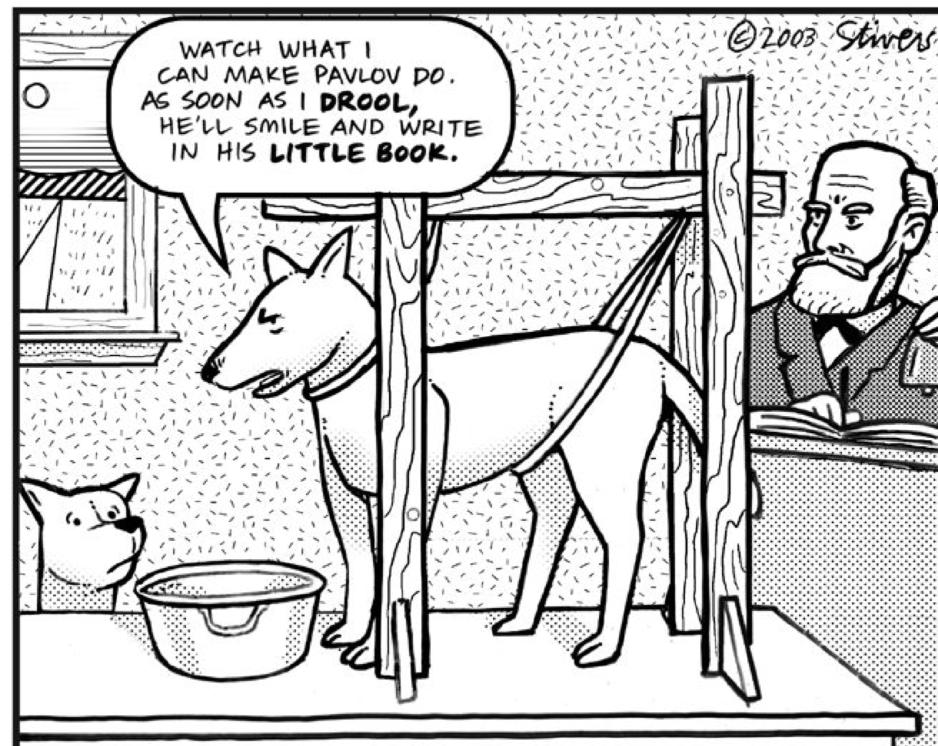
Reinforcement Learning

Pavlocs experiments on conditioning

- Operational conditioning
- Pavlov reinforced the connection between salvation and ringing a bell using a food stimulus
  - At the end the dog drooled just on ringing the bell without being exposed to any food



http://api.ning.com/files/YwzKAeVaunUIhGHXAAMUzA*L4WUBommkdp6pOLnWq0s_/ageofthesage.org.gif

Q-learning

- A reinforcement learning variant
- Does not require a world model to exist
- Requires only direct feedback from the environment after an action was executed

Reward I

- Immediate reward
  - the reward (positive or negative) is the “feed back” directly after the action was executed
  - Is optimal for learning as you immediately know if your action was a good or bad
  - Domains with immediate rewards are rare since actions take time to affect the environment

http://www.weber-rudolf.de/images/pruegelstrafe2.jpg

Reward II

- Delayed reward
  - A reward that is undisclosed to you only after a sequence of actions
  - This kind of reward is delayed such that you never know which of your actions was good or bad

http://i.ytimg.com/vi/cwE-DgImzFs/0.jpg

Terms and Definitions

- State: is the actual state s_t of the agent
- Action: a potential action a that brings the agent into a new state s_{t+1}
- Reward: is received in state s_{t+1} after Action a was applied

http://media.kunst-fuer-alle.de/img/37/g/37_2770123~_daniela-hofer-(f1-online)_bullterrier,-kampfhund,-hungrig.jpg

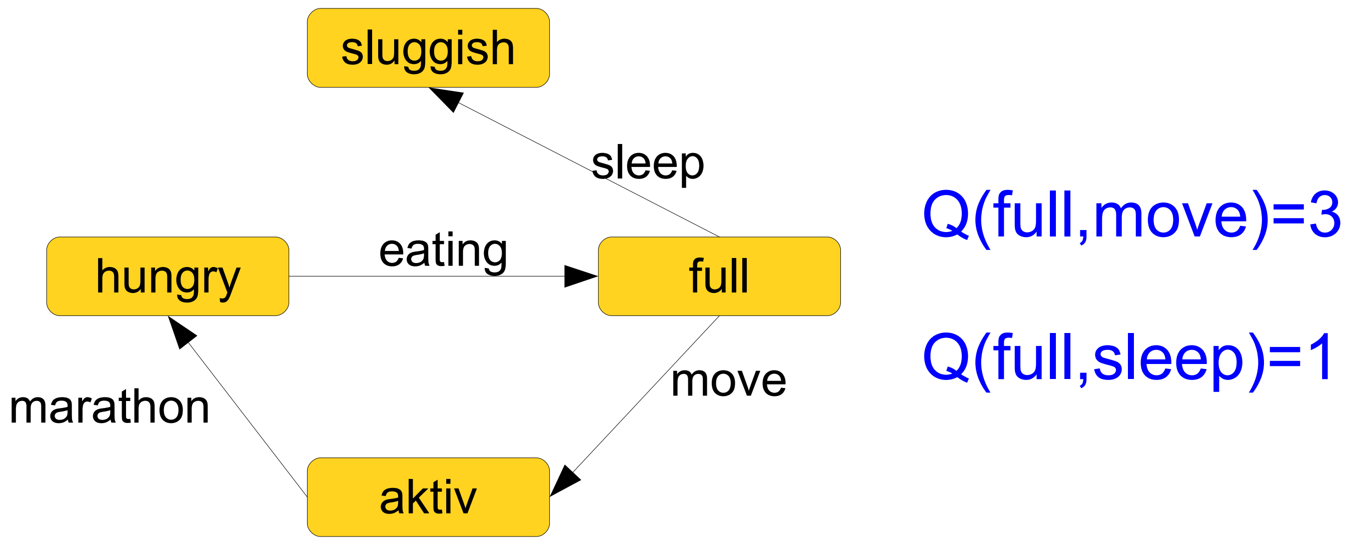
http://www.fotobank.ru/img/DV05-9815.jpg?size=l

http://images2.cafemom.com/images/user/gallery/post_1492134_1236080064_med.jpg?imageId=13070230

s_t a s_{t+1}

Terms and Definitions

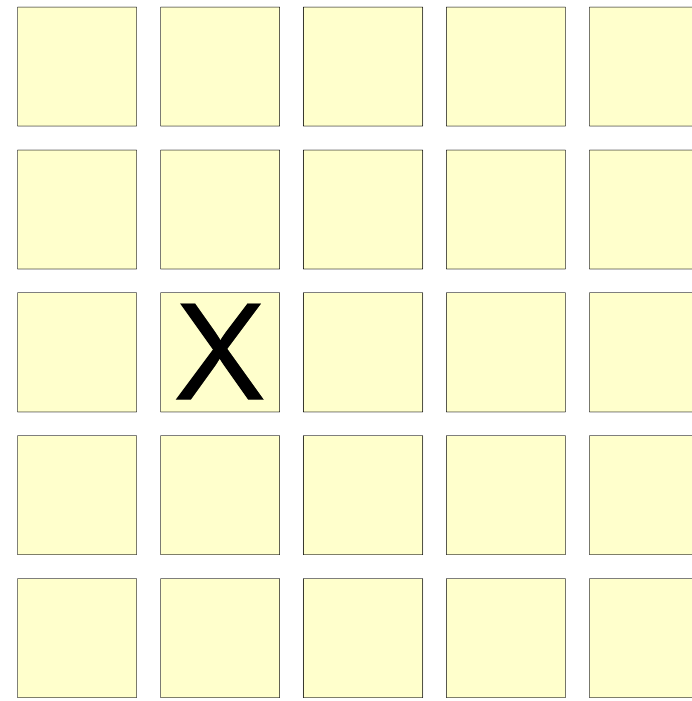
- Policy Q is a mapping from states to actions Q(s,a)
- Q stores the expected reward after Action a is applied in State s



- Goal: find a policy Q such that the long-term reward is maximized

Q-Learning example

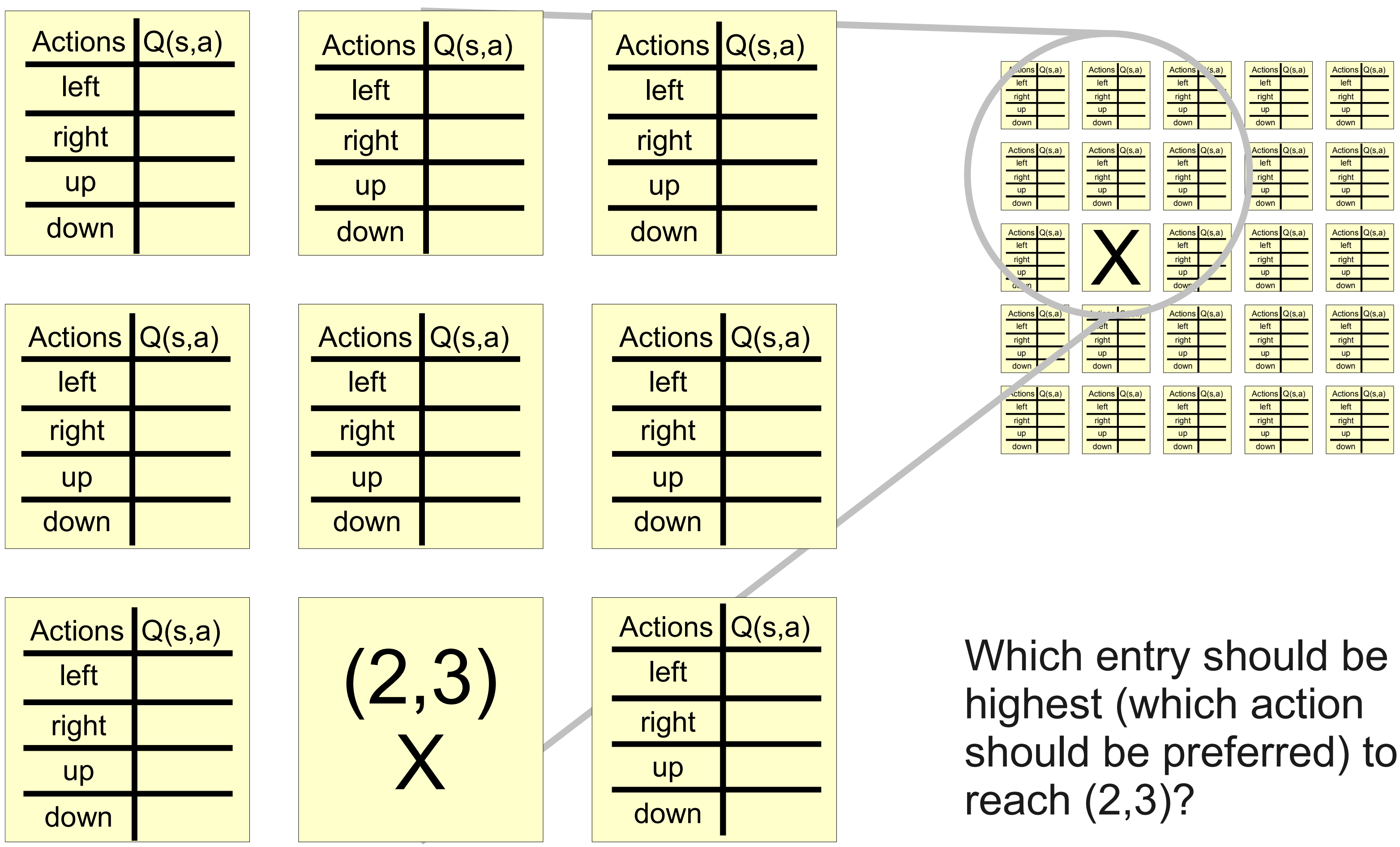
- Actions A={left,right,up,down}



- States S are 25 “places”
- An immediate reward is payed at field (2,3)
- Delayed Reward
  - If we start at (2,1) and go down two fields to (2,3) we must transform the immediate reward at state (2,1) into a delayed reward to “remember” which way we have to go when starting from (2,1) the next time

Q-Learning example

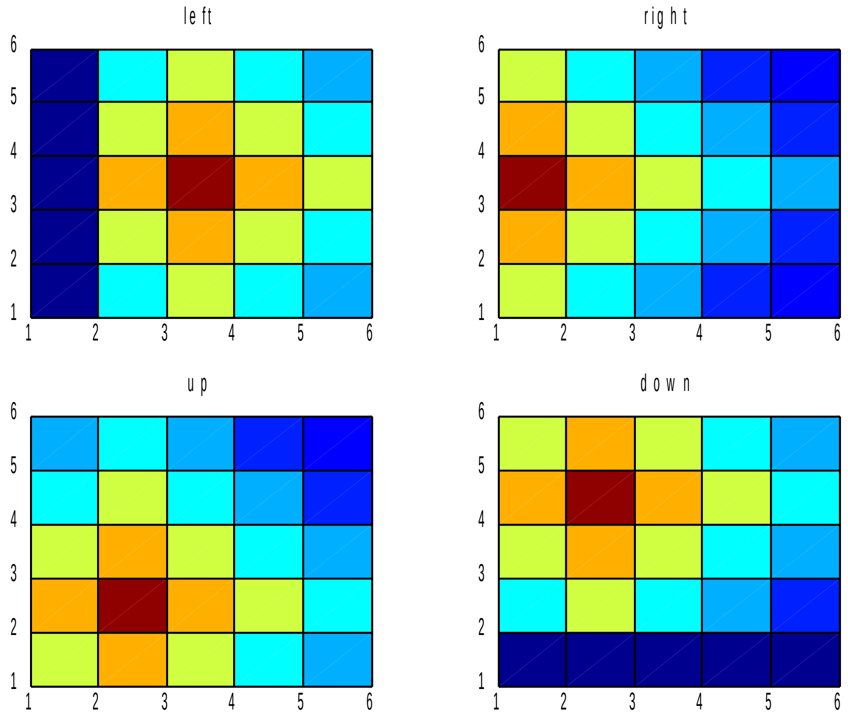
- Find a Q(S,A) such that the reward is maximized



Which entry should be highest (which action should be preferred) to reach (2,3)?

Q-Learning example

- Use Qlearning.m example from the Material Section to see the Q(S,A) crowding
- In each iteration the algorithm updates the Q-Value for each action in each state
  - This can be intepreted as the robot can “try” all possible actions in all possible states



Q-Learning Algorithm

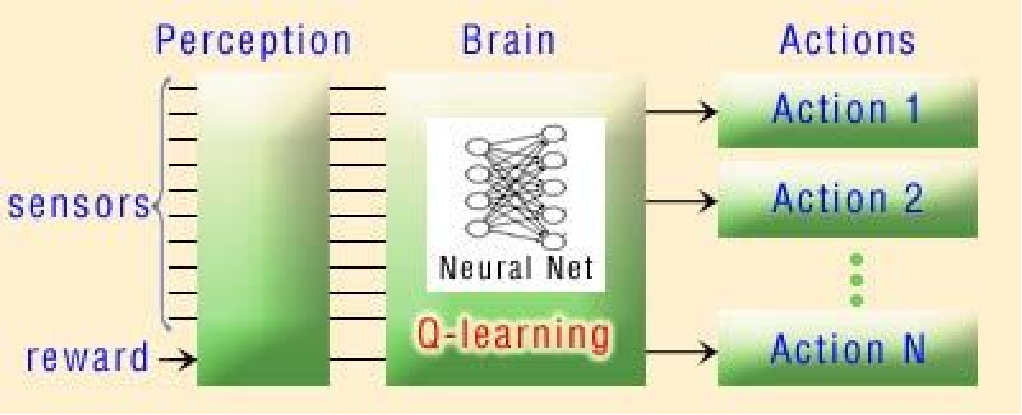
- Update formula using a discount factor $0 \leq \gamma \leq 1$

$$\overbrace{Q(s_t, a_t)}^{\text{The expected benefit if Action } a_t \text{ would be executed in State } s_t} = \underbrace{r_{t+1}}_{\text{The reward that awaits us after Action } a_t \text{ was executed in } s_t} + \gamma \cdot \overbrace{max_{a \in A} Q(s_{t+1}, a)}^{\text{The maximal reward we can expect in the new State}_{t+1}}$$

- For all states and all actions, update the formula until no values change any more
- Use the Policy Q to execute the optimal action a_t given the actual State s_t

Extension of Standard Q-Learning

- Needs to discretize the world (into state) and its manipulations (into actions)
  - Solution: Q-Learning with continuous states and actions
- The Table Q(S,A) can have many entries and could become a mess
  - Use a function that "approximates" Q(S,A)
    - e.g. A neural net



http://elsy.gdan.pl/images/stories/how_works/idea.jpg

Q-Learning Framework

- Example in QLearning folder of Materials.zip
  - Try qApollo.jar to see a robot learning to fly a landing capsule with reinforcement learning
  - For this domain it is a good choice to use a simulator to make the first steps!