Chapter 1 - Solutions

Alessio Caciagli

March 2021

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?*

If self-play were adopted as a learning strategy, the optimal policy for selecting moves should be the same that would be achieved by playing against a (human) opponent. This is guaranteed by the observation that a RL method as employed in tic-tac-toe would converge to the true probability of winning from each state played by an opponent, be it human or itself.
The advantage provided by self-play is the faster convergence - the algorithm will explore a higher number of states in a shorter time. This stems from two main reasons. The first, and most obvious, reason is that self-play can simulate games much faster than a human. The second, and more subtle, reason is that self-play ensures that the environment is always the "right difficulty" for an AI to improve. This means that the balance between exploratory and exploitative moves is better during self-play: the algorithm would not attempt "silly" exploratory moves when playing against a strong opponent (read, stronger version of itself) as it has already explored them when playing against a weak opponent (read, weaker version of itself). This has the clear advantage of speeding up convergence as well as ensuring that a higher number of overall states are in fact explored during any stage of the learning process.

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?*

In tic-tac-toe, one could use the 4 axes of symmetry present in a square (the horizontal and vertical directives, plus the two diagonal directives) to define the

grid and occupancy rules. This would result in a dramatic boost of the speed of convergence since it drastically reduces the number of possible states. It would also result in a much smaller memory footprint, since our environment could be parameterized by a smaller state vector than using a full cartesian grid, similar to how a circle can be conveniently parameterized in polar coordinates (using one parameter less than using full cartesian coordinates).
If our opponent did not take advantage of symmetries, we still should! In its essence, symmetries just allow us to explore less states, since many positions are themselves the same. Symmetrically equivalent positions should then necessarily have the same values, because the probability of winning from each symmetric state is exactly the same, given optimal play by the opponent. And even if the opponent did not play optimally (as in this case, not using symmetries), our algorithm still should!

3 Exercise 1.3

Greedy Play *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?*

If the RL player only played greedy moves, it would overall play worse than a nongreedy player. Although greedy moves will always yield the greatest value, only playing them would reject the trade-off between exploration and exploitation, a paramount principle in Reinforcement Learning. Only exploiting experienced actions (that is, playing greedily) will lead to sub-optimal play as better action selections are never discovered through exploration (which a nongreedy player would normally do).

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?*

Under the assumptions posed in the question, the set of probabilities computed when we do not learn from exploratory moves will converge to the true probabilities of winning from each state given optimal play. On the other side, the set of probabilities computed when we do learn from exploratory moves will, instead, converge to the probabilities of winning assuming sub-optimal play, where 'sub-optimal' is here quantified by our exploration policy itself (for example, by playing a random move with probability ϵ). The latter is worse

and will lead to less wins.
As a proof, suppose you are playing a game of chess, and you can checkmate your opponent in your next move. Under the exploration policy, you happen to play a different (random) move. If we were learning from exploratory moves, we would be updating the previous state's value (which should be 1, given that you win the game in your next move) to a lower value, because your opponent's next move might not lead to checkmate anymore. But this would, at the end, lead to an algorithm playing worse (or learning slower) than if we were not learning from exploratory moves.

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?*

Tic-tac-toe has a small state set, therefore the game is solvable directly (using recursion and dynamic programming, for instance). Discarding this improvement as obvious, we could still tabulate the opening and end-game moves (similarly to the game of chess). In practical terms, this would translate to different, better initial values of the states, which in turn would speed up convergence of the learning algorithm.
Another idea is to become aware of our opponent's play, and vary the learning rate or the exploration rate based on the variance of the opponent's moves. This would avoid slow learning because of local, sub-optimal maxima in the algorithm's convergence. For instance, if the opponent were always making the same moves, a fixed exploratory policy would end up in a slower convergence because we would be 'stuck' playing mostly the same configurations. If we increase the probability of performing an exploratory move, thus adapting to our opponent's low variance in his/her playing style, would on the other side explore more states and improve the RL player.