Reinforcement Learning: An Introduction
Attempted Solutions
Chapter 1

Scott Brownlie & Rafael Rui

1 Exercise 1.1: Self-Play

Suppose, instead of playing against a random opponent, the reinforcement learning algorithm described above played against itself, with both sides learning. What do you think would happen in this case? Would it learn a different policy for selecting moves?

Suppose that player X discovers a good board state, such as one with two Xs on the top row. Then she will start to make moves that get her into this state more often. But player O will also experience this state more often and learn that it is bad for her, and thus begin to make moves which help her avoid this state. Player X would then have to adapt and find more elaborate ways to arrive at the winning board states. The exact same learning process would occur for player O and both players would gradually become more and more intelligent at the same rate. In the long run they would both learn very similar value functions, that is, the value of a state for player X should be the same as the value of the opposite state for player O, where opposite means switching the Xs and Os.

2 Exercise 1.2: Symmetries

Many tic-tac-toe positions appear different but are really the same because of symmetries. How might we amend the learning process described above to take advantage of this? In what ways would this change improve the learning process? Now think again. Suppose the opponent did not take advantage of symmetries. In that case, should we? Is it true, then, that symmetrically equivalent positions should necessarily have the same value?

Instead of learning the value function for all board states we could learn the value function only for unique states after removing symmetries. So, for example, the state with an X in the top left corner and a O in the bottom right corner would have the same value as the state with a O in the top left corner and an X in the bottom right corner. This would significantly reduce the number of states, meaning that each state would be visited more often and the value function would converge in less time steps (assuming that the step-size parameter is reduced properly over time).

If the opponent did not take advantage of symmetries then her value function would take longer to converge. This may actually be beneficial due to increased exploration. Suppose that player X, who is taking advantage of symmetries, got lucky and won from a bad board state during one of the early games. Then this state would be reinforced along with all symmetrical state, making it even more likely to be experienced again. As player O is not taking advantage of symmetries and learning more slowly, player X may get lucky several more times and continue to reinforce the bad state. Player O could also learn different strategies from symmetrically equivalent states, whereas player X would be stuck with the same strategy for all equivalent states, irrespective of the opponent's strategy. This could lead to a sub-optimal policy for player X.

As the value of a state depends on the opponent's strategy, symmetrically equivalent states should only have the same value if the opponent's strategy is the same for those symmetrically equivalent states. This should be true for skilled human opponents, but when the reinforcement learning algorithm plays against itself there is no guarantee of this, especially in the earlier stages of learning.

3 Exercise 1.3: Greedy Plan

Suppose the reinforcement learning player was greedy, that is, it always played the move that brought it to the position that it rated the best. Might it learn to play better, or worse, than a nongreedy player? What problems might occur?

It would likely play worse than a nongreedy player. To begin with, all non-terminal states have value 0.5, so the greedy player has to select moves randomly. She could get lucky and win from a bad board state, is which case the move would be reinforced and the player would be locked into that move whenever she encountered the possibility of moving into the same state again. Alternatively, the greedy player could be unlucky and lose from a good board state, in which case the state's value would be reduced and no other move which led to that state would ever be chosen again.

4 Exercise 1.4: Learning from Exploration

Suppose learning updates occurred after all moves, including exploratory moves. If the step-size parameter is appropriately reduced over time (but not the tendency to explore), then the state values would converge to a different set of probabilities. What (conceptually) are the two sets of probabilities computed when we do, and when we do not, learn from exploratory moves? Assuming that we do continue to make exploratory moves, which set of probabilities might be better to learn? Which would result in more wins?

When we do not learn from exploratory moves the value of each state converges to the true probability of winning from that state, assuming that after convergence we always move greedily. When we do learn from exploratory moves the value of each state converges to the true probability of winning from that state, assuming that after convergence we continue to make exploratory moves. For example, consider a board state with two empty spaces (and thus two possible moves) such that player X wins if he chooses one of the moves and loses if he chooses the other. When we do not learn form exploratory moves the value of this

Thus, assuming that we do continue to make exploratory moves, the latter set of probabilities would be better to learn and would result in more wins.

5 Exercise 1.5: Other Improvements

Can you think of other ways to improve the reinforcement learning player? Can you think of any better way to solve the tic-tac-toe problem as posed?

Instead of backing-up the value of the next state to the previous state only, we could back-up the value to two or more time steps before, perhaps with some discount factor which gives less weight to states further back. In the extreme we could back-up the value to all previous states in the game. This should not change the final probabilities, but would increase the speed of learning.