Universidad Carlos III de Madrid

Aprendizaje Automático

Grado en Ingeniería Informática

Grupo 83

Tutorial 4: Introduccíon al aprendizaje por refuerzo

Autores:
Daniel Medina García
Alejandro Rodríguez Salamanca

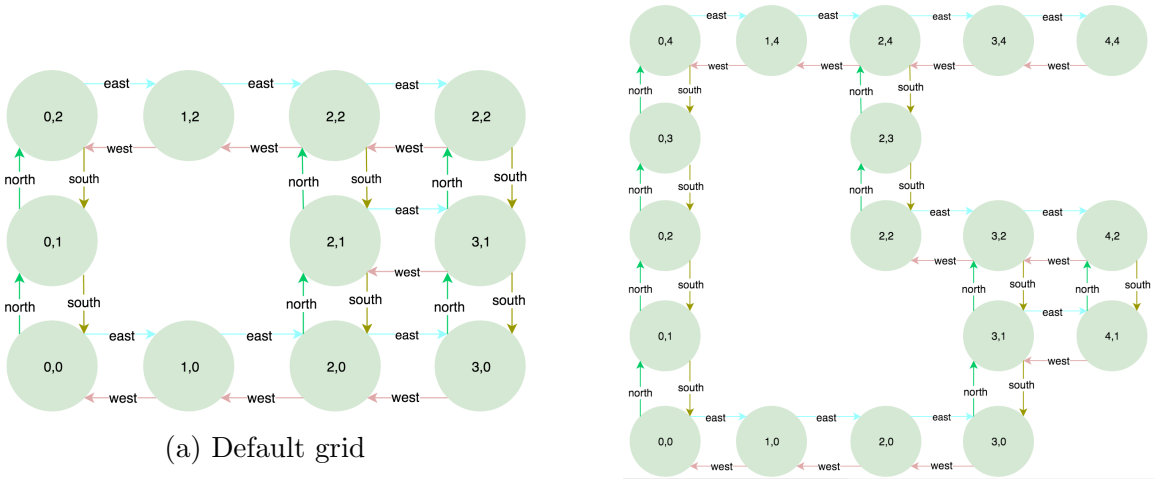
12 de abril de 2016

Índice

Introducción

1. Ejercicio 1

En este primer ejercicio tomamos contacto con el programa ejecutando tanto el agente manual como el estándar. Por cada movimiento ejecutado, en la salida por consola aparecen el estado inicial, la acción tomada, el estado al que se llega y la recompensa obtenida. El agente por defecto es *random*, que toma acciones aleatorias entre las permitidas. La Figura 1 muestra los MDP deterministas requeridos en el enunciado del tutorial:



(a) Default grid

(b) AAGrid

Figura 1: Deterministic MDPs

Estos laberintos, al igual que los demás incluidos en el código proporcionado, se almacenan en forma de arrays bidimensionales. Cada posición de la matriz representa una posición del laberinto, con los siguientes símbolos:

- **Un espacio** representa una casilla común, sin recomensa ni coste algunos, por la que el agente puede pasar.
- **Un valor numérico** representa una casilla con una recompensa o un precio. Si el agente se mueve a esta casilla cambiará su puntuación añadiéndole el valor contenido en dicha casilla.
- **Una almohadilla (#)** es considerada un muro, o casilla inalcanzable por el agente. Ninguna acción legal puede llevar al agente a estas posiciones.
- Por último, la posición de la que parte el agente se guarda como **una 'S'**.

De esta forma, podemos crear un nuevo laberinto siguiendo las instrucciones mencionadas. La siguiente figura muestra un laberinto creado por nosotros, en la que el agente deberá buscar una ruta sorteando los muros si quiere conseguir la mayor recompensa:

```
[[' ',' ',' ',' ',]
 ['8','#','3',' ',]
 ['#',' ',' ','#',]
 [' ',' ',' ',' ',]
 [' ','S','#',' ',]]
```

Figura 2: New grid

La elección de la política para guiar a un agente es clave para su éxito, y para ello se busca elegir la política óptima. Sin embargo, esta política no tiene por qué ser única y se caracteriza por compartir el mayor *valor Q* de todas las políticas posibles. Así, para el laberinto por defecto tenemos dos políticas óptimas, las cuales muestra la figura a continuación.

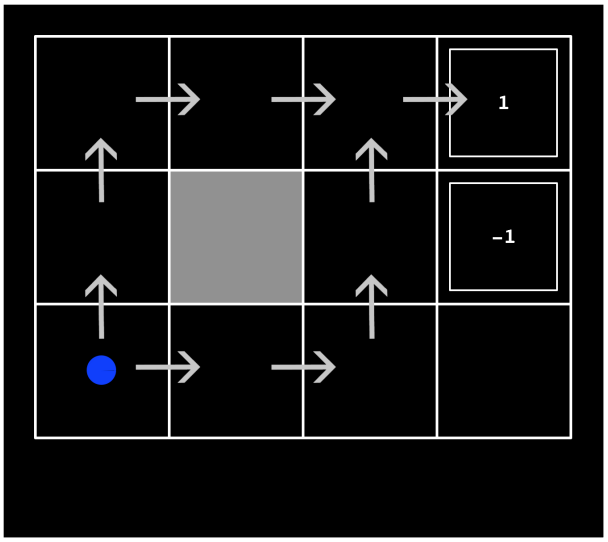


Figura 3: Políticas óptimas para el laberinto por defecto

2. Ejercicio 2

En este ejercicio, implementamos el código necesario para que un agente automático fuese capaz de basar su política de acción en una *tabla Q* dada a través de un fichero. Para ello:

- El constructor de la clase inicializa los *valores Q* a través de `readQtable`.
- `readQtable` guarda en un array `q_table` los *valores Q* de cada acción para cada uno de las acciones posibles desde el archivo donde se almacena dicha tabla.
- `writeQtable` actualiza el fichero con la actualizada *tabla Q* tras la ejecución del juego.
- `computeActionFromQValues` devuelve, dado un estado, la acción con mayor *valor Q* según la *tabla Q* actual. De esta forma, devuelve la acción elegida por la política más avara de todas.
- `computeQValueFromQValues` devuelve, dado un estado, una acción y la *tabla Q*, el *valor Q* actual para dicha acción. Se tuvo que refactorizar el código para que el nombre de esta función fuese el mismo, dado que por defecto se denominaba `getQValue`.

Inicializamos la *tabla Q* de dos maneras: con $\gamma = 0{,}9$ en primera instancia como requería el enunciado, y más allá con todos los valores iguales a cero:

state #	North	East	South	West	Exit
0	0.0	0.0	0.0	0.0	0.0
1	0.0	0.0	0.0	0.0	0.0
2	0.0	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0	0.0
5	0.0	0.0	0.0	0.0	0.0
6	0.0	0.0	0.0	0.0	0.0
7	0.0	0.0	0.0	0.0	0.0
8	0.0	0.0	0.0	0.0	0.0
9	0.0	0.0	0.0	0.0	0.0
10	0.0	0.0	0.0	0.0	0.0
11	0.0	0.0	0.0	0.0	0.0

(a) Initialized with $\gamma = 0$

state #	North	East	South	West	Exit
0	0.0	0.0	0.0	0.0	0.0
1	0.0	0.0	0.0	0.0	0.0
2	0.0	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0	0.0
5	0.0	0.0	0.0	0.0	0.0
6	0.0	0.0	0.0	0.0	0.0
7	0.0	0.0	0.0	0.0	0.0
8	0.0	0.0	0.0	0.0	0.0
9	0.0	0.0	0.0	0.0	0.0
10	0.0	0.0	0.0	0.0	0.0
11	0.0	0.0	0.0	0.0	0.0

(b) Initialized to 0

Figura 4: Initial Q tables

3. Ejercicio 3