Multi-Agent Systems

Final Homework Assignment

MSc AI, VU

E.J. Pauwels

Version: December 7, 2020— **Deadline: Friday 8 Jan 2020 (23h59)**

IMPORTANT

- This project is an **individual assignment.** So everyone should hand in their own copy.
- Both of the questions below require programming. In addition to the report (addressing the questions and discussing the results of your experiments), also provide the code on which your results are based (e.g. as Python notebooks). However, make sure that the report is self-contained, i.e. that it can be read and understood without the need to refer to the code. Store the pdf-report and the code in a zipped folder that you upload via canvas.
- This assignment will be **graded.** The max score is 4 and will count towards your final grade.
- Your **final grade (on 10)** will be computed as follows:

Assignments 1 thru 5 (max 1) + Individual assignment (max 4) + Final exam (max 5)

1 Monte Carlo Tree Search (MCTS)

Construct a binary tree (each node has two child nodes) of depth $d = 12$ (or more – if you're feeling lucky) and assign different values to each of the 2^d leaf-nodes. Specifically, pick the leaf values to be *real numbers* randomly distributed between 0 and 100 (use the uniform continuous distribution $U(0, 100)$, so don't restrict yourself to integer values!).

- Implement the MCTS algorithm and apply it to the above tree to search for the optimal (i.e. highest) value.
- Collect statistics on the performance and discuss the role of the hyperparameter c in the UCB-score.

Assume that the number MCTS-iterations starting in a specific root node is limited (e.g. to 10 or 50). Make a similar assumption for the number of roll-outs starting in a particular ("snowcap") leaf node (e.g. 1 or 5).

2 Reinforcement Learning: SARSA and Q-Learning for Gridworld

Consider the 9×9 gridworld example depicted in the figure below. The blue gridcells represent walls that cannot be traversed. The green cell represent a treasure and transition to this cell yields a reward of $+50$ whereupon the episode is terminated (i.e. absorbing state). The red cell represents the snakepit: this state is also absorbing and entering it yields a negative reward of -50. All other cells represent regular states that are accessible to the agent. In each cell, the agent can take four actions: move north, east, south or west. These actions result in a deterministic transition to the corresponding neighbouring cell. An action that makes the agent bump into a wall or the grid-borders, leaves its state unchanged. All non-terminal transitions (including running into walls or grid borders) incur a negative reward ("cost") of -1.

For the questions below, we assume that the agent is not aware of all the above information and needs to discover it by interacting with the environment (i.e. model-free setting). Perform, and comment on, the following experiments:

- Use Monte Carlo policy evaluation to compute the state value function $v_\pi(s)$ for the **equiprobable policy** π (i.e. all 4 actions have probability $1/4$). Visualize the result (e.g. by using some sort of heat map where the color of each cell corresponds to the state value).
- Use SARSA in combination with greedification to search for an optimal policy.
- Use Q-learning to search for an optimal policy. Compare to the solution obtained by SARSA.