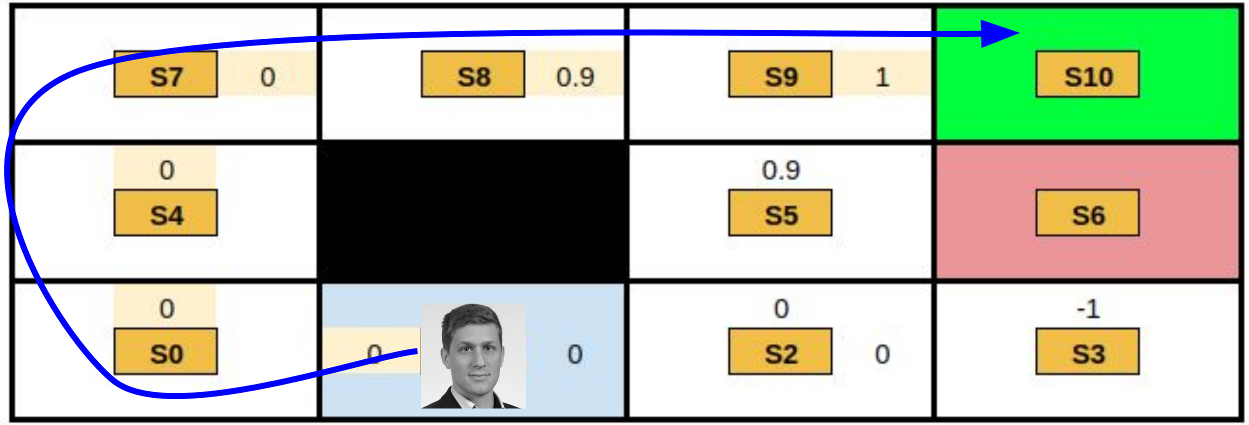
A Simplified walkthrough - Attempt 3



State	Left	Right	Up	Down
S0			0	
S1	0	0		
S2		0	0	
S3			-1	
S4			0	
S5			0.9	
S7		0		
S8		0		
S9		1		

$$Q(S, A) \leftarrow R + \gamma \max_a Q(S', a)$$

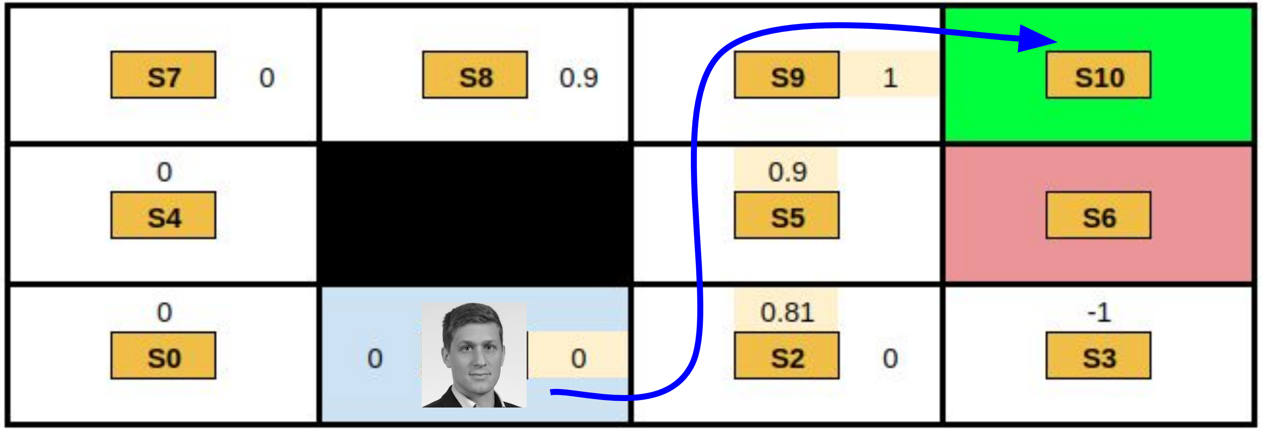
A Simplified walkthrough - Attempt 4



State	Left	Right	Up	Down
S0			0	
S1	0	0		
S2		0	0	
S3			-1	
S4			0	
S5			0.9	
S7		0		
S8		0.9		
S9		1		

$$Q(S, A) \leftarrow R + \gamma \max_a Q(S', a)$$

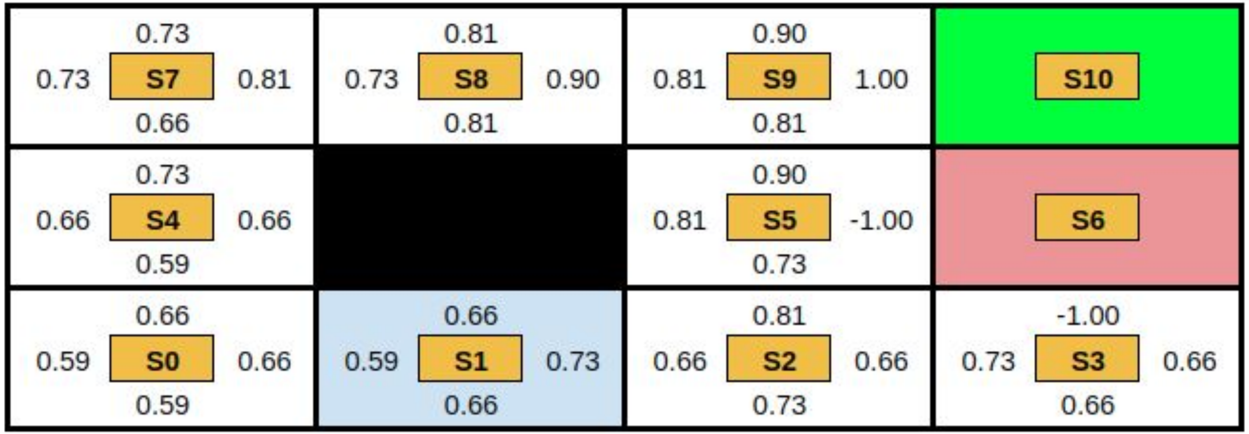
A Simplified walkthrough - Attempt 5



State	Left	Right	Up	Down
S0			0	
S1	0	0		
S2		0	0.81	
S3			-1	
S4			0	
S5			0.9	
S7		0		
S8		0.9		
S9		1		

$$Q(S, A) \leftarrow R + \gamma \max_a Q(S', a)$$

A Simplified walkthrough - After enough attempts



State	Left	Right	Up	Down
S0	0.59	0.66	0.66	0.59
S1	0.59	0.73	0.66	0.66
S2	0.66	0.66	0.81	0.73
S3	0.73	0.66	-1	0.66
S4	0.66	0.66	0.73	0.59
S5	0.81	-1	0.9	0.73
S7	0.73	0.81	0.73	0.66
S8	0.73	0.9	0.81	0.81
S9	0.81	1	0.9	0.81

$$Q(S, A) \leftarrow R + \gamma \max_a Q(S', a)$$

The real algorithm for stochastic scenarios

$$Q(S_t, A_t) \leftarrow R_{t+1} + \gamma \max_a Q(S_{t+1}, a)$$

When the left and right doesn't match

$$Error = [R_{t+1} + \gamma \max_a Q(S_{t+1}, a)] - Q(S_t, A_t)$$

An enhanced learning process

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha[Error]$$

The final formula

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha[R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t)]$$

Tabular Q

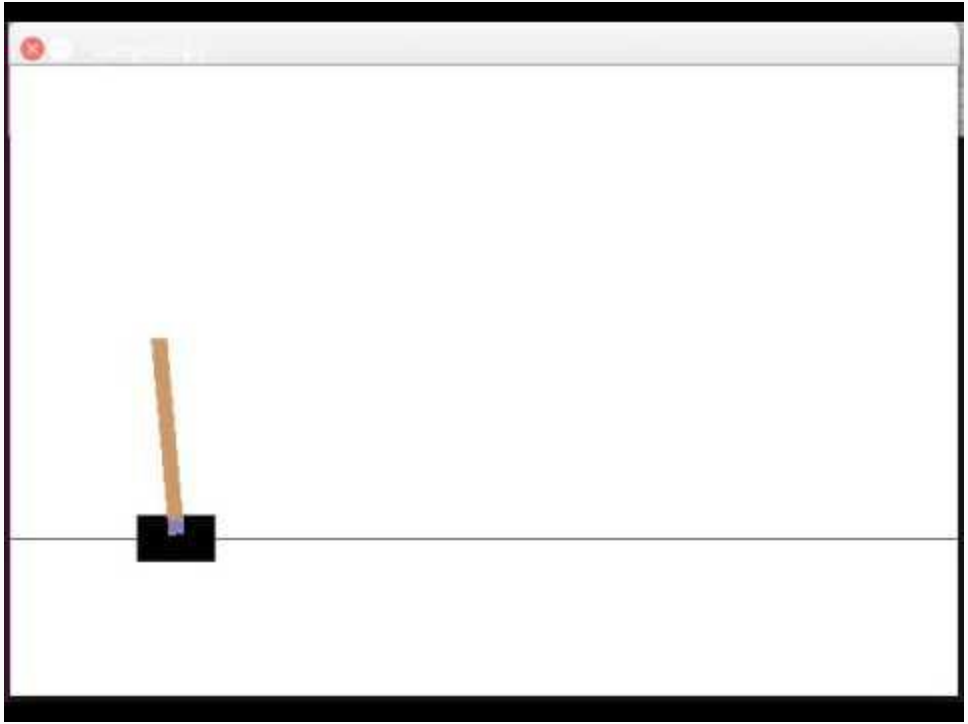
Q-learning (off-policy TD control) for estimating $\pi \approx \pi_*$

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(\mathit{terminal}, \cdot) = 0$

Loop for each episode:
  Initialize S
  Loop for each step of episode:
    Choose A from S using policy derived from Q (e.g., ε-greedy)
    Take action A, observe R, S'
    $Q(S, A) \leftarrow Q(S, A) + \alpha\big[R + \gamma \max_a Q(S', a) - Q(S, A)\big]$
    $S \leftarrow S'$
  until S is terminal

Exercise - Frozen Lake

Dealing with continuous state spaces



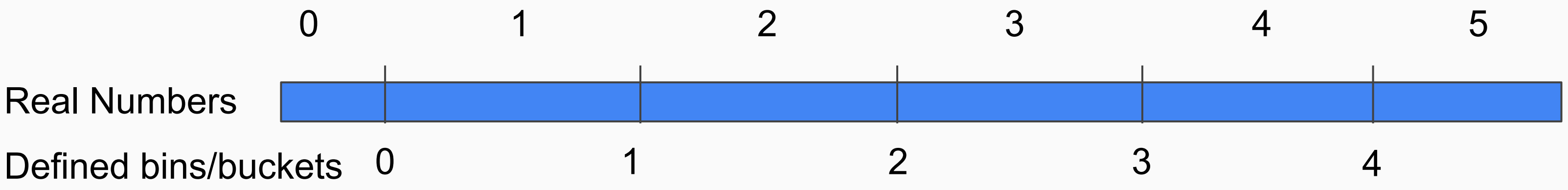
Dealing with continuous state spaces

Observation

Type: Box(4)

Num	Observation	Min	Max
0	Cart Position	-2.4	2.4
1	Cart Velocity	-Inf	Inf
2	Pole Angle	~ -41.8°	~ 41.8°
3	Pole Velocity At Tip	-Inf	Inf

Cartpole - Continuous value problem



Homework walkthrough

-

Some thoughts and next week

- Q learning
  - Temporal Difference learning
  - Values propagating back from later states
  - Learning based on raw experience
- Challenges
  - State space and sufficient exploration (e.g. images of cartpole as state)
  - No notion of state spaces that are nearby
- Understanding of Q learning is important for next week's DQN