Report AI

Question 1 – Reflex Agent.

For the reflex agent, I started with giving a score based on the amount of foods that are left, but if no food is close and thus no food is eaten, the score would be the same and thus the pacman does not know where to go. So I implemented a way to get a higher score, the closer it gets to the nearest food. If there is no food left, the score goes up by 100, because this is the only way to get the highest score. When there is food left, the score is 1 divided by the distance of the nearest food. This way it will always want to go to the nearest dot.

There is one exception, when a ghost is about to eat the pacman. The pacman should play safe and keep away from the ghosts. When a ghost is one block away from the pacman, the score goes down by 100 points to prevent the possibility to get eaten. This way the pacman will always (if possible) prevent being too close to the ghost.

At last, the actual score of the game is added to make sure the pacman doesn't just stay still and not hunt for food.

Question 5 – Evaluation Function.

For the more advanced evaluation function, I first made sure that a gamestate in the winning state is infinitely good and losing is infinitely bad. Otherwise, the score is initialized to the actual score of the game.

Taking the amount of food there is left into account did not work out in the reflex agent. Although, with the help of other features that manipulate the score, it does work in this implementation. I chose a constant (25.0) and divided it by the amount of food left to give this feature more weight. The result is added to the score.

For each active ghost in the maze, the Manhattan distance is calculated. Again a constant is chosen to weight it more, in case of the ghost is scared, or (negatively) less, in case the ghost is trying to eat the pacman. So the score gets lower when the ghost in normal condition is closer and the score gets higher when the ghost in scared condition is closer.

Finally, I (again) calculate the Manhattan distance of nearest dot from the pacman and add a constant (10.0) divided by this distance to the score.

Question – Fruitful features.

Taking the nearest dot works pretty well for a great score. Lowering the score depending on the distance to a ghost also works well, if the pacman gets eaten by a ghost, the score would go down a lot.

Question – Reciprocals.

Using the reciprocals of these features would result in the opposite of what it should do. The score gets higher if the pacman gets further away from the nearest dot or when it gets closer to a ghost that tries to eat it.