UiO : University of Oslo

institutt for informatikk

IN3050/IN4050, Lecture 12
Reinforcement learning

5: Q-learning example

Kai Olav Ellefsen

Next video: On-policy and off-policy learning

Q-learning

- Values are learned by "backing up" values from the current state to the previous one:

$$Q(s_t, a_t) \leftarrow \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\mu}_{\text{learning rate}} \cdot \left(\overbrace{\underbrace{r_{t+1}}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}}}^{\text{learned value}} - \underbrace{Q(s_t, a_t)}_{\text{old value}} \right)$$

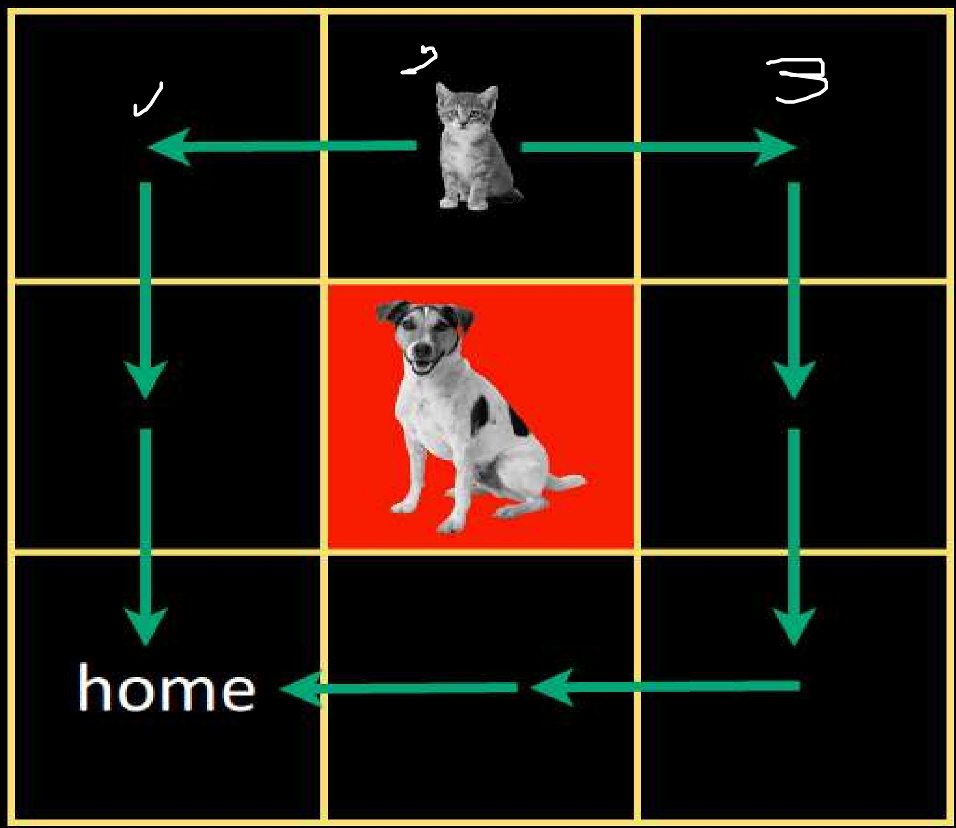
- The same can be done for v-values:

$V(s_t) \leftarrow V(s_t) + \mu(r_{t+1} + \gamma V(s_{t+1}) - V(s_t))$

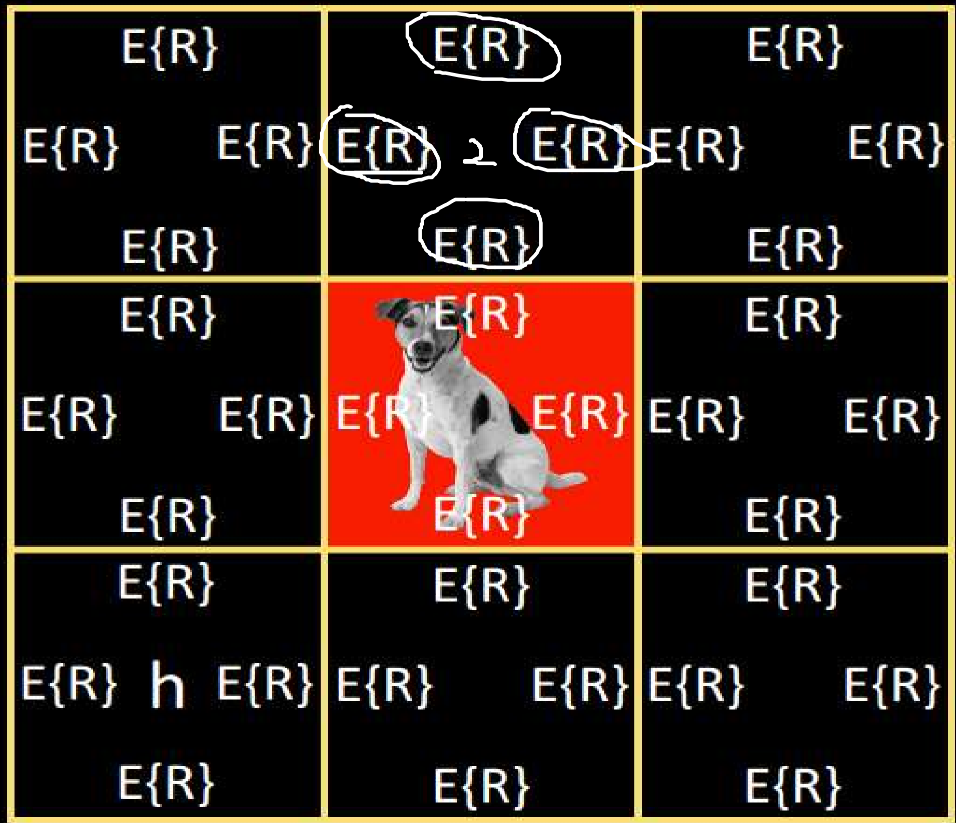
Q-learning example

- Credits: Arjun Chandra

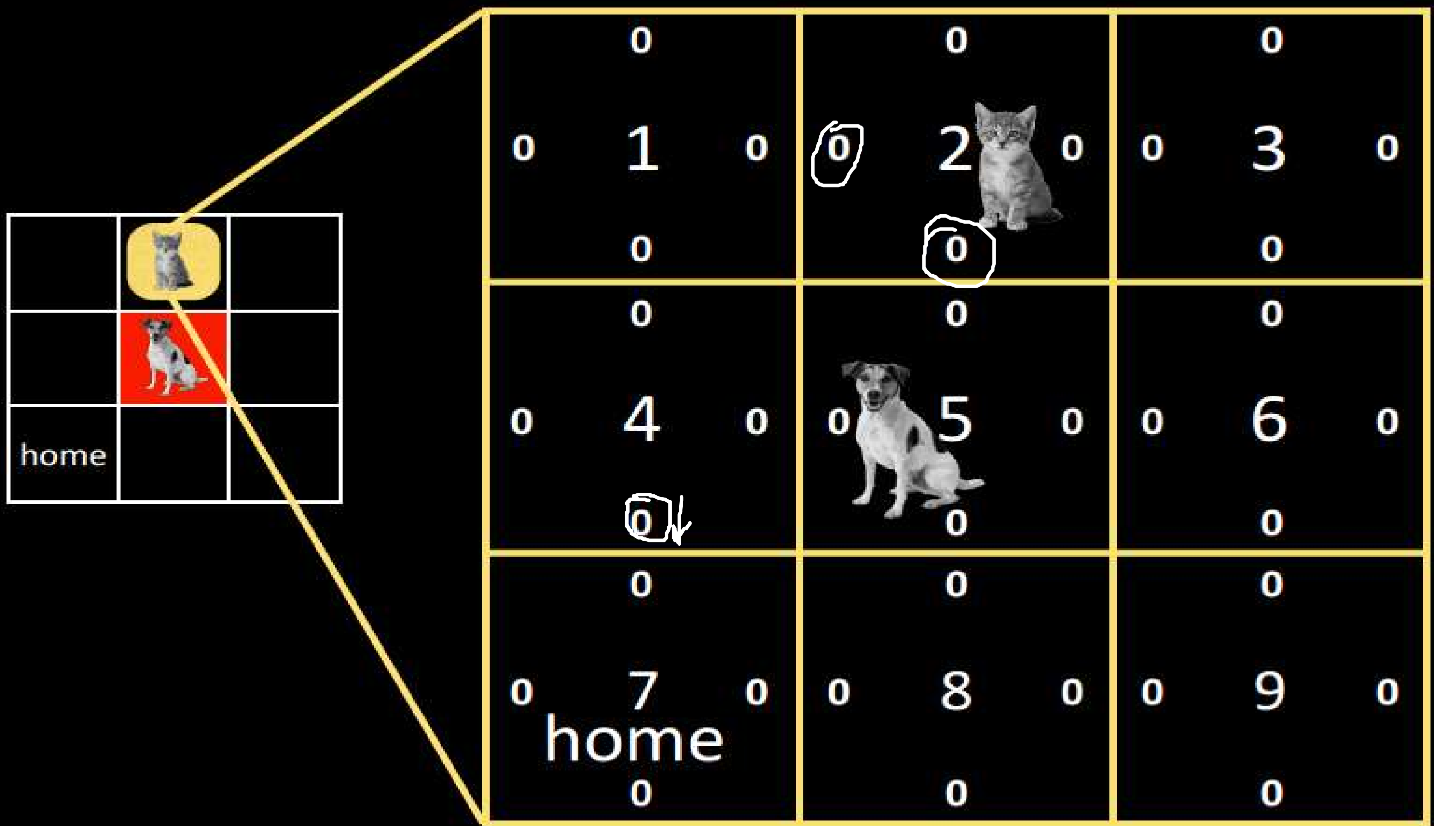
toy problem



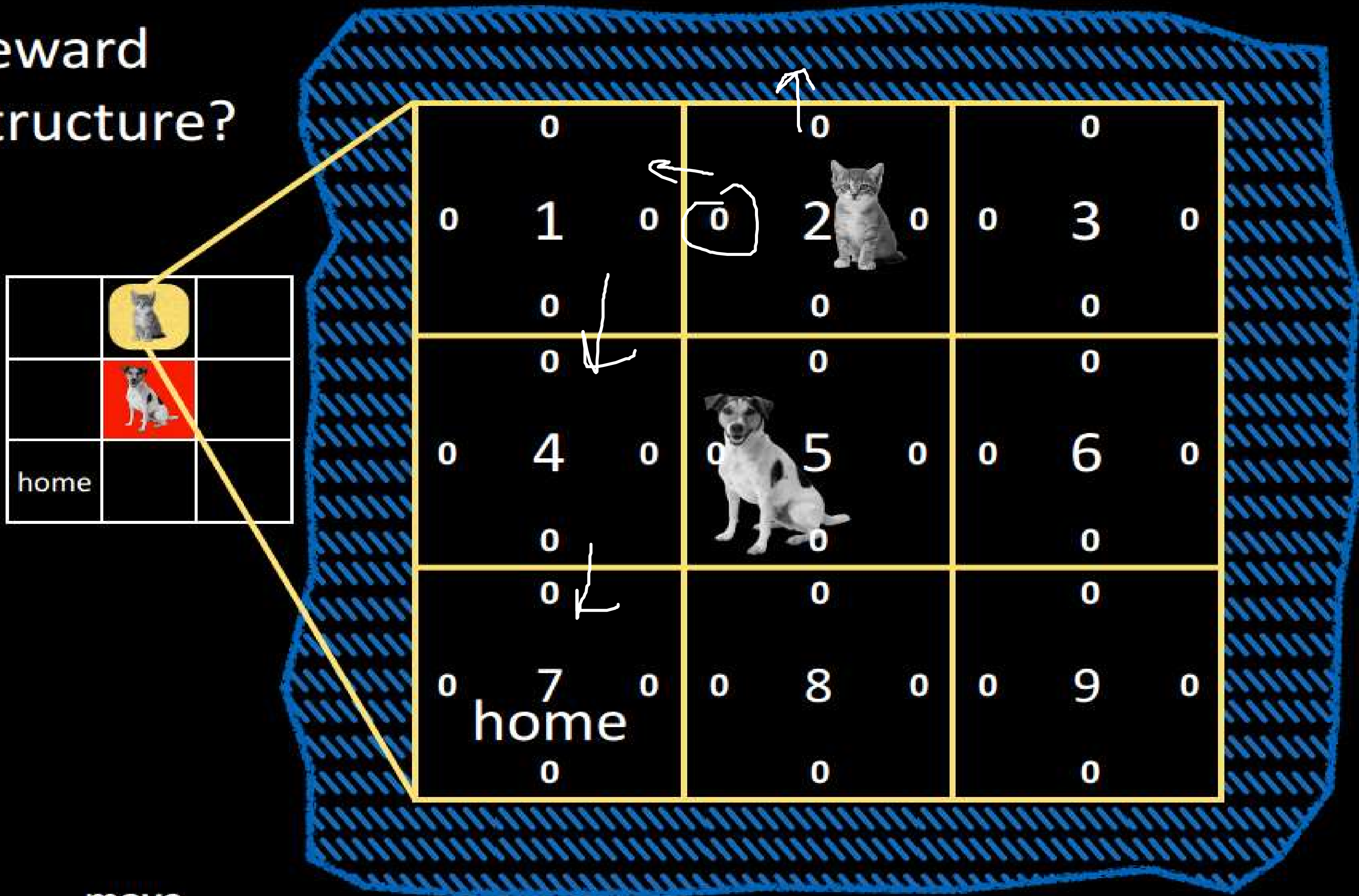
expected long term value of taking
some action in each state,
under some action selection scheme?



our toy problem
lookup table



reward structure?



move...

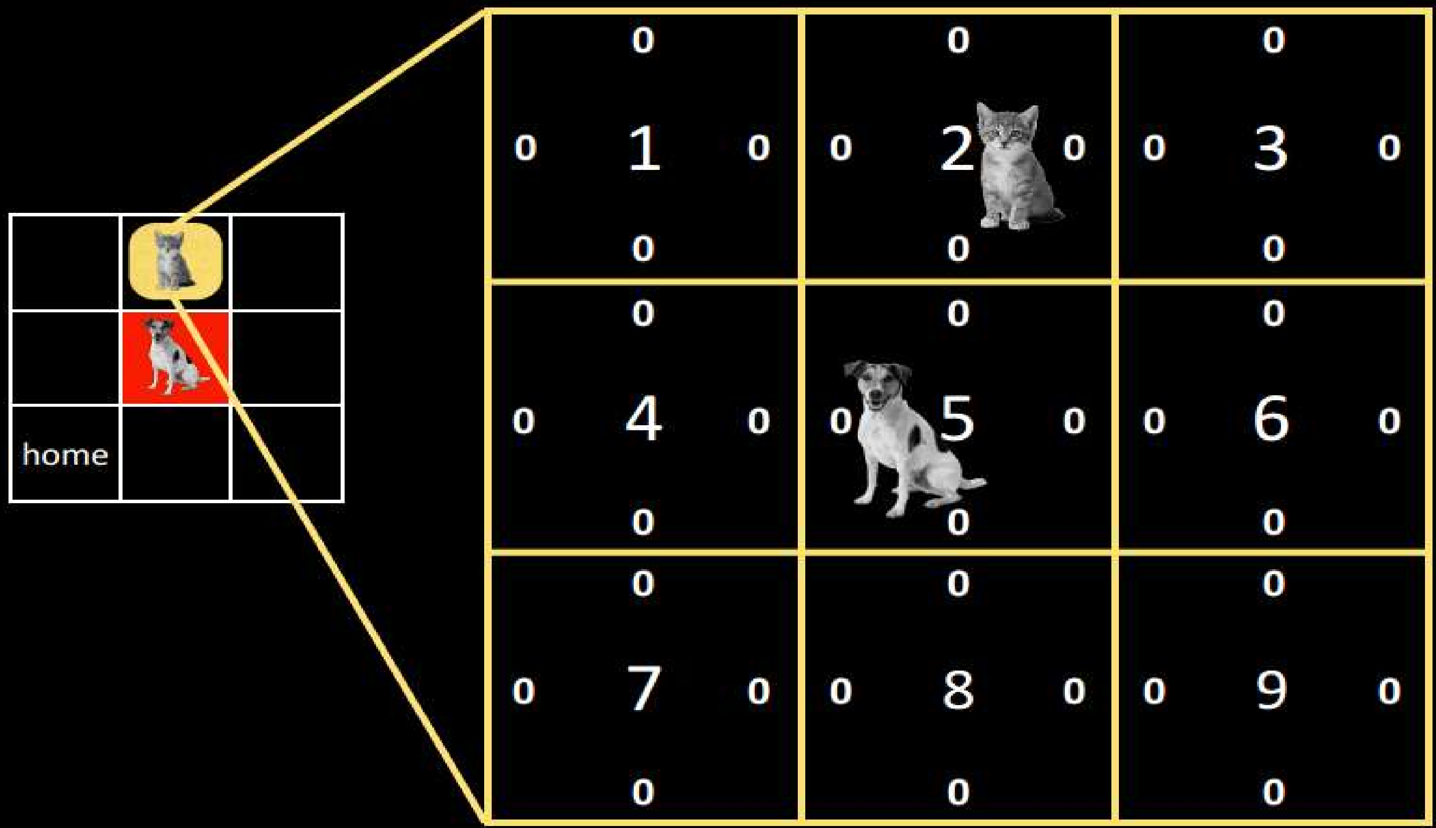
to any cell except 5 and 7:
-1

out of bounds:
-5

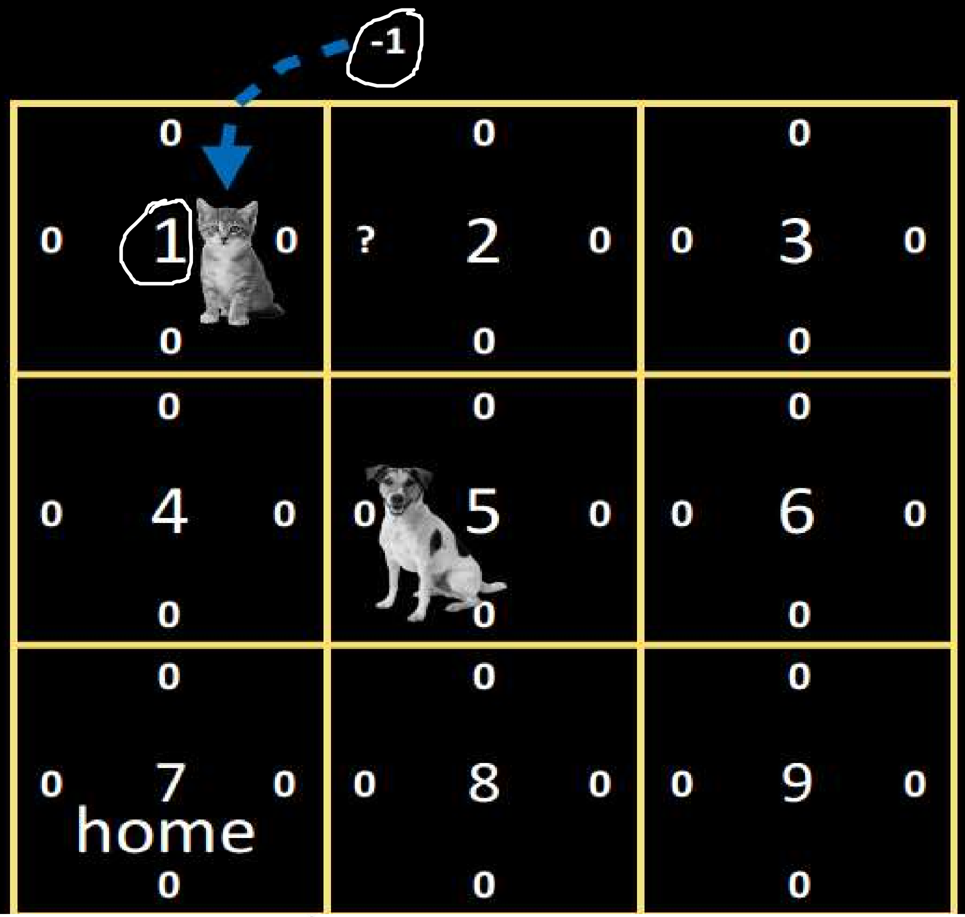
to 5:
-10

to 7/home:
10

let’s fix $\mu = 0.1$, $\gamma = 0.5$

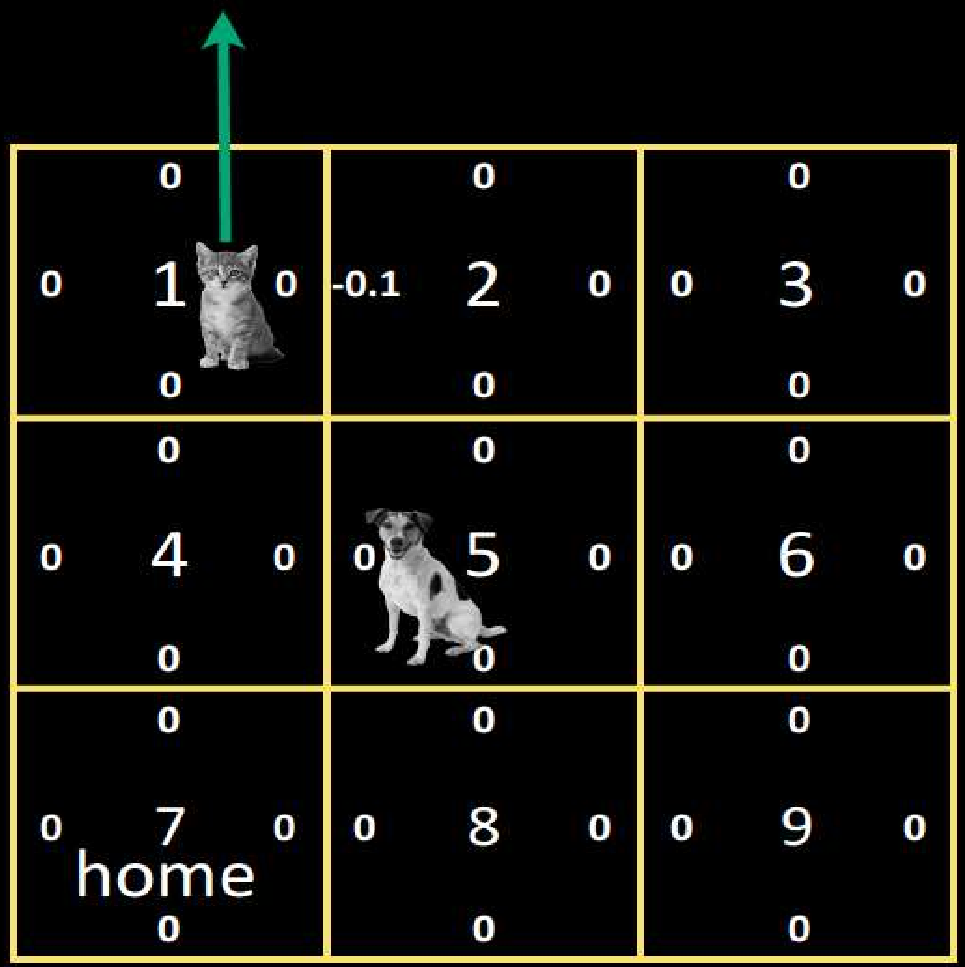


episode 1 begins...



$$\overset{-0.1}{\underline{Q(s_t, a_t)}} \leftarrow \underbrace{\overset{0}{Q(s_t, a_t)}}_{\text{old value}} + \underbrace{\overset{0.1}{\mu}}_{\text{learning rate}} \cdot \left(\overbrace{\underbrace{\overset{-1}{r_{t+1}}}_{\text{reward}} + \underbrace{\overset{0.5}{\gamma}}_{\text{discount factor}} \cdot \underbrace{\overset{0}{\max_a Q(\underline{s_{t+1}}, a)}}_{\text{estimate of optimal future value}}}^{\text{learned value}} - \underbrace{\overset{0}{Q(s_t, a_t)}}_{\text{old value}} \right)$$


0
0 1 0
0
0
-0.1 2 0
0
0
0 3 0
0
0
0 4 0
0
0
0 5 0
0
0
0 6 0
0
0
0 7 0
home
0
0
0 8 0
0
0
0 9 0
0

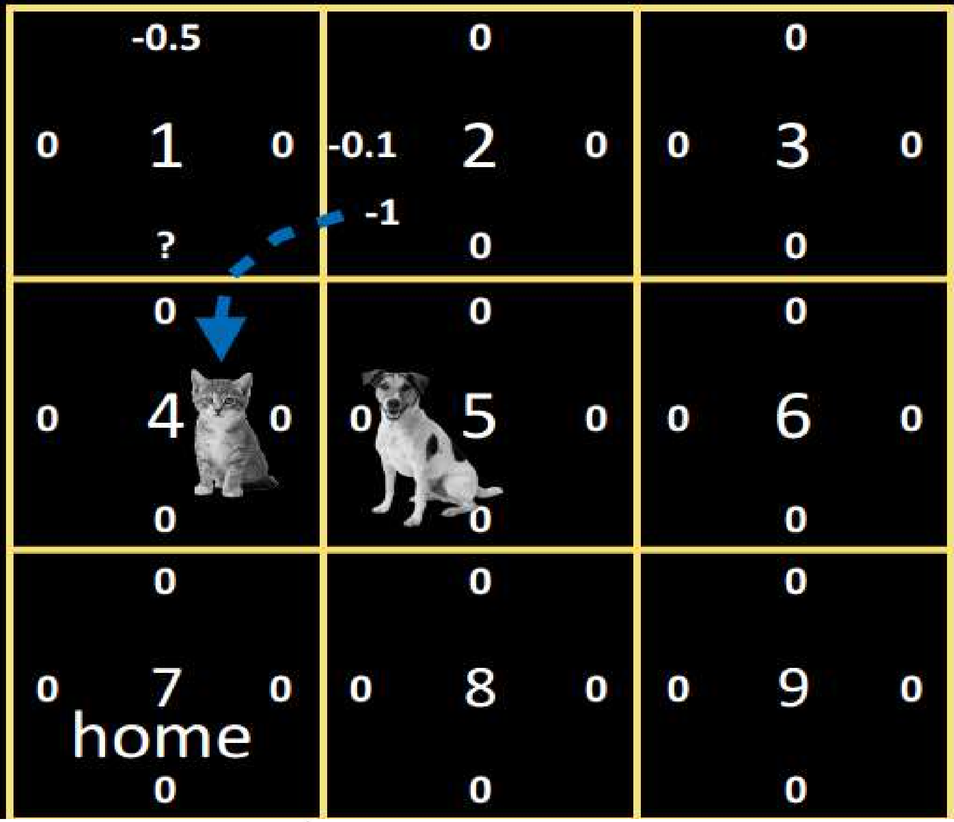


0
0 1 0
0
0
-0.1 2 0
0
0
0 3 0
0
0
0 4 0
0
0
0 5 0
0
0
0 6 0
0
0
0 7 0
home
0
0
0 8 0
0
0
0 9 0
0

$$\overset{-0.5}{Q(s_t,a_t)} \leftarrow \overset{0}{\underbrace{Q(s_t,a_t)}_{\text{old value}}} + \overset{0.1}{\underbrace{\mu}_{\text{learning rate}}} \cdot \left(\overbrace{\overset{-5}{\underbrace{r_{t+1}}_{\text{reward}}} + \overset{0.5}{\underbrace{\gamma}_{\text{discount factor}}} \cdot \overset{0}{\underbrace{\max_{a} Q(s_{t+1},a)}_{\text{estimate of optimal future value}}}}^{\text{learned value}} - \overset{0}{\underbrace{Q(s_t,a_t)}_{\text{old value}}} \right)$$

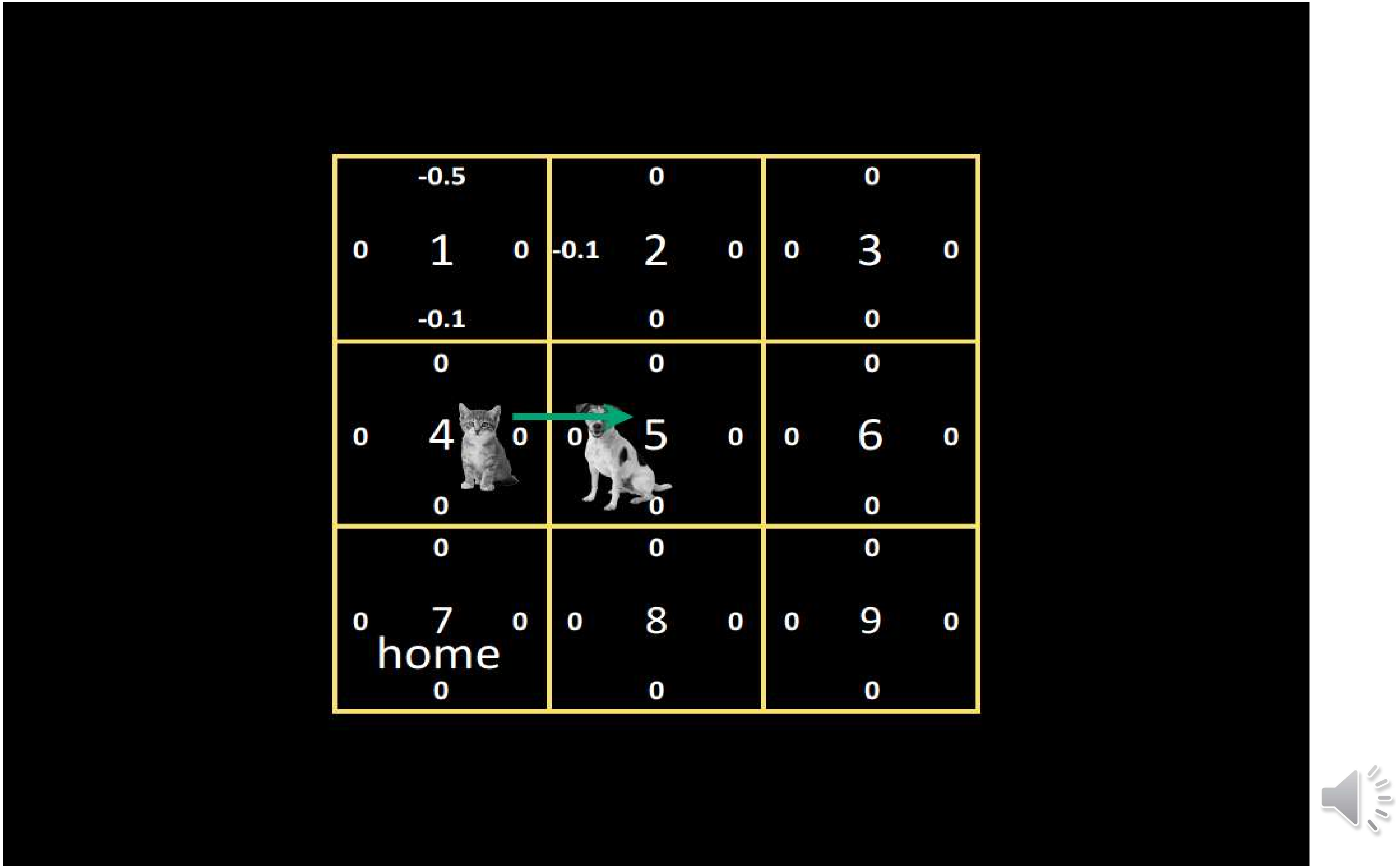

-0.5
0 1 0
0
0
-0.1 2 0
0
0
0 3 0
0
0
0 4 0
0
0
0 5 0
0
0
0 6 0
0
0
0 7 0
home
0
0
0 8 0
0
0
0 9 0
0


-0.5
0 1 0
0
-0.1 2 0
0
0
0 3 0
0
0
0 4 0
0
0
0 5 0
0
0
0 6 0
0
0
0 7 0
home
0
0
0 8 0
0
0
0 9 0
0



-0.1

$$Q(s_t,a_t) \leftarrow \underbrace{Q(s_t,a_t)}_{\text{old value}} + \underbrace{\mu}_{\text{learning rate}} \cdot \left(\overbrace{\underbrace{r_{t+1}}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1},a)}_{\text{estimate of optimal future value}}}^{\text{learned value}} - \underbrace{Q(s_t,a_t)}_{\text{old value}} \right)$$

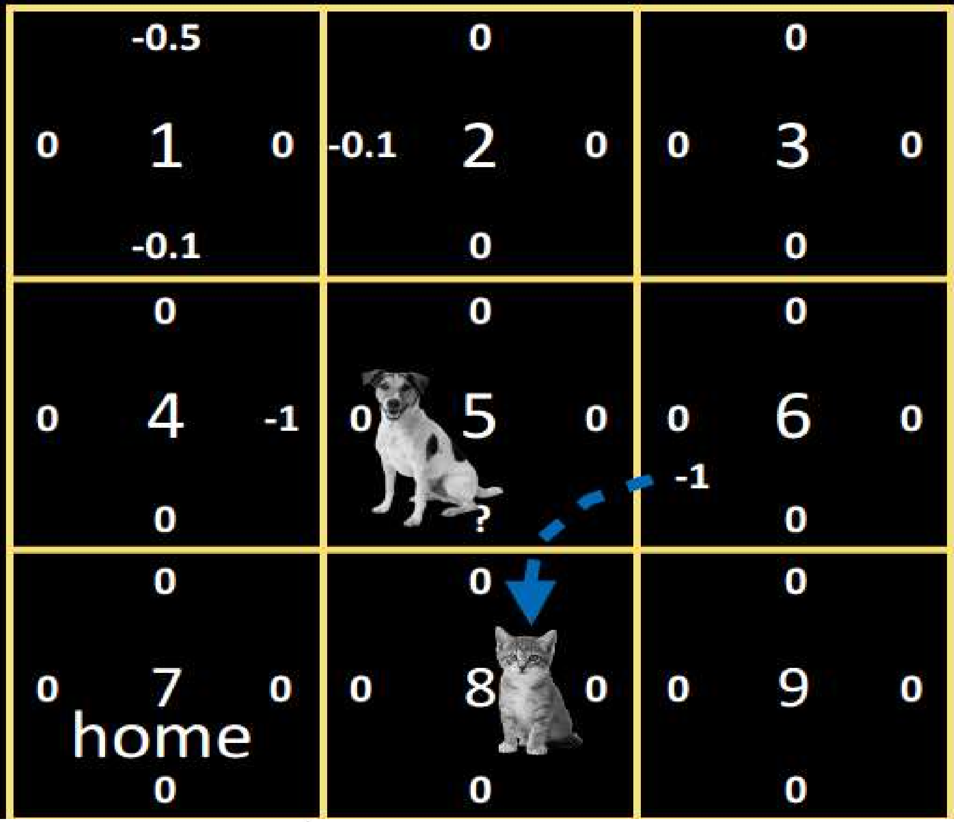

-0.5
0
1
0
-0.1
0
-0.1
2
0
0
0
0
3
0
0
0
0
4
0
0
0
0
5
0
0
0
0
6
0
0
0
0
7
0
home
0
0
0
8
0
0
0
0
9
0
0

-10

$$Q(s_t, a_t) \leftarrow \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\mu}_{\text{learning rate}} \cdot \left(\overbrace{\underbrace{r_{t+1}}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}}}^{\text{learned value}} - \underbrace{Q(s_t, a_t)}_{\text{old value}} \right)$$

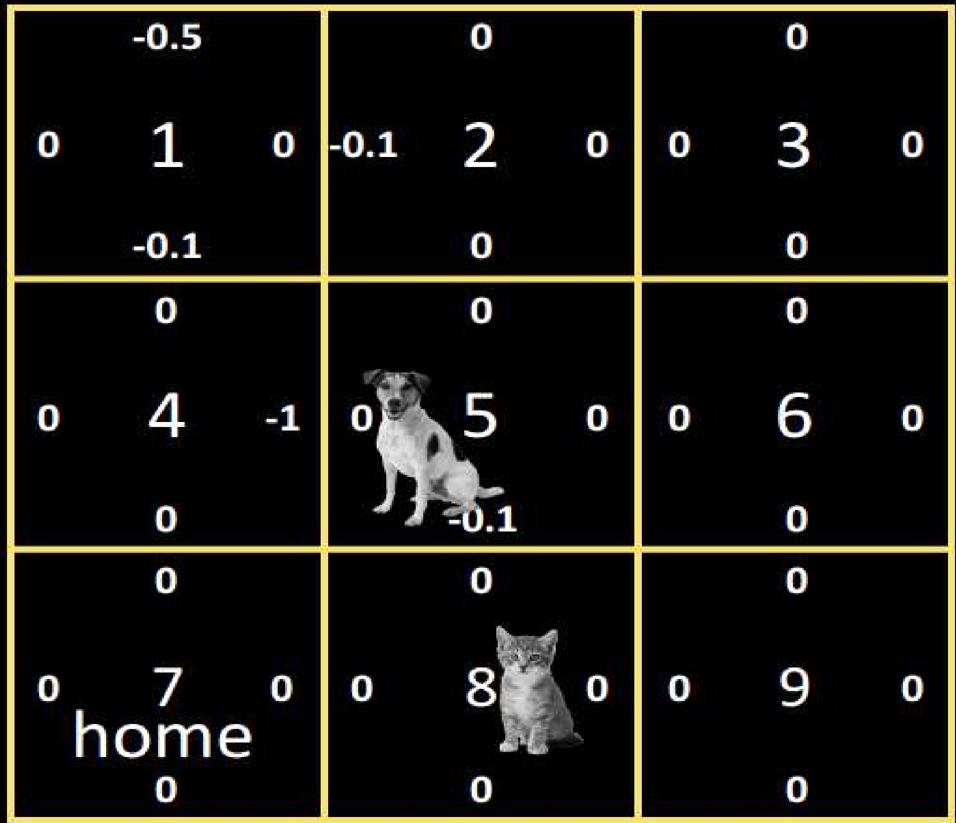

-0.5
0
0
0
1
0
-0.1
2
0
0
3
0
-0.1
0
0
0
0
0
0
4
-1
0
5
0
0
6
0
0
0
0
0
0
0
0
7
0
0
8
0
0
9
0
home
0
0
0


-0.5
0 1 0
-0.1
0
-0.1 2 0
0
0
0 3 0
0
0
0 4 -1
0
0
0 5 0
0
0
0 6 0
0
0
0 7 0
home
0
0
0 8 0
0
0
0 9 0
0

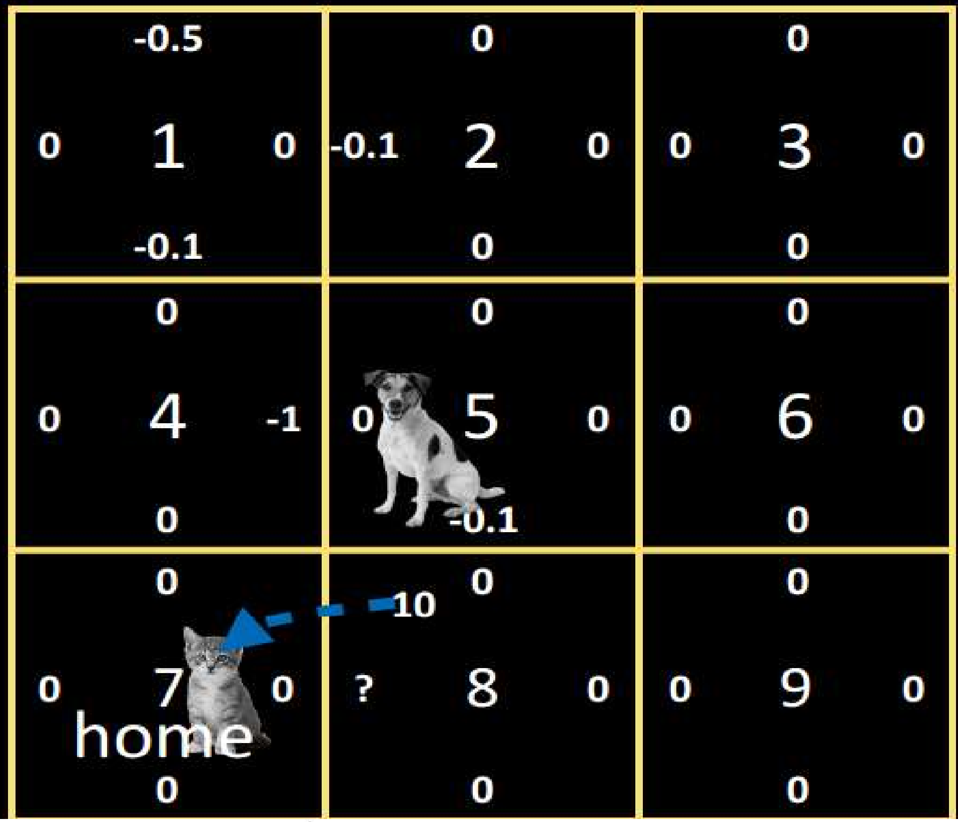


-0.1

$$Q(s_t, a_t) \leftarrow \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\mu}_{\text{learning rate}} \cdot \left(\overbrace{\underbrace{r_{t+1}}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}}}^{\text{learned value}} - \underbrace{Q(s_t, a_t)}_{\text{old value}} \right)$$


-0.5
0 1 0
-0.1
0
-0.1 2 0
0
0
0 3 0
0
0
0 4 -1
0
0
0 5 0
-0.1
0
0 6 0
0
0
0 7 0
home
0
0
0 8 0
0
0
0 9 0
0


-0.5
0
0
0
1
0
-0.1
2
0
0
3
0
-0.1
0
0
0
0
0
0
4
-1
0
5
0
0
6
0
0
-0.1
0
0
0
0
0
0
7
0
0
8
0
0
9
0
home
0
0
0



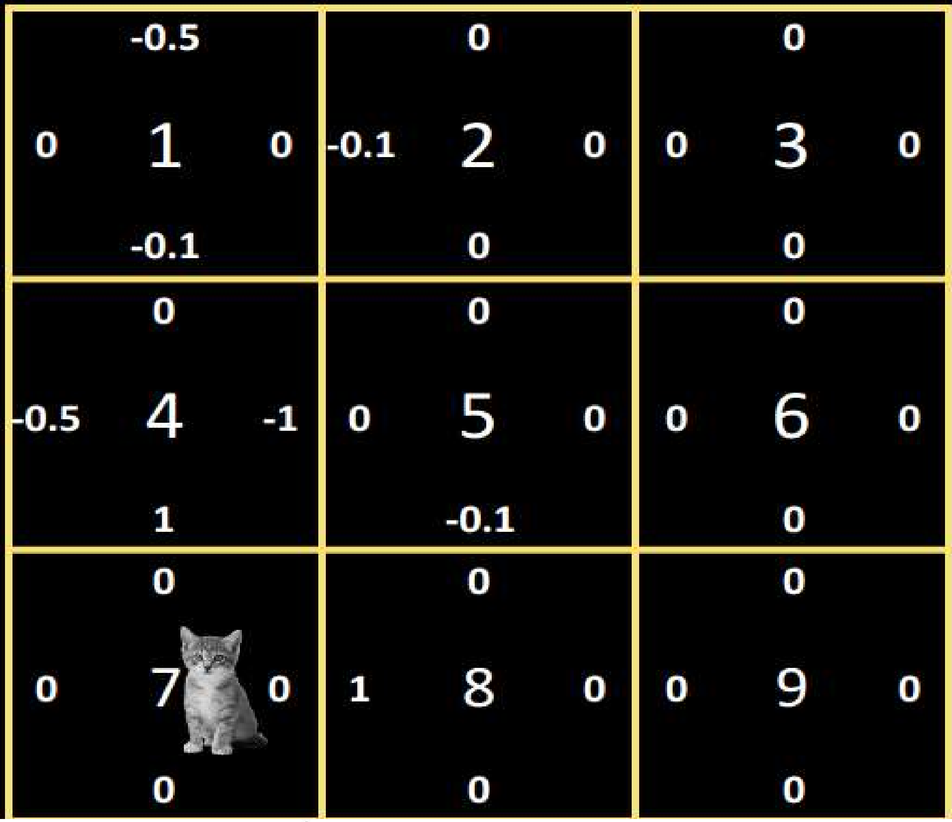
1.0

$$Q(s_t, a_t) \leftarrow \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\mu}_{\text{learning rate}} \cdot \left(\overbrace{\underbrace{r_{t+1}}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}}}^{\text{learned value}} - \underbrace{Q(s_t, a_t)}_{\text{old value}} \right)$$

let's work out the next episode, starting at state 4

go WEST and then SOUTH

how does the table change?

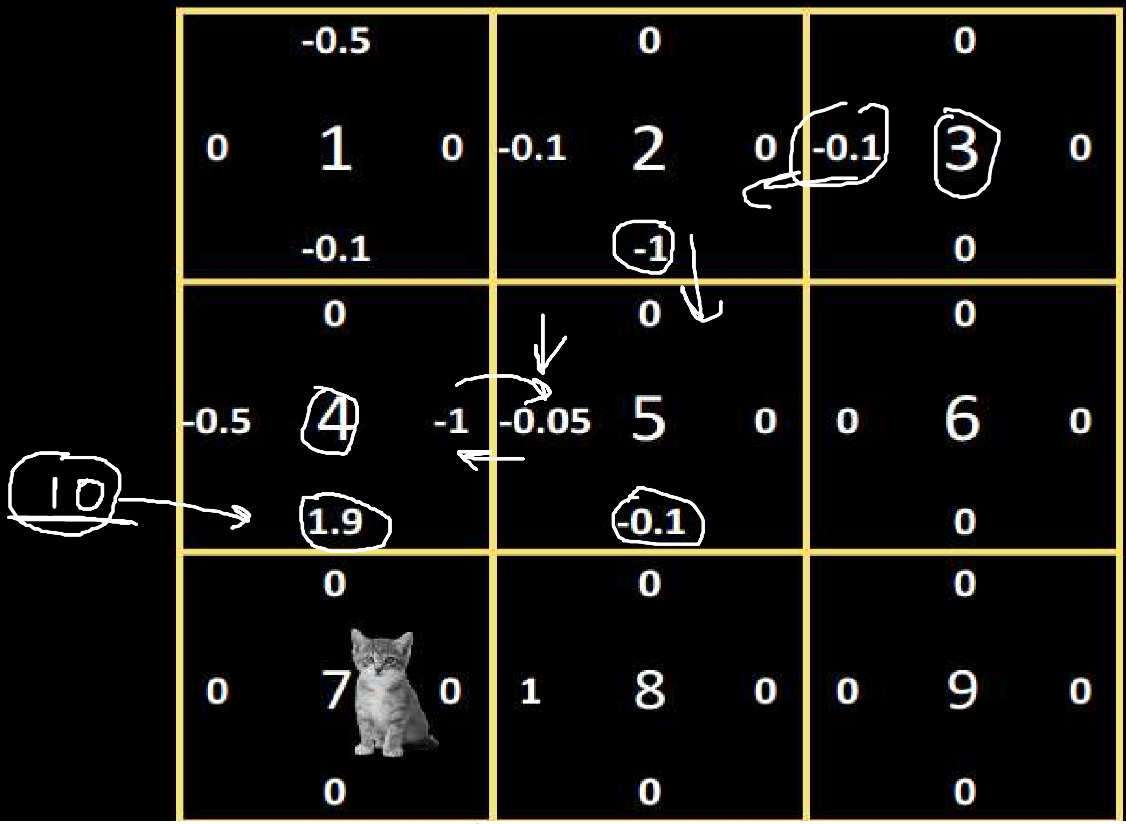


$$Q(s_t,a_t) \leftarrow \underbrace{Q(s_t,a_t)}_{\text{old value}} + \underbrace{\mu}_{\text{learning rate}} \cdot \left(\overbrace{\underbrace{r_{t+1}}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1},a)}_{\text{estimate of optimal future value}}}^{\text{learned value}} - \underbrace{Q(s_t,a_t)}_{\text{old value}} \right)$$

and the next episode, starting at state 3

go WEST -> SOUTH -> WEST -> SOUTH

how does the table change?



$$Q(s_t,a_t) \leftarrow \underbrace{\overset{0}{Q(s_t,a_t)}}_{\text{old value}} + \underbrace{\overset{01}{\mu}}_{\text{learning rate}} \cdot \left(\overbrace{\underbrace{\overset{-1}{r_{t+1}}}_{\text{reward}} + \underbrace{\overset{0.5}{\gamma}}_{\text{discount factor}} \cdot \underbrace{\overset{1.0}{\max_a Q(s_{t+1},a)}}_{\text{estimate of optimal future value}}}^{\text{learned value}} - \underbrace{\overset{0}{Q(s_t,a_t)}}_{\text{old value}}\right)$$

UiO : University of Oslo

institutt for informatikk

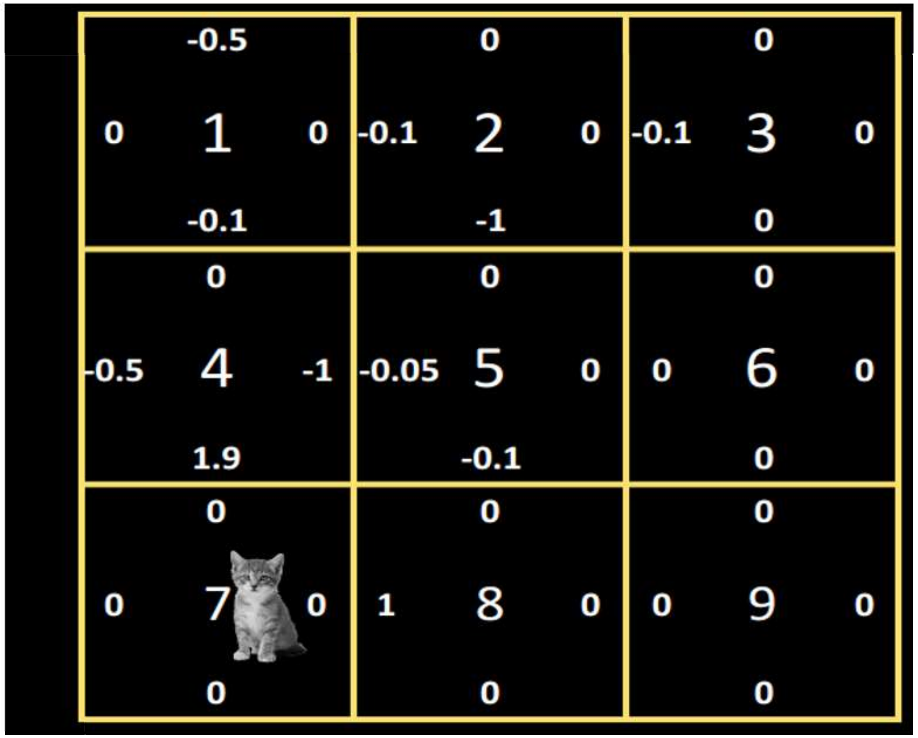
IN3050/IN4050, Lecture 12 Reinforcement learning

6: On-policy and off-policy learning

Kai Olav Ellefsen

Action selection

- Estimate the *value* of each action: $Q_{s,t}(a)$
- Decide whether to:
  - Explore, or
  - exploit



Action selection

- The function deciding which action to take in each state is called the policy, $\boldsymbol{\pi}$. Examples:
  - Greedy: Always choose most valuable action
  - ϵ-greedy: Greedy, except small probability (ϵ) of choosing the action at random
- The q-learning we just saw is an example of *off-policy learning*:

$$Q(s_t, a_t) \leftarrow \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\mu}_{\text{learning rate}} \cdot \left(\overbrace{\underbrace{r_{t+1}}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}}}^{\text{learned value}} - \underbrace{Q(s_t, a_t)}_{\text{old value}} \right)$$

Off-Policy Learning

The Q-Learning Algorithm

- **Initialisation**
  - set $Q(s, a)$ to small random values for all s and a
- Repeat:
  - initialise s
  - repeat:
    - select action a using ϵ-greedy or another policy
    - take action a and receive reward r
    - sample new state s'
    - update $Q(s, a) \leftarrow Q(s, a) + \mu(r + \gamma \max_{a'} Q(s', a') - Q(s, a))$
    - set $s \leftarrow s'$
  - For each step of the current episode
- Until there are no more episodes

Source: Marsland

On-Policy Learning

The Sarsa Algorithm

- **Initialisation**
  - set $Q(s, a)$ to small random values for all s and a
- Repeat:
  - initialise s
  - choose action a using the current policy
  - repeat:
    - take action a and receive reward r
    - sample new state s'
    - choose action a' using the current policy
    - update $Q(s, a) \leftarrow Q(s, a) + \mu(r + \gamma Q(s', a') - Q(s, a))$
    - $s \leftarrow s'$, $a \leftarrow a'$
  - for each step of the current episode
- Until there are no more episodes

Off-Policy Learning

The Q-Learning Algorithm

- **Initialisation**
  - set $Q(s, a)$ to small random values for all s and a
- Repeat:
  - initialise s
  - repeat:
    - ∗ select action a using ϵ-greedy or another policy
    - ∗ take action a and receive reward r
    - ∗ sample new state s'
    - ∗ update $Q(s, a) \leftarrow Q(s, a) + \mu(r + \gamma \max_{a'} Q(s', a') - Q(s, a))$
    - ∗ set $s \leftarrow s'$
  - For each step of the current episode
- Until there are no more episodes

Source: Marsland

On-Policy Learning

The Sarsa Algorithm

- **Initialisation**
  - set $Q(s, a)$ to small random values for all s and a
- Repeat:
  - initialise s
  - choose action a using the current policy
  - repeat:
    - ∗ take action a and receive reward r
    - ∗ sample new state s'
    - ∗ choose action a' using the current policy
    - ∗ update $Q(s, a) \leftarrow Q(s, a) + \mu(r + \gamma Q(s', a') - Q(s, a))$
    - ∗ $s \leftarrow s'$, $a \leftarrow a'$
  - for each step of the current episode
- Until there are no more episodes

Off-Policy Learning

The Q-Learning Algorithm

- **Initialisation**
  - set $Q(s, a)$ to small random values for all s and a
- Repeat:
  - initialise s
  - repeat:
    - $*$ select action a using ϵ-greedy or another policy
    - $*$ take action a and receive reward r
    - $*$ sample new state s'
    - $*$ update $Q(s, a) \leftarrow Q(s, a) + \mu(r + \gamma \underline{\max_{a'}} Q(s', a') - Q(s, a))$
    - $*$ set $s \leftarrow s'$
  - For each step of the current episode
- Until there are no more episodes

On-Policy Learning

The Sarsa Algorithm

- **Initialisation**
  - set $Q(s, a)$ to small random values for all s and a
- Repeat:
  - initialise s
  - choose action a using the current policy
  - repeat:
    - $*$ take action a and receive reward r
    - $*$ sample new state s'
    - $*$ choose action a' using the current policy
    - $*$ update $Q(s, a) \leftarrow Q(s, a) + \mu(\underline{r} + \gamma \boxed{Q(s', a')} - Q(s, a))$
    - $*$ $s \leftarrow s'$, $a \leftarrow a'$
  - for each step of the current episode
- Until there are no more episodes

Source: Marsland

On-policy vs off-policy learning

- Q-learning (off-policy):

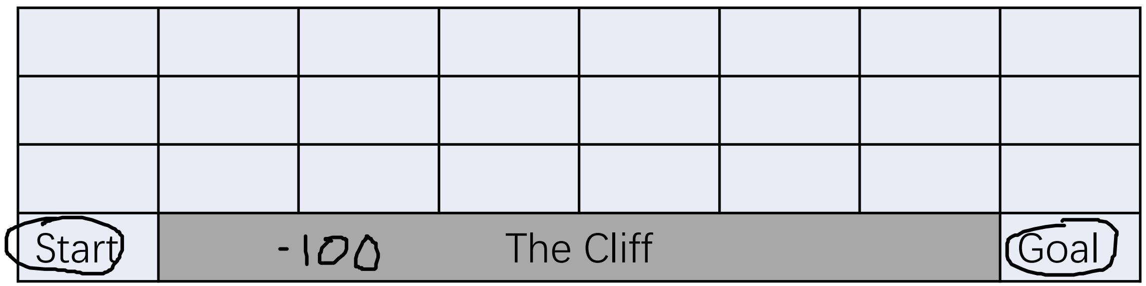
$$Q(s_t, a_t) \leftarrow \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\mu}_{\text{learning rate}} \cdot \left(\overbrace{\underbrace{r_{t+1}}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}}}^{\text{learned value}} - \underbrace{Q(s_t, a_t)}_{\text{old value}} \right)$$

- Sarsa (on-policy):

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \mu[r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)]$$

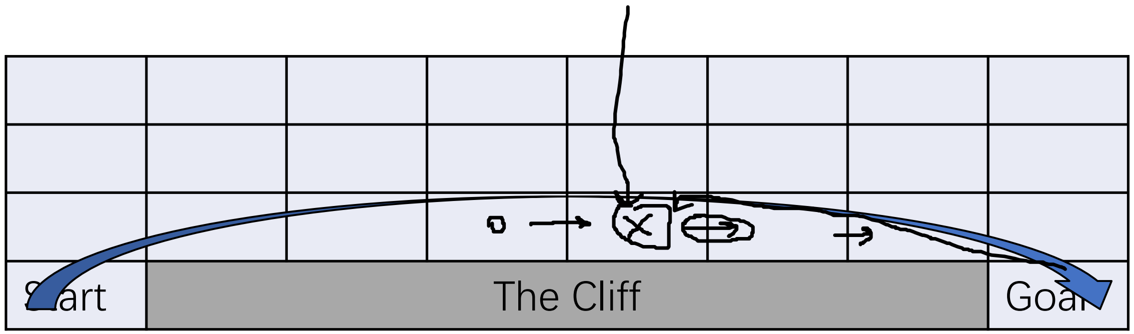
On-policy vs off-policy learning

- Reward structure: Each move: -1. Move to cliff: -100.
- Policy: 90% chance of choosing best action (exploit). 10% chance of choosing random action (explore).



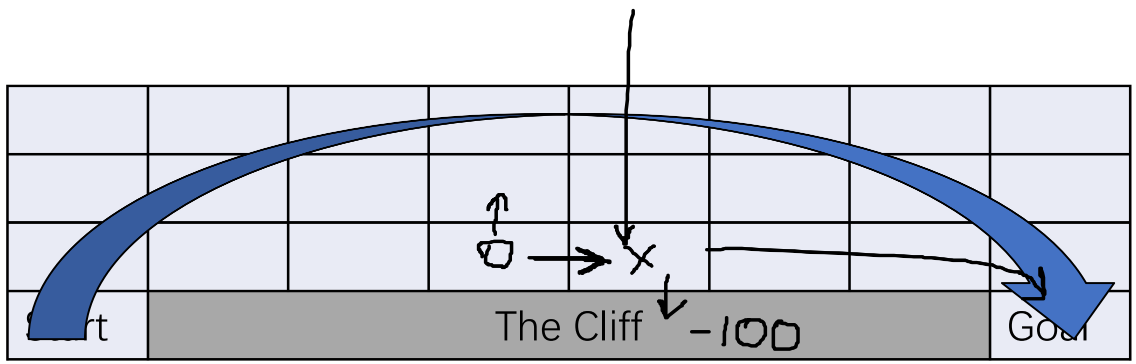
On-policy vs off-policy learning: Q-learning

- Always assumes optimal action -> does not visit cliff often while learning. Therefore, does not learn that cliff is dangerous.
- Resulting path is efficient, but risky.



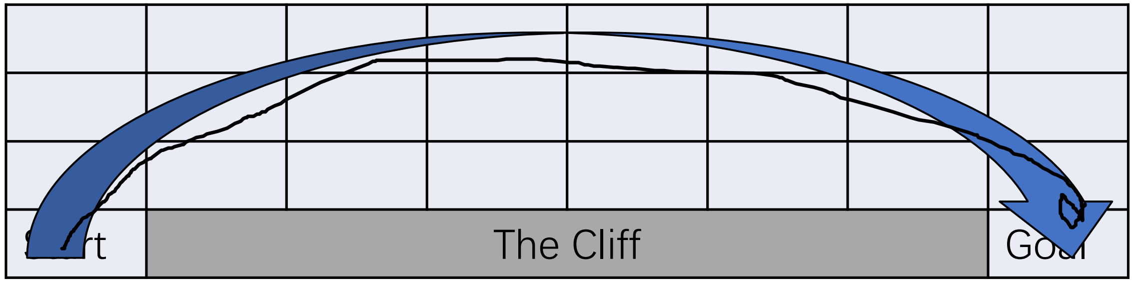
On-policy vs off-policy learning: sarsa

- During learning, we more frequently end up outside the cliff (due to the 10% chance of exploring in our policy).
- That info propagates to all states, generating a safer plan.



Which plan is better?

- sarsa (on-policy):



- Q-learning (off-policy):