Reinforcement Learning: Coursework Assignment 1 (*Semester 2, 2015/2016*)

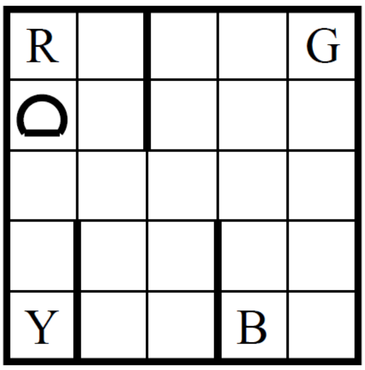
Instructions

- This homework assignment is to be done *individually*, without help from your classmates or others (except your RL tutor). Plagiarism will be dealt with strictly as per University policy.
- Solve all problems and provide your complete solutions (with adequate reasoning behind each step) in a report in computer-printed or *legibly* handwritten form.
- Before you start to write a program, read all questions below carefully.
- Only the report will be marked. The code itself will not be marked, but may be used to clarify any questions arising from the report. If you are using code that you have not written yourself, then acknowledge this appropriately (you do not have to mention the code provided for the tutorials). Include references to books and papers that you have been using (you do not have to cite the lecture slides for this course).
- Use graphical representations wherever suitable. If you use numerical output for demonstrating your results, make sure that the numbers are appropriately rounded and presented in an accessible way. If your problem involves randomness, explain why your result is representative for most of the possible realisations of the underlying random processes.
- Include your code in the submission (preferably Matlab). If you are presenting numerical results in your report, specify all major numerical parameters that were involved in the generation of the data shown.
- This assignment will count for 10% of your final course mark.
- Please submit your assignment by 4 pm on 11th February as a paper copy to ITO as well as an electronic version (including your code) via the submit system (directory "rl").

Questions

This assignment studies a benchmark problem for RL algorithms, namely the taxi problem (Dietterich, 2000). Below is the specification given in Ref. (Dutech, 2005)

- State variables: taxiLocation {1, ..., 25}, passengerLocation {1, ..., 5} (i.e. waiting at pickup/drop-off {R,G,B,Y} or in the taxi), drop-offLocation {1, ..., 4} (i.e. {R,G,B,Y}).
- Initialisation of a trail: Taxi is uniformly randomly in any of the 25 grid squares, passengerLocation is uniformly randomly in one of the 5 passenger states, dropoffLocation is uniformly randomly one of the 4 drop-off locations
- Termination of a trial: Passenger was successfully dropped-off or after a time constraint (passenger just wants to get out off the taxi and does not care where)
- Actions: 1: go north, 2: go south, 3: go west, 4: go east, 5: pick up passenger, 6: drop off passenger
- Reward is 0, except in the following cases: -1 for an unsuccessful movement (e.g. if blocked by a wall), 1 for a successful pick-up, 10/(number of steps since pick-up) for a successful drop-off, -1 for an attempted drop-off with no passenger or at the wrong location, -1 for an attempted pick-up at the wrong location (or if the passenger is already in the taxi).



1. Before you start programming, provide some rough estimates the following quantities
    - typical time horizon of the problem
    - mean (immediate) reward for a good policy
    - maximal value of a state-action pair
    - number of trials that a standard algorithm will need in order to find a good solution

   Explain how you arrive at the estimates and how these estimates can be used in the design of the algorithm. You should not use a computer for this first subquestion, marks will be given for a good explanation, not for precision. Any numbers that do not ruin the success of the following tasks are fine. Discuss also whether the problem is deterministic or stochastic. (10/100)
2. Solve the problem using *Q*-learning (Now using a computer program!).
    - Use graphical representations (e.g. averaged reward over learning time) to show the performance of your algorithm for random starting and random goal positions. Compare the results obtained now with the estimates for mean reward and maximal value that you have obtained in question 1. (10/100)
    - Represent your solution using an example (Fix e.g. initial passenger position at "Y", goal at “B”, and plot the value of the spatial states and the best action for each state). (10/100)
    - Consider the time course of the values and the reward and define a convergence time for the algorithm. (10/100)
    - Discuss, possibly using numerical examples, how the convergence time depends on the exploration strategy and other parameters of the algorithm. (20/100)
3. Solve the problem using SARSA and answer the same questions as above. (20/100)
4. Compare the performance of *Q*-learning and SARSA on the present version of the Taxi problem. (10/100)
5. Taking into account the compositional structure of the problem, discuss how the learning speed can be improved? Consider that partial policy followed by the cab is essentially the same when moving, e.g., towards pick-up site "R" in order to pick up a passenger and when doing this in order to drop her off at site "R". Explain how you would take this structure into account. (10/100)
6. Add a paragraph of conclusions drawn from the solution of these problems.

References:

Dietterich, T. G. (2000). Hierarchical reinforcement learning with the MAXQ value function decomposition. *J. Artif. Intell. Res. (JAIR)*, *13*, 227-303.
Dutech, A., Edmunds, T., Kok, J., Lagoudakis, M., Littman, M., Riedmiller, M., Whiteson, S. (2005) Reinforcement learning benchmarks and bake-offs II. *Workshop at Advances in Neural Information Processing Systems conference*.