Aprendizaje Automático

GRADO EN INGENIERÍA INFORMÁTICA

Competición

Curso 2020/2021

Jorge Rodríguez Fraile, 100405951, Grupo 83, 100405951@alumnos.uc3m.es
Carlos Rubio Olivares, 100405834, Grupo 83, 100405834@alumnos.uc3m.es

Índice

Descripción del agente

Se emplea aprendizaje por refuerzo, mediante Q-learning.

Nuevos estados: Hemos eliminado la orientación del Pac-man ya que hemos supuesto que generaba bastantes problemas de reconocimiento de estado por nuestra parte, además la función de refuerzo que implicaba este atributo era bastante débil por lo que hemos decidido eliminarlo.

Para suplir la eliminación de este atributo hemos hecho de la distancia al Pac-man un atributo más sensible añadiendo nuevos grados de cercanía. Un problema bastante grande que nos hemos encontrado ha sido en el tercer mapa, ya que el callejón sin salida donde se encuentra unos de los fantasmas es un obstáculo bastante grande a superar ya que alejarse del fantasma más cercano es fatal para nuestro agente.

Para intentar relacionar al agente con los muros hemos creado un nuevo atributo, una lista de 4 booleanos que nos dice si hay algún muro en alguna de sus 4 posiciones.

- **Distancia de Pac-man al fantasma más cercano (computeNearestGhostDistance2):** Se calculará en cuál de estas regiones se encuentra observando cual es la menor distancia a los fantasmas vivos.
    - **Muy cerca**: Distancia menor de 4 unidades.
    - **Cerca**: Distancia entre 4 y 7 unidades.
    - **Medio**: Distancia entre 7 y 10 unidades.
    - **Lejos**: Distancia entre 10 y 13 unidades.
    - **Muy Lejos**: Distancia mayor de 13 unidades.

- **Muros alrededor (nearWallUp, nearWallDown, nearWallLeft y nearWallRight)**: Vector de 4 booleanos para observar si en cada una de las posiciones adyacentes del Pac-man hay muros. Las posiciones a las que pertenecen las direcciones son las siguientes: Siguiendo que cada uno es 1 o 0, 1 si hay muro y 0 si no lo hay, el número que codifica este vector es ‘Arriba Abajo Izquierda Derecha’.

- **Distancia a la comida más cercana (computeNearestDotDistance):** Se mantiene el definido para la primera alternativa.
    - **Cerca**: Distancia menor de 7 unidades.
    - **Media**: Distancia entre 7 y 15 unidades.
    - **Lejos**: Distancia más de 15 unidades

- **Dirección al fantasma cercano reducido (computeNearestGhostDirection2):** En este caso, hemos pasado de 8 posibles direcciones en las que puede identificar la dirección a solo 2, que se identifican si es moviéndose en el eje x o en el eje y. Lo que nos permite que se codifique con un solo bit, el criterio para tomar un valor u otro es:
    - **Eje x:** Si la diferencia entre las dimensiones x del fantasma y Pac-man son menores que las y. Es el valor 0.
    - **Eje y:** Al contrario que el anterior, si las diferencia en el eje y es menor que la del eje x. Tomará el valor 1.

Número de estados: Todas las combinaciones de los atributos de los estados nos dejan un total de 3*2*5*16 = 480 estados.

Función de transición: Si es legal la acción pedida se realiza, en otro caso, se queda en el mismo estado, actualizando la tabla Q con los nuevos valores de refuerzo y se vuelve a estimar la acción a tomar.

Función de refuerzo: La función de refuerzo de este conjunto de atributos se ha basado en la suma de valores de las siguientes funciones:

- **sumaFantasma:** Función que otorga al agente un refuerzo positivo por comerse a un fantasma, el valor con el que recompensa es de 400 unidades.

- **sumaComida:** Función que otorga al agente un refuerzo positivo por comerse un punto de comida, al dar la mitad de los puntos otorga la mitad de refuerzo que comerse un fantasma 200 unidades.

- **sumaAcercarseF2:** Función que recompensa al agente en función de su grado de cercanía al fantasma más cercano, cuanto más se acerca más premia (haciéndolo solo al transitar entre zonas), el castigo de salir de una zona es mayor que el de entrar para que no entre y salga constantemente. Los refuerzos siguen el siguiente esquema:
    - **Muy lejos a lejos:** 1.25 unidades.
    - **Lejos a medio:** 2.5 unidades.
    - **Medio a cerca**: 5 unidades.
    - **Cerca a muy cerca:** 10 unidades.
    - **Muy cerca a cerca:** -20 unidades.
    - **Cerca a medio:** -10 unidades.
    - **Medio a lejos:** -5 unidades.
    - **Lejos a muy lejos**: -2.5 unidades.

- **sumaAcercarseC:** Función que recompensa al agente en función de su grado de cercanía a la comida más cercana, funciona igual que en la alternativa 1.

- **sumaIrDirecciónCorrecta:** Función que refuerza positivamente al agente por ir en la dirección correcta al fantasma más cercano.

- **choquePared2**: Función que penaliza al Pac-man por tener un muro adyacente en la misma dirección que el fantasma más cercano como ocurría en la primera alternativa, pero en este caso hemos buscado que castigue por los muros adyacentes al que el impide ir directo, al igual que el estar acorralado en un pasillo. Para poder hacer lo anterior, mantenemos la detección de muro en la dirección de Pac-man, pero vamos añadiendo castigo según los muros.

  De tal manera que tener un muro en la dirección es un castigo de -40 unidades, -40 más por muros adyacentes y otros -40 si le arrinconan.

- **avoidLoop**: Función que penaliza al agente por volver a visitar un estado que ha visitado en la iteración anterior, evitando bucles, para detectar este evento lo que hacemos es

recurrir al estado siguiente y mirando la dirección que tiene en el estado actual podemos saber de dónde proviene, por lo que sí coinciden es un bucle de longitud 1. Cuando esto ocurre le castigamos con -40 unidades.

- A parte de esto, también se le resta 1 por cada tick debido a la pérdida de score en cada turno.