Q-Learning in the snake game

Adam Zaiter

***Abstract*—**

***Index Terms*—**

I. Introduction

II. Snake game

The snake game was programmed in python. It supports square grids of any size that is $>= 3$. The snake starts in the middle of the grid with a head of 1 square in size and attached body with 2 squares in size. There also is food spawned in a random square that is not occupied by the snake. The objective for the snake is to eat as much food as possible. If the food is eaten, the snake gains + 10 score and grows an additional body part of 1 square in size. If the snake collides with the borders of the grid or the head collides with it's own body, the game ends.

III. Episode

An episode contains all of the actions taken in one game of the snake. The episode ends when a terminal state is reached (game ends), described in II. In the case of the snake game that is either the head of the snake colliding with a border or the head colliding with it's own body.

IV. Q-Learning

Q-Learning finds the optimal policy by learning the best Q values for each state-action pair.

The Q function accepts the current state and an action. It then returns the expected reward for taking said action for the given state.

The agent then plays episodes. For each step in the episode the Q values in the Q table are updated using the Bellman equation. This process goes on until the q function converges to the optimal function q*.

V. State space

For this project, the computing resources are somewhat limited and as such, these limited resources have to be taken into consideration when designing the state and action spaces. The state space for a snake game with a variable size would be too big, if it were taken as is. That is, because the snake and it's body adds many additional states, as the snake can grow it's body to be the same size as all of the squares in the grid. Also, the head is distinct from the other body parts and would have to be accounted for in all of the states.

VI. Training

Best results were seen when training on a 3x3 grid. The training did not take as much time, as fewer moves were made on average.