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?

Value function update rule for each of the competing agents:

$$V_o^t \leftarrow V_o^t + \alpha\left(V_o^{t+1} - V_o^t\right)$$
$$V_x^t \leftarrow V_x^t + \alpha\left(V_x^{t+1} - V_x^t\right)$$

$$V_?^{END} = \begin{cases} +1 & \text{win} \\ 0 & \text{tie} \\ -1 & \text{loss} \end{cases}$$

$V_o^t = V_o(S_t)$ ... Value fcn. of O player

$V_x^t = V_x(S_t)$ .. Value fcn. of X player

When both sides learn they exploit each other's weaknesses caused by different random initializations of their own policies. Learning will eventually converge to the optimal minimax agent on both sides.

$$V_0(S_0) = \max_{S_0} V_o^0 \max_{S_1} V_X^1 \max_{S_2} V_o^2 \ldots \max_{S_N} V_?^N$$

$$\begin{cases} +1 & \text{win} \\ 0 & \text{tie (final value)} \\ -1 & \text{loss} \end{cases}$$

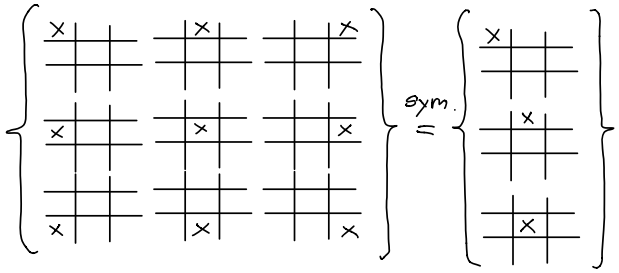
Previous values receive the update delayed.

Initially this value is 50%. but over time and episodes it learns the actual value of the state.

Formal proof can be made by contradiction. Assume that one of the players converges to some sub-optimal policy. The opponent should eventually explore and then exploit this weakness. Exploitation makes the sub-optimal player adjust it's policy causing a contradiction with the convergence to a sub-optimal behavior. Therefore the converged policies must be optimal.

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?

Tic-Tac-Toe has a natural mirror- rotational symmetry. By taking advantage of this the learning process can be accelerated because the learning process will visit a particular symmetrically equivalent state much more frequently and will need to learn values of smaller overall count of states.



- X player starts (without any loss of generality)
- 9 possible moves can be reduced to 3 symmetric
- Symetry operation is ±90° or 180° rotation and mirroring

When the opponent doesn't play symmetrically the agent shouldn't either. Symmetric value function isn't able to tell apart the symmetric variants and react differently in each scenario. It's unable to exploit the sub-optimal play happening in one of the four symmetric situations.

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? ☑

Constantly greedy agent won't be able to play better than an exploring one. Thanks to the greed it isn't able to advance beyond it's current actions and explore for potentially more rewarding actions over a longer horizon.

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? ☑

In case when the step size parameter is reduced but the agent still explores the learned value function would be a lower than for an equivalent non-exploring policy. The lower value reflects the fact that in some fraction of states the exploring agent makes the wrong non-optimal move.

Example is a "rope walk" scenario when the agent needs to closely follow some narrow path. The exploring agent will occasionally step into the "void" with the intent of finding a better reward. This leads to worse performance than an agent trained with attenuated exploration.

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?

having a dataset of reasonably good human play could potentially help to jumpstart the learning process by providing state transitions of commonly used strategies. These transitions can be used to initialize the value function without the need for potentially lengthy and computationally expensive exploration of the state space. Good dataset can be obtained from a classical minimax AI self-play.