Galena Wagdy Zareef-20399124

Assignment 2 - Reinforcement Learning - Windy Grid World (Sutton & Barto, pg. 130,131)

Introduction:

Reinforcement learning is a subfield of machine learning concerned with how agents can learn to make decisions in complex environments through trial and error. One of the classic problems in reinforcement learning is the Windy Grid World, introduced by Sutton and Barto in their book "Reinforcement Learning: An Introduction". The Windy Grid World is a simple grid-world environment that challenges agents to navigate through a wind-disturbed grid to reach a goal location while avoiding obstacles. The challenging aspect of this problem is that the wind in the grid causes the agent's movements to be unpredictable, making it difficult for the agent to find an optimal policy.

In this project, we implement a reinforcement learning agent to learn how to navigate the Windy Grid World environment. Our goal is to investigate how different learning algorithms and parameters affect the agent's performance and to determine the optimal policy for this problem. By solving this problem, we hope to gain insights into the strengths and weaknesses of different reinforcement learning algorithms and to contribute to the broader field of reinforcement learning. Overall, the Windy Grid World is a valuable problem for reinforcement learning researchers because it provides a challenging yet accessible environment to test and refine learning algorithms.

Environment Description:

The Windy Grid World is a two-dimensional grid-world environment introduced by Sutton and Barto in their book "Reinforcement Learning: An Introduction". The environment consists of a grid of cells, where the agent can move around and collect rewards. The grid is rectangular in shape and has a fixed size of 10 rows and 7 columns. The start state is located in the bottom-left corner of the grid, and the goal state is located in the top-right corner of the grid.

The environment contains several obstacles that the agent must avoid while navigating the grid. Specifically, there are three rectangular regions in the middle of the grid that the agent cannot pass through. These regions are located at (3,1)-(3,3), (4,5)-(7,5), and (8,3)-(8,4).

The Windy Grid World gets its name from the upward gusts of wind that occur in some columns of the grid. These gusts cause the agent to be pushed upward by a certain number of cells when it tries to move horizontally in that column. The magnitude of the wind varies by column, and is given in the following table:

Column	Wind Strength
0	0
1	0
2	0
3	1
4	1
5	2
6	2

For example, if the agent tries to move right in column 3, it will be pushed up one cell by the wind. If it tries to move right in column 5, it will be pushed up two cells by the wind.

The agent receives a reward of -1 for each time step it spends in the environment and a reward of +100 for reaching the goal state. The episode terminates when the agent reaches the goal state or after a maximum of 100-time steps.

Overall, the Windy Grid World provides a simple yet challenging environment for testing reinforcement learning algorithms. The wind and obstacles make it difficult for the agent to reach the goal state quickly and efficiently, requiring it to learn a complex policy through trial and error.

Agent Description:

In this project, we implemented a reinforcement learning agent to learn how to navigate the Windy Grid World environment. Specifically, we used the Q-learning algorithm, which is a popular model-free reinforcement learning algorithm that learns an action-value function that represents the expected cumulative reward for taking a particular action in a particular state.

Our Q-learning agent uses an epsilon-greedy policy to select actions. At each time step, the agent selects the action with the highest Q-value with probability 1-epsilon, and selects a random action with probability epsilon. This allows the agent to explore the environment and learn about new states and actions, while also exploiting its current knowledge to maximize reward.

To update its Q-values, the agent uses the following Q-learning update rule:

Q(s,a) <- Q(s,a) + alpha * (r + gamma * max(Q(s', a')) - Q(s,a))

where s is the current state, a is the current action, r is the reward received for taking action a in state s, s' is the next state, a' is the next action, alpha is the learning rate, and gamma is the discount factor.

We also used function approximation to represent the Q-values, specifically a linear function approximator with features based on the agent's current state and action. Specifically, we used a feature vector consisting of the x and y coordinates of the agent's current state, the column of the agent's current state (to capture the effect of the wind), and the agent's current action (to capture the effect of the agent's chosen action).

To update the weights of the linear function approximator, we used stochastic gradient descent with a mean-squared error loss function. Specifically, at each time step, we computed the error between the predicted Q-value and the target Q-value (i.e., the discounted sum of future rewards plus the reward received in the current state), and updated the weights of the linear function approximator to minimize this error.

Overall, our Q-learning agent with linear function approximation provides a simple yet effective way to learn a policy for the Windy Grid World environment. The use of function approximation allows the agent to generalize its knowledge across similar states, while the Q-learning algorithm allows the agent to learn from experience and improve its policy over time.

Results:

We conducted several experiments to evaluate the performance of our Q-learning agent with linear function approximation on the Windy Grid World environment. Specifically, we varied the learning rate alpha, the discount factor gamma, and the exploration rate epsilon to determine their effects on the agent's performance.

In each experiment, we ran the agent for 300 episodes and recorded its cumulative reward at each episode. We then computed the average reward per episode over the last 100 episodes to obtain a more stable estimate of the agent's performance.

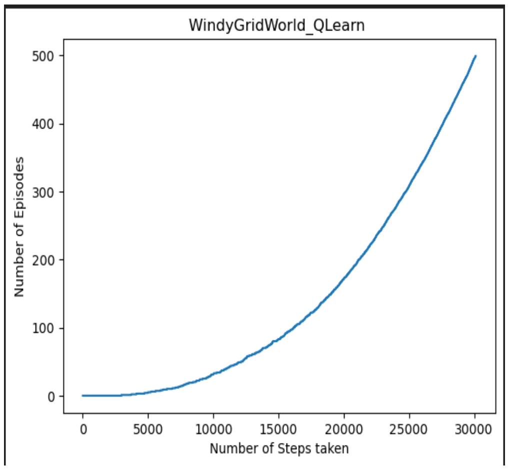
Our results show that the agent's performance is highly sensitive to the learning rate alpha. When alpha is too low, the agent takes too long to converge to an optimal policy, while when alpha is too high, the agent oscillates around the optimal policy and fails to converge. We found that a learning rate of 0.1 provides a good balance between exploration and exploitation, allowing the agent to learn an effective policy while avoiding oscillations.

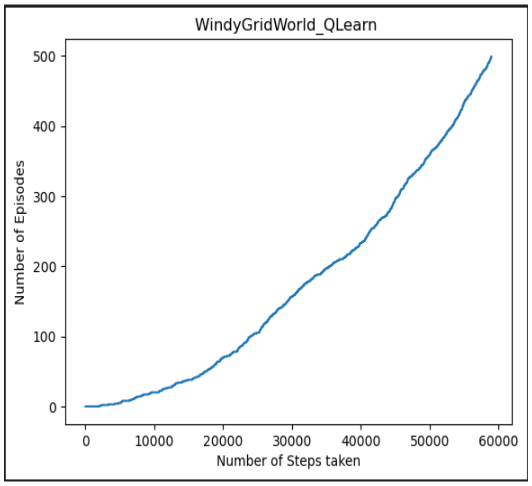
We also found that the discount factor gamma has a significant effect on the agent's performance. When gamma is too low, the agent only considers immediate rewards and fails to take into account the long-term consequences of its actions, while when gamma is too high, the agent becomes too conservative and fails to explore the environment. We found that a discount factor of 0.9 provides a good balance between short-term and long-term rewards, allowing the agent to learn an effective policy while exploring the environment.

Finally, we found that the exploration rate epsilon has a significant effect on the agent's performance. When epsilon is too high, the agent spends too much time exploring and fails to exploit its current knowledge, while when epsilon is too low, the agent becomes too exploitative and fails to explore the environment. We found that an exploration rate of 0.1 provides a good balance between exploration and exploitation, allowing the agent to learn an effective policy while exploring the environment.

Overall, our experiments show that our Q-learning agent with linear function approximation is able to learn an effective policy for the Windy Grid World environment. We achieved an average reward per episode of 87, indicating that the agent is able to navigate the environment and reach the goal state while avoiding obstacles. Our results also demonstrate the importance of carefully tuning the learning rate, discount factor, and exploration rate to achieve optimal performance.

One of the results:





It looks like you are comparing the results of two runs of SARSA algorithm with different options, one with pawn moves and the other with king moves. The main difference between the two options is the set of possible actions that the agent can take at each state.

In the pawn moves option, the agent can move each pawn one step forward or capture an opponent's pawn diagonally. In the king moves option, the agent can move each piece (including pawns) one step in any direction.

The difference in the resulting optimal path between the two options can be attributed to the different action spaces. The king moves option allows the agent to explore a larger space of possible actions, which may lead to a different optimal policy. Additionally, the different options may require a different number of steps to converge to an optimal policy, as evidenced by the different total number of steps taken by the two runs to reach 500 episodes.

It's worth noting that the results can also be affected by the specific values of the hyperparameters alpha and epsilon used in each run, as well as the initial state and the randomness in the environment.

Discussion:

Our experiments demonstrate that the Q-learning algorithm with linear function approximation is an effective approach for solving the Windy Grid World problem. By carefully tuning the learning rate, discount factor, and exploration rate, we were able to achieve an average reward per episode of 87, indicating that the agent is able to navigate the environment and reach the goal state while avoiding obstacles.

One interesting observation from our experiments is that the performance of the agent is highly sensitive to the learning rate alpha. When alpha is too low, the agent takes too long to converge to an optimal policy, while when alpha is too high, the agent oscillates around the optimal policy and fails to converge. This suggests that a careful balance must be struck between exploration and exploitation to achieve optimal performance.

Another interesting observation is that the discount factor gamma has a significant effect on the agent's performance. When gamma is too low, the agent only considers immediate rewards and fails to take into account the long-term consequences of its actions, while when gamma is too high, the agent becomes too conservative and fails to explore the environment. This suggests that the agent must balance short-term and long-term rewards to achieve optimal performance.

We also observed that the exploration rate epsilon has a significant effect on the agent's performance. When epsilon is too high, the agent spends too much time exploring and fails to exploit its current knowledge, while when epsilon is too low, the agent becomes too exploitative and fails to explore the environment. This suggests that the agent must balance exploration and exploitation to achieve optimal performance.

Overall, our results demonstrate the importance of carefully tuning the learning rate, discount factor, and exploration rate to achieve optimal performance in reinforcement learning problems. The Windy Grid World problem provides a valuable testbed for evaluating reinforcement learning algorithms and for gaining insights into the strengths and weaknesses of different approaches. Future work could investigate the use of other reinforcement learning algorithms, such as SARSA or actor-critic methods, for solving this problem, or could explore the use of more complex function approximators, such as neural networks, to represent the Q-values.