Mark Paluta

DMU Project 2: Reinforcement Learning

Introduction

This report outlines a strategy for reinforcement learning in three examples of Markov Decision Processes (MDPs). Sampled data from the three MDPs was provided, but no further direct sampling was possible. A Q learning algorithm in a model free environment was used to generate policies to attempt to maximize reward in the three cases.

Algorithm

Modified Q learning was implemented. The largest change in structure from standard Q learning was that exploration was not possible without the opportunity to sample more or a model in place, and thus the entire policy had to come from already known data. The algorithm iterated down each line of the data set—state, action, next state, and reward—and stored a utility value in a matrix for the appropriate state-action pairing. Thus, after one single time step, the reward matrix is very sparse. The algorithm then repeats this process, stepping through time and using the existing knowledge of the Q matrix, which encourage actions that lead to delayed rewards. Due to the limited data provided and lack of model, this algorithm is not guaranteed convergence. Had a model been implemented, the algorithm would only be guaranteed convergence to that specific model, not the real model used to generate the data.

Once the Q matrix has been generated and appears relatively stable, or after a desired number of time steps, the policy must be extracted. This is done by finding the value of highest utility of all possible actions given one state. In matrix form, this means iterating by column (states) and finding the row index (action) that provides the maximum value of that column using an argmax function. This resulting vector of row indices is the policy since it is a vector one action per state that has been shown to maximize future utility.

Policies

For the small MDP problem, the optimal policy was found. This is known because the score for an optimal policy was provided for reference (52971). The policy involves moving around in grid world in a direction toward the single square that offers reward and then staying put once that square is reached.

For the medium problem, the biggest flaw of Q learning was exposed: sparse rewards. This in combination with the limited data set caused no propagation of rewards out to other states. The policy generated took no action at any states. This resulted in a score of 6339—very common on the leaderboard—which suggests that many model-free methods performed poorly with the sparse rewards of the medium data set.

For the large MDP, Q learning performed okay compared to other submitted policies. Even after many time steps, the action matrix was fairly sparse and many states had a 157-way tie between possible actions, each with 0 expected future reward. Initially, just action 1 was used in the event of a tie, but this strategy was replaced with random actions in the event of a tie. This worked much better because it is akin to the idea of exploration (although it isn't strictly exploration). That is to say that taking some random actions gives the reward more opportunity to propagate through the matrix than deterministically taking action in the event of a tie. Random actions increase the possibility of landing in a state with a known trajectory toward reward. This method did not perform any better in the medium problem due to its sparse rewarding.

Conclusion

Q learning is limited in its effectiveness. It works well in small data sets with sampling that is significant relative to the total possible state-action space. With fewer state-action samples, too many pathways are obscured to be a useful method. Also of note, sparsity of rewards has a large negative impact on Q learning. This algorithm could be improved upon with generated models for the data. It could also potentially be improved by grouping similar states and treating them to have similar properties deserving of similar actions.