Competition

María Ángeles Magro Garrote - 100472867
Marina Gómez Rey - 100472836

Description

Firstly, our state tuple, which is used to identify the situation of the game, has two attributes: the direction in which the closest element (ghost or pac dot) is at and the discretized distance based on if it is close, mid-close, mid-far or far.

Furthermore, our q-table is generated initially with 16 rows as our two attributes have 4 different possible values. So the total number of possible combinations is 16 states. Also, there are five columns that correspond to the different actions the agent can take (Stop remains at 0 always).

What is more, the definition of alpha, epsilon and discount were studied so that the best value could be setted when testing. Specifically, the value of epsilon was changed constantly depending on the phase of learning the Pacman was (the first time it was set in 1 and then it kept decreasing).

Our reward function only rewards when something is eaten (it gives more reward to the pac dots so it prioritizes it as the game finishes when the ghosts are eaten) and only penalizes when a wall inside the grid is faced (with the exception of the borders) so that it does not get stuck in any place.

The methods that were coded inside the QLearningAgent class are: update, where the q-table is updated in every tick depending on the reward and the previous value; getReward, where the reward value was returned based on the previous idea; and computePosition, where the row of the q-table we are at right now is computed based on the attributes present on the q-table (closest element direction and distance). Regardless of them, we included two extra methods that were necessary to compute the previous ones: countFood, that returns the number of pac dots that are left in the grid, and closestElement, which computes the coordinates of the pac dots and the ghosts and calculates the Distances distance to them to return the smallest one.