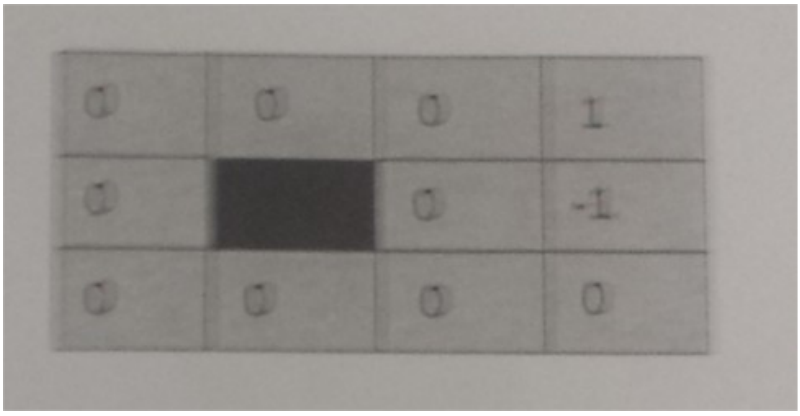
Reinforcement Learning Klausur SS2016

52 Punkte Insgesamt (Eine 4,0 gabs ab 20 Punkte)

Question 1 --- Value Iteration & Policy Iteration (10 Pts)

Given an MDP as in the map there are two terminal states with a reward of 1 and -1. The reward function is $r(s, a, s') = r(s')$ (where the values of r(s') are given in the map). The agent can take four actions: {left, right, up down}. Assuming that the dynamics is deterministic and $\gamma = 1$.



a) Using Value Iteration: Initialize $V_0(s) = 0$ for all s

- Write the update formula for the value functions (2pts)
- Compute $V_1(s), forall s$ (the value function after 1st iteration) (2pts)

b) Using Policy Iteration: Assume that the initial policy $\pi_0(s) = right, \forall s$

- Write the Policy Update formula of the policy iteration algorithm (2 Pts)
- Compute the value function of π_0 (2 Pts)
- Compute the policy at the second iteration $\pi_1(s), \forall s$ (2 Pts)

Some Solutions:

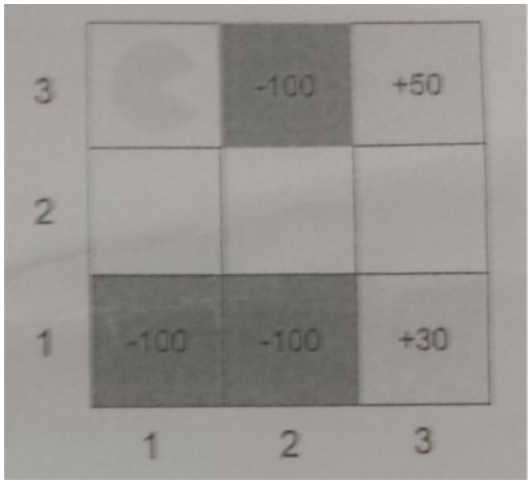
a) ii)

0	0	1	1
0	////////////////////////////////	0	-1
0	0	0	0

b) iii)

Right	Right	Right	Right
Up	////////////////////////////////	Up	Right
Right	Right	Right	Right

Question 2 --- Monte Carlo Reinforcement Learning (4 Pts)



Consider grid-world game given below, with an agent trying to learn the optimal policy. At each shaded grid cell we place a box containing either a reward or a penalty. These items are only revealed to the agent when it takes the Open action from one of the shaded states, upon which the game terminates (T). This is an undiscounted game.

The agent starts from the top left corner and you are given the following episodes from runs of the agent through this grid-world. Each line in an Episode is a tuple containing (s, a, s', r).

Each action a € {Up, Down, Left, Right, Open}. Each state s = (z, y) representated by its horizontal (x) and vertical (y) coordinates.

Episode 1	**Episode 2**	**Episode 3**	**Episode 4**	**Episode 5**
(1,3), ↓, (1,2), 0	(1,3), ↓, (1,2), 0	(1,3), ↓, (1,2), 0	(1,3), ↓, (1,2), 0	(1,3), ↓, (1,2), 0
(1,2), →, (2,2), 0	(1,2), →, (2,2), 0	(1,2), →, (2,2), 0	(1,2), →, (2,2), 0	(1,2), →, (2,2), 0
(2,2), →, (3,2), 0	(2,2), ↓, (2,1), 0	(2,2), →, (3,2), 0	(2,2), →, (3,2), 0	(2,2), →, (3,2), 0
(3,2), ↑, (3,3), 0	(2,1), Open, T, -100	(3,2), ↓, (3,1), 0	(3,2), ↑, (3,3), 0	(3,2), ↓, (3,1), 0
(3,3), Open, T, +50		(3,1), Open, T, +30	(3,3), Open, T, +50	(3,1), Open, T, +30

Fill in, with detailed calculations, the following Q-Values obtained from direct evaluation from the samples:

Q((3,2), ↑) = Q((3,2), ↓) = Q((2,2), →) =

Question 3 --- Q-Learning (6 Pts)

(Using the problem given in Question 2) Consider Q-Learning algorithm with linear function approximation:

$$Q(s,a) = \omega_1\phi_1(s,a) + \omega_2\phi_2(s,a) + \omega_3\phi_3(s,a)$$

where we design the features as $\phi_1(s,a)$ =x-coordinate of state s; $\phi_2(s,a)$ =y-coordinate of state s; and $\phi_3(s,a) = f(a)$with $f(\uparrow) = 1, f(\downarrow) = 2, f(Open) = 3, f(\rightarrow) = 4, f(\leftarrow) = 5.$

- (4 Pts) All ω_i are initialized to 0. What are their values after the first episode given in question (a)? Also explain your solution briefly. Assume the learning rate $\alpha = \frac{2}{3}$ for all calculations.
  $\omega_1 =$ $\omega_2 =$ $\omega_3 =$
- (2 Pts) Assume the weight vector ω is equal to (1,1,1). What is the best action prescribed by the Q-function in state (2,2) ?

Solutions:

i) $\omega_1 = \omega_2 = \omega_3 = 100$

ii) left

Question 4 --- Q-Learning (4 Pts)

Question 4 — Q-Learning (4Pts)

Q-learning's standard update equation is:

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha(r_{t+1} + \gamma \max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t))$$

where γ is the discount factor, α is the learning rate and the sequence of observations are $(\cdots, s_t, a_t, r_{t+1}, s_{t+1}, \cdots)$. Here we look at a 1-step window, $(s_t, a_t, r_{t+1}, s_{t+1})$, to update the Q-values. One can think of using an update rule that uses a larger window to update these values. Give one possible update rule for $Q(s_t, a_t)$ using 2-step window, $(s_t, a_t, r_{t+1}, s_{t+1}, a_{t+1}, r_{t+2}, s_{t+2})$, which might employ $Q(s_{t+1}, \cdot)$ and/or $Q(s_{t+2}, \cdot)$.

$Q(s_t, a_t) =$

$Q(s_t, a_t) =$

$Q(s_t, a_t) =$

Question 5 --- General Topic (4 Pts)

Thinking about Reinforcement Learning which ones of the following statements are true (multiple choice):

(a) The maximization of the future cumulative reward allows to Reinforcement Learning to perform global decisions with local information

(b) Q-learning is a temporal difference RL method that does not need a model of the task to learn the action value function

(c) Reinforcement Learning only can be applied to problems with a finite number of states

(d) In Markov Decision Problems (MDP) the future actions from a state depend on the previous states

Thinking about reinforcement learning which one of the following statements is true (only one):

(a) Estimation using Dynamic Programming is less computational costly than using Temporal Difference Learning

(b) Estimating using Montecarlo methods has the advantage that it is not needed to have absorbent states in the problem

(c) Temporal Difference learning allows on-line learning and Montecarlo methods need complete training sequences for estimation

(d) Dynamic Programming and Montecarlo methods only work if we know the transitions probabilities for the actions and the reward function

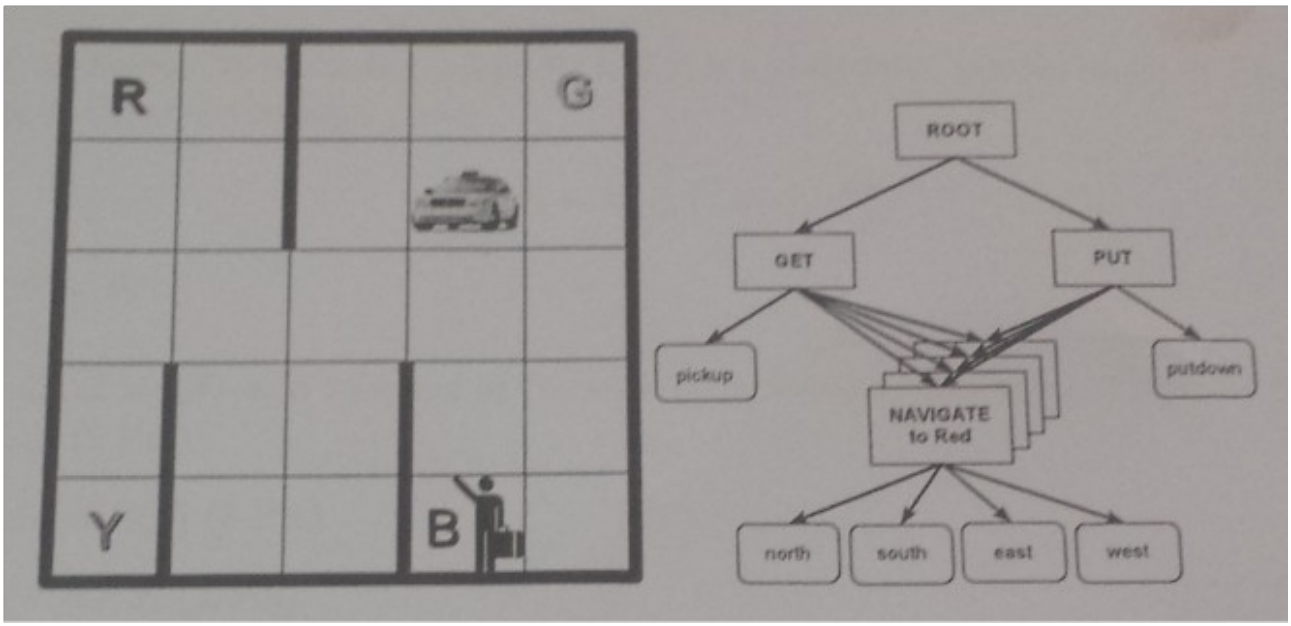
SOLUTIONS:

i) a+b

ii) c

Question 6 --- Hierarchical Reinforcement Learning (6 PTS)

Given the Taxi domain and an action hierarchy as in the map. Assuming that the rewards of the movement {left, right, up, down, pickup, putdown} are -1. if the putdown is successful, the agent receives and additional reward of 100. If the pickup and putdown are at wrong locations the agent receives a reward of -10.



Assuming that the initial state (s_0) si given in the map, from that the agent executes a sample trajectory:

Root, Get{(Navigate_to_Blue),Pickup}, Put{(Navigate_to_Red), Putdown}

where Navigate_to_Blue = {down, down, down}, and Navigate_to_Red = {up, up, right, right, right, up, up}. Based on the above data, you are asked to estimate the following terms w.r.t. the current policy of the agent:

- The completion term C(Root, s_0, Get) =
- The completion term C(Get, s_0, Navigate_to_Blue) =
- The completion term C(Navigate_to_Blue, s_0, down) =
- The reward term V(Down, s_0) =
- The reward term V(Root, s_0) =

PART OF A SOLUTION:

V(Root, s_0) = -3 + (-1) + (-7) + (-10) = -21

Question 7 --- Policy Gradient (10 Pts)

Question 7 — Policy Gradient (10Pts)

For policy gradient algorithms, the objective function is

$$J(\theta) = \int p(\tau;\theta)R(\tau)d\tau$$

where $\theta \in R^m$ is the parameter of the policy $\pi(a|s;\theta)$, and τ is a trajectory $\{s_0, a_0, r_0, s_1, a_1, r_1, \ldots\}$. Assuming that the discount factor is γ. Given a Gaussian policy

$$\pi(a|s;\theta) = \mathcal{N}\left(a; \theta^\top \phi(s), \sigma^2\right)$$

where $a \in \Re$, and $\phi : \mathcal{S} \mapsto R^m$.

- Write the formula of $p(\tau;\theta)$ as a function of the transition function $T(s'|s,a)$, $\pi(a|s,\theta)$, and the starting sta distribution $p_0(s)$. (2 Pts)
- Write the formula of $R(\tau)$ (2 Pts)
- Compute $\nabla_\theta \log \pi(a|s;\theta)$ (2 Pts)
- Given a data set $\mathcal{D} = \{\tau_i\}_{i=1}^M$, where $\tau_i = \{s_0^{[i]}, a_0^{[i]}, r_0^{[i]}\, s_1^{[i]}, a_1^{[i]}, r_1^{[i]}, \ldots, s_T^{[i]}\}$ generated from the policy at iteration, $\pi(a|s;\theta_k)$, what is the update of θ at the iteration $k+1$: θ_{k+1}, using Monte-Carlo estimation? (4

Question 8 --- G(PO)MDP (8 Pts)

Question 8 — G(PO)MDP (8Pts)

The intuition behind the G(PO)MDP algorithm is *the rewards at a given time are independent from future actions*. In other words,

$$\mathbb{E}\left[\nabla_\theta \log \pi(a_t|s_t;\theta) r_j\right] = 0, \forall j < t$$

- Prove the above property of G(PO)MDP (4)
- Derive the update for the baseline of G(PO)MDP (4)