Queen Mary, University of London
School of Electronic Engineering and Computer Science

ECS7002P: AI in Games

Reinforcement Learning

Group-AK

Tarek Uddin Ahmed	170431981
Mughees Asif	180288337

Due: 10 January 2022

Q1: Explain how your code for this assignment is organised.

1. The code was written using `Python3` syntax, is fully documented and commented to increase reusability, and contained within the `code` folder with the following structure:
    - **`1_environment`**: Defines the `FrozenLake` environment.
    - **`2_tabular_model_based_rl`**: Contains the `policy_evaluation`, `policy_improvement`, `policy_iteration`, and `value_iteration` methods.
    - **`3_tabular_model_free_rl`**: Outlines the ϵ-greedy algorithm, `SARSA`, and Q-learning algorithms.
    - **`4_non_tabular_model_free_rl`**: Contains the `LinearWrapper` class to emulate the environment, the `linear_SARSA`, and `linear_q_learning` methods.
    - **`5_main`**: Main method that defines the controlling parameters and executes all the learning algorithms for the *small* `FrozenLake` environment.
    - **`6_main_big_lake`**: Main method for the *big* `FrozenLake` environment.
2. The code was developed using the exact instructions from the assignment manual without added deviation. Additionally, generic code snippets from online resources and the official laboratory exercises were used to develop the computational solutions.
3. Different parameters can be modulated within the `./code/5_main.py` directory. The same file can then be run to observe the changes in the learning trajectories.
4. The output is contained within the `output.txt` file located in the root folder.

Q2: How many iterations did policy iteration require to find an optimal policy for the big frozen lake? How many iterations did value iteration require? Which algorithm was faster?

The policy iteration and value iterations methods took 6 and 10 iterations to find the optimal policy, respectively. Therefore, the policy iteration algorithm was faster.

Q3: How many episodes did `SARSA` control require to find an optimal policy for the small frozen lake? How many episodes did Q-learning control require?

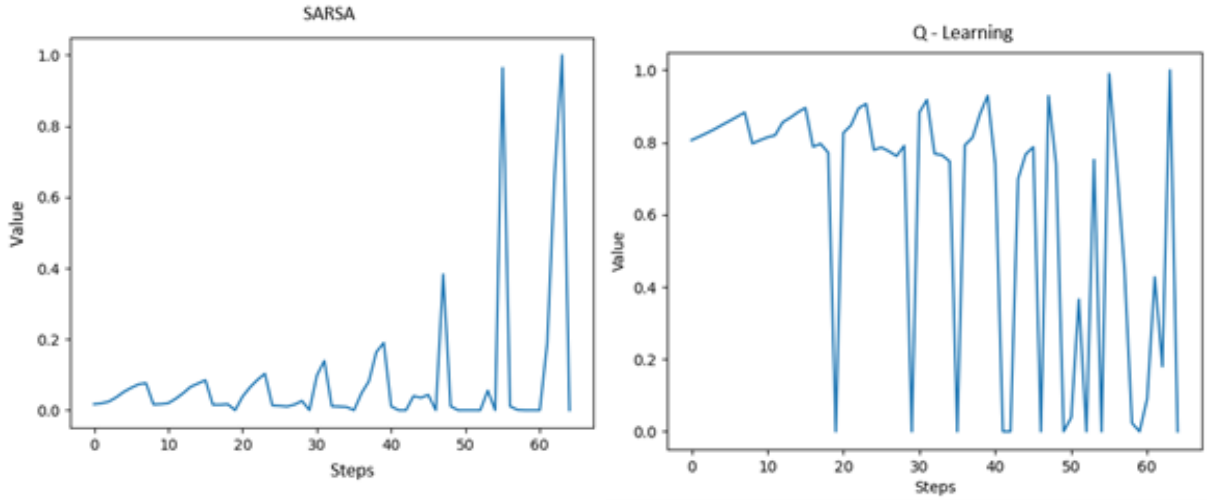
Policy Evaluation is an iterative algorithm used to evaluate how good a policy is and would eventually converge to V^{π}, where V^{π} is the state value function and would be the actual value of the policy after convergence. Q-learning control took 750 episodes, whereas the `SARSA` control took 2200 episodes to find the optimal policy. This suggests that Q-learning control is a faster iterating algorithm when compared to `SARSA` control.

Q4: In linear action-value function approximation, how can each element of the parameter vector θ be interpreted when each possible pair of state s and action a is represented by a different feature vector $\varphi(s, a)$ where all elements except one are zero? Explain why the tabular model-free reinforcement learning algorithms that you implemented are a special case of the non-tabular model-free reinforcement learning algorithms that you implemented.

Linear action-value function approximation uses stochastic gradient descent to approximate action-value functions. The parameter vector θ is interpreted when computing the value for each action in a specific state. The value of each action in a specific state is calculated by computing the inner product between the feature vector $\varphi(s, a)$ of an action for a specific state and the current parameter vector θ to update the relevant weights where all elements except one are zero. This means that irrelevant state-action pairs would not be affected when the parameter vector is updated during training for each iteration. The feature vector $\varphi(s, a)$ for all state-action pairs would need to be available before computing the inner product.

The non-tabular model-free reinforcement learning algorithm encodes the current state, where the state-action pairs can then be retrieved for the state using a decode policy without storing any values to the memory, whereas the tabular model-free Reinforcement Learning algorithms are a special case of the non-tabular reinforcement learning algorithms, where the encoded values are stored in memory, in the form of a table which is created in the beginning eliminating iterative calculations.

Q5: Try to find an optimal policy for the big frozen lake by tweaking the parameters for `SARSA` control and Q-learning control (maximum number of episodes, learning rate, and exploration factor).



(a) `SARSA` control

(b) Q-learning

Figure 1: Iterative policy values

When using the same parameters as the small frozen lake for the big `FrozenLake`, the values produced are zero. This may be due to the algorithm not being able to construct a path to the goal state. Since, the environment is larger than the small `FrozenLake` environment, the number of iterations may not be sufficient, and the discount factor may be decaying rewards faster resulting in the algorithm failing to learn from the environment.

When increasing the max_step, discount factor and max_iterations, the values near the goal state are populated, whereas the values near the start state are still zero. This suggests that the standard computation budget is not sufficient to use for the big environment resulting in values being produced near the goal state but not at and near the start state.

Decreasing the exploration factor alone has no effect on the results obtained. This may be due to excessive exploitation leading to poor long-term performance. Increasing the exploration factor alone also makes no difference in the results. This may be due to excessive exploration leading to poor short-term performance. Modifying the learning rate and the maximum number of episodes also showed no signs of improvement unless the number of iterations and max_step was modified.

The optimal policy for the big `FrozenLake` could not be found by tweaking the parameters for `SARSA` control and Q-learning control alone. However, results show that the algorithm can produce values by increasing the max_steps, max_iterations, max_episodes and discount factor parameters. Taking the experiment further, by increasing the learning rate and exploration factor, both the `SARSA` control and Q-learning control algorithms can produce a path from the starting state to the goal state.