Ben Goldstein — Gin Rummy and Tonic

A Summary of Monte-Carlo Counterfactual Regret

The Counterfactual Regret (CFR) minimization algorithm is a method used to numerically find a Nash Equilibrium in a game of incomplete information. Intuitively, in a human player, this would involve thinking back on past actions in a game and wondering “Well, what would have happened if I had acted differently?”. This would allow the player to find decisions that they regretted making, and decide what would be a better course of action if a similar decision ever came up in the future. Truth be told, the CFR minimization algorithm is much simpler than what goes on in the human mind, but it still will eventually lead to a Nash Equilibrium.

In Monte-Carlo CFR (MCCFR), when the player who is training is at a decision point, they set the probability for each potential action to some ε (usually 0.6), divided by the number of actions, plus the probability that they would usually take that action weighted by 1 – ε. That is, with probability ε, they will sample uniformly from these actions, and with probability 1 - ε they will choose based off of their regret matching strategy. This allows the player to randomly select from the actions they can take, choosing from more likely options more often. After weighting the probabilities, the player selects a random action from their list of available actions. If the current node is not terminal, then they calculate the counterfactual utility as the counterfactual utility received from recursively repeating the entire process for the child node resulting from the selected action; however, if the current node is terminal, then they calculate the counterfactual utility for the node as its utility divided by the probability that they played to this turn.

Then, the player updates their total regret for each possible action. For the action that was chosen, the player adds to the infoset’s total regret the action’s utility, weighted by 1 – the probability that they took that action, times the probability that they played to the terminal node that they reached in the recursion. However, for each action that wasn’t chosen, the player subtracts from the infoset’s total regret the action’s utility weighted by the probability that they took that action, times the probability that they played to the terminal node that they reached in the recursion. Coming out of the recursion, the player updates the probability that they played to the terminal node that they reached in the recursion by the probability that they chose that action, and returns both the node’s counterfactual utility, and the probability that they played to said terminal node.

While this is all going on for the player that’s training, the player who isn’t training updates the average strategy instead of the regret. The player mostly performs the same process as above, with a few differences. Rather than sampling uniformly with probability ε and choosing based off of the average strategy with probability 1 – ε, they always choose based off of the latter. Additionally, there is no weighting in the updating of the strategy like in updating the regret. Rather, for each possible action, the player adds to the strategy for the infoset the probability they chose that action, divided by the probability that they played to the selected terminal node.

Eventually, the above process will converge to a Nash Equilibrium for the given game.