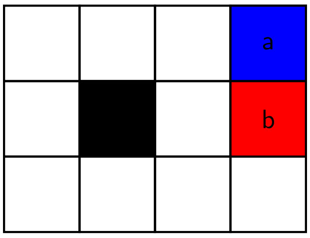
COMP3702/COMP7702 Artificial Intelligence
Semester 2, 2020
Tutorial 11

Example domain

Consider the gridworld below, which we saw in Tutorial 9:



States in this environment are the positions on the tiles. The world is bounded by a boundary wall, and there is one obstacle, at [1,1] (using python or zero indexing starting from the bottom left corner). In addition, there are two terminal states, indicated by the coloured squares.

Actions and Transitions: In this world, an agent can generally choose to move in four directions — *up*, *down*, *left* and *right* However, the agent moves successfully with only $p = 0.8$, and moves perpendicular to its chosen direction with $p = 0.1$ in each perpendicular direction. If it hits a wall or obstacle, the agent stays where it is (i.e. no collision cost). In addition, once the agent arrives on a coloured square with a value, it has only one special action available to it; that is, to *exit* the environment.

Rewards: The values stated on the coloured squares are the reward for *exiting* the square and the environment, so the reward is not repeatedly earned; that is, $R([3, 2], exit) = a$, and $R([3, 1], exit) = b$. All other states have 0 reward.

Discount factor: $\gamma = 0.9$.

Exercise 11.1.

a) List the dimensions of complexity for a reinforcement learning environment: Modularity, Planning horizon, Representation, Computational limits, Learning, Sensing uncertainty, Effect uncertainty, Preference, Number of agents, Interaction.

b) In RL, what is the connection between the environment's state transition function and the exploration policy used by the agent?

Exercise 11.2. For the grid world above, develop a simulator for the state-action-reward-state loop.

- In particular, you should develop a `Grid.apply_move()` function that takes the current state and actions as inputs and updates the state while returning the reward for the state-action-state transition.
- The provided code includes `Grid.player_x` and `Grid.player_y` variables that represent the state of the player on the (x, y) grid.
- You should make use of the provided code, especially the `stoch_action()` and `attempt_move()` methods.
- Include a way to restart the agent in a random map location, avoiding obstacles, after it exits the environment.

(Note: you may wish to look at how the Laser Tank environment's `apply_move()` method works, as provided in the Assignment 4 support code).

Exercise 11.3. Using your simulator, implement Q-learning for this gridworld problem.

a) First, write a function to choose an action given the Q-value estimates. Incorporate your agent's exploration strategy in this function.

b) Then write a `next_iteration()` function, which will probe the simulator to receive a reward and a new state, and use this to update your agent's Q-values.

- Be sure to include an exploration strategy
- See the provided code.