Computación Bioinspirada - Práctica N° 7

PROFESOR DEL CURSO: Dennis Barrios Aranibar **FECHA:** 1 de Octubre del 2018
ASISTENTE DEL CURSO: Kevin Christian Rodríguez Siu

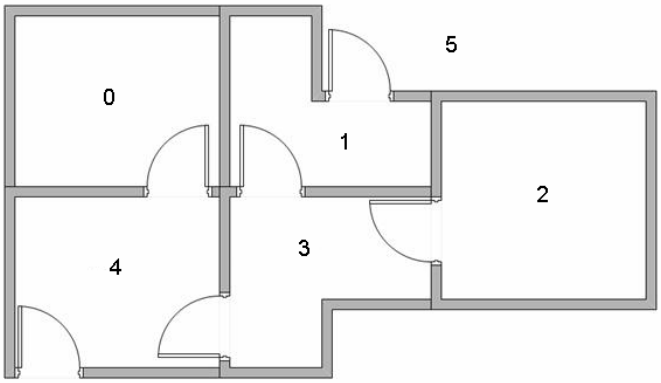
Objetivos de la Sesión

- Analizar como funciona Q-Learning y sus aplicaciones.

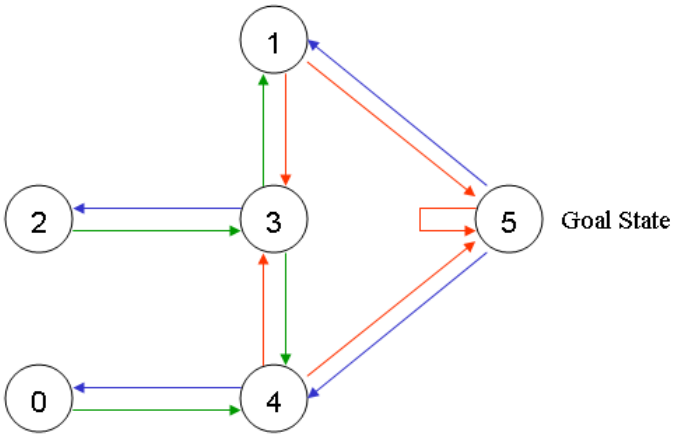
Contexto

Para explicar la técnica de Q-Learning, usaremos un ejemplo práctico: Describiremos como puede ser usado para lograr que un agente pueda ser entrenado para poder descrubrir información en un ambiente desconocido.

Supongamos que tenemos 5 cuartos en un edificio conectados por puertas, como se muestra en la figura inferior. Cada cuarto tiene un número, desde 0 hasta 4. El exterior del edificio se considera un gran cuarto marcado con 5. En este caso en particular, las puertas 1 y 4 llevan a este cuarto 5 (o al exterior).

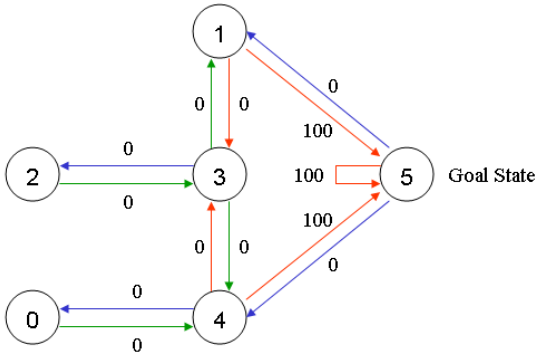


Podemos representar el edificio como un grafo, cada cuarto siendo un nodo y cada puerta como una arista.



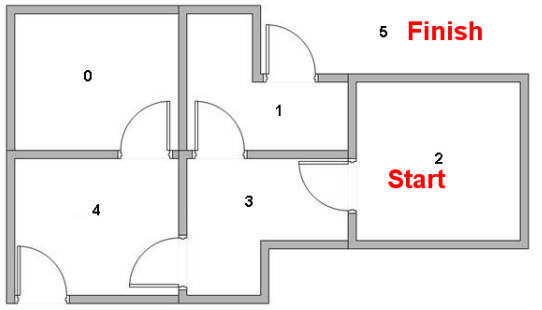
Para este ejemplo, quisieramos poner un agente en cualquier cuarto y, desde aquel cuarto, encontrar una ruta hacia el exterior (siendo este nuestro cuarto-objetivo). En otras palabras, la

meta es llegar al cuarto número 5. Para poner este cuarto como tal, vamos a asociar cada puerta con un valor de recompensa (es decir, asignaremos un peso a la arista). Las puertas que lleven directamente al exterior tendrán una recompensa de 100. Otras puertas tendrán una recompensa de 0. Ya que las puertas conectan los cuartos en doble dirección (así como 0 lleva a 4, 4 lleva a 0), se asignan dos aristas para cada puerta. Cada una contiene una recompensa separada, como se muestra aqui:

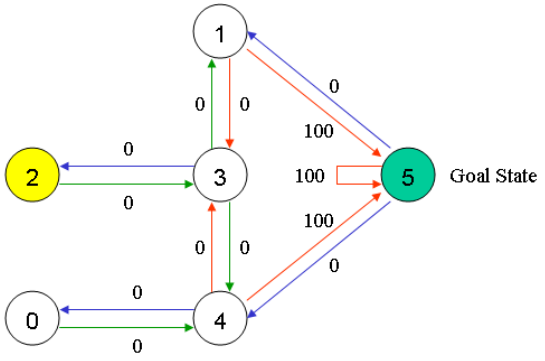


Por supuesto, el cuarto 5 tiene un lazo con una recompensa de 100, y todas las conexiones directas a la meta llevan una recompensa de 100. En Q-Learning, la meta es llegar al estado con la recompensa más alta de tal forma que cuando el agente llegue a esa meta, se quede allí para siempre. A este tipo de meta se le conoce como una «meta absorbente».

Ahora, imaginemos que el agente que queremos programar aprende por experiencia. El agente pude pasar de un cuarto a otro, pero no tiene conocimiento del ambiente, y no sabe cual secuencia de puertas lleva hacia el exterior. Supongamos, por ejemplo, que tenemos al agente actualmente en el cuarto 2 y queremos que aprenda a alcanzar el exterior del edificio (el cuarto 5).



En Q-Learning, utilizamos los términos «estado» y «acción». Cada cuarto, incluyendo el exterior, es una estado. Y cada movimiento del agente de un cuarto a otro será una acción. En el diagrama mostrado, un estado se representa como un nodo, y una acción se representa por las aristas.



El agente se encuentra en el estado 2. De este estado, puede ir al estado 3, ya que el estado 2 está conectado al estado 3. Del estado 2, sin embargo, el agente no pude ir directamente al estado 1 ya que no hay una puerta que conecte estos cuartos directamente (por tanto, no hay una arista). Del estado 3, puede ir al estado 1 o al estado 4 o de regreso al estado 2 (observa las aristas que están unidas al estado 3). Si el agente estuviera en el estado 4, entonces las tres posibles acciones son ir al estado 0, 5 o 3. Si el agente está en el estado 1, puede ir al estadao 5 o 3. Del estado 0, sólo puede ir al estado 4.

Podemos representar todos estos caminos, el diagrama de estado y los valores de recompensa instantanea dentro de la siguiente matriz de recompensas, la matriz R. Representamos con el valor de -1 cuando no exista un camino directo entre esos estados.

$$R = \begin{array}{c} \\ \text{State} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \text{Action} \\ \begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} -1 & -1 & -1 & -1 & 0 & -1 \\ -1 & -1 & -1 & 0 & -1 & 100 \\ -1 & -1 & -1 & 0 & -1 & -1 \\ -1 & 0 & 0 & -1 & 0 & -1 \\ 0 & -1 & -1 & 0 & -1 & 100 \\ -1 & 0 & -1 & -1 & 0 & 100 \end{bmatrix} \end{array}$$

Ahora, usaremos una matriz similar, Q, que será el cerebro de nuestro agente, representando la memoria de lo que el agente ha aprendido por experiencia. Las filas de la matriz Q representan el estado actual del agente, y las columnas representan las posibles acciones que llevan al siguiente estado (los enlaces entre los nodos).

El agente empieza sin saber nada, por lo que la matriz Q se inicializa a cero (0). En este ejemplo, por simplicidad, asumimos que el número de estados es conocido (tenemos 6 estados). Si no supieramos cuantos estados están implicados, la matriz Q iniciaría con solo un elemento. Es una tarea simple el añadir más columnas y filas en la matriz Q si un nuevo estado fuera encontrado.

La regla de transición en Q-Learning es una fórmula simple:

$$Q(estado, accion) = R(estado, accion) + \Gamma \times max(Q(siguienteestado, todaslasacciones)$$

De acuerdo a esta fórmula, el valor de un elemento específico en la matriz Q es igual a la suma del valor correspondiente en la matriz R y el parámetro de aprendizaje Γ multiplicado por el valor máximo de Q para todas las acciones posibles en el siguiente estado.

Nuestro agente virtual aprenderá por experiencia sin ninguna guía, (a esto se le conoce como aprendizaje no supervisado). El agente explorará de estado a estado hasta que llegue a la meta. Llamamos a cada exploración un episiodio. Cada episodio consiste en el agente moviéndose desde el estado inicial al estado objetivo. Cada vez que el agente llega al estado objetivo, el programa va al siguiente episodio.

El algoritmo de Q-Learning va entonces de la siguiente forma:

1. Selecciona el parámetro Γ y las recompensas de ambiente en la matriz R.
2. Inicializa la matriz Q a cero.
3. En cada episodio:
    - Selecciona un estado inicial aleatoriamente.

- Mientras no se haya alcanzado el estado objetivo:
  (a) Selecciona una de todas las posibles acciones del estado actual.
  (b) Usando esta posible acción, considera el siguiente estado.
  (c) Obtén el máximo valor Q para este siguiente estado basándose en todas las acciones posibles.
  (d) Calcula: $Q(estado, accion)$ con la fórmula.
  (e) Coloca el siguiente estado como el estado actual.

Este algoritmo es usado por el agente para aprender por experiencia. Cada episodio es equivalente a una sesión de entrenamiento. En cada sesión de entrenamiento, el agente explora el ambiente (representado por la matriz R), recibe una recompensa (si es posible) hasta que llega a su estado objetivo. El propósito del entrenamiento es el mejorar el «cerebro» del agente, representado por la matriz Q. Mucho más entrenamiento resulta en una matriz Q más optimizada. En este caso, si la matriz Q ha sido mejorar, en lugar de explorar o ir de una habitación a otra, el agente logrará encontrar la ruta más rápida al estado objetivo.

El parámetro Γ tiene un rango entre 0 y 1 ($0 \leq \Gamma < 1$). Si Γ está más cerca a 0, el agente tenderá a considerar solamente las recompensas inmediatas. Si Γ está cerca a 1, el agente considerará recompensas futuras con mayor peso, haciendo que demore la recompensa.

Para usar la matriz Q, el agente simplemente traza la secuencia de estados, desde el estado inicial al estado objetivo. El algoritmo encuentra las acciones con la mayor recompensa grabados por la matriz Q en el estado actual. El algoritmo de su uso es así:

1. Colocar como estado actual el estado inicial.
2. Desde el estado actual, encontrar la acción con el valor Q más alto.
3. Elegir como estado actual ese estado siguiente.
4. Repetir pasos 2 y 3 hasta que el estado actual sea el estado objetivo.

Este algorimto permite que se retorne una secuencia de estados desde el estado inicial al estado objetivo.

Ejercicios

1. Programar el algoritmo de Q-Learning para que pueda representar el problema del que se habla en la sección Contexto. Todos los elementos del mismo deben estar correctamente identificados:
   - Estados y acciones
   - Matrices R y Q-
   - Coeficiente de aprendizaje Γ
2. Mostrar la matriz Q después de 100 episodios con diferentes valores de Γ:
   - $\Gamma = 0.1$
   - $\Gamma = 0.5$
   - $\Gamma = 0.9$
   - $0.1 < \Gamma < 0.5$
   - $0.5 < \Gamma < 0.9$

3. Calcular las rutas más cortas hacia el estado 5 desde cada uno de los otros estados luego de cada episodio. Indicar después de cuantos episodios en cada uno de los casos anteriores se ha logrado encontrar la ruta hacia el final.

No es necesaria una interfaz gráfica avanzada, pero se debe mostrar las matrices R y Q, como van cambiando con cada iteración y, al final del proceso, las rutas que el sistema ha aprendido. Puedes usar el lenguaje de programación de tu preferencia.

Actividades

1. Mostrar las rutas encontradas por el algoritmo. ¿Después de cuantos episodios se logra encontrar la ruta con cada coeficiente de aprendizaje Γ? ¿Cuál fue el más rápido y cual fue el más lento?

2. ¿Qué parámetro Γ logra al final de los 100 episodios una matriz Q con valores con mayor diferencia entre sí? ¿Qué crees que significa esto? Convierte los valores a porcentaje. ¿Esta diferncia se mantiene?

3. Menciona 3 problemas que podrían ser resueltos con Q-Learning y describe brevemente los posibles estados y acciones, y como serían las matrices R y Q.

Desarrollo y Entrega

- El trabajo debe ser desarrollado en la sesión de laboratorio.
- Se debe entregar digitalmente (en un PDF vía email de preferencia) un informe conteniendo el desarrollo de todas las actividades, imágenes de los gráficos y los códigos implementados.
- Plazo de entrega del informe: 1 de Octubre del 2018.

Cuadro 1: Rúbrica de Evaluación - Práctica 4

Criterio	Deficiente (25%)	Regular (50%)	Bueno (75%)	Excelente (100%)	Total de Puntos
Modelamiento de la Solución	No existe un modelado de la técnica o solución programada, o no está definido de forma clara.	Se han definido los aspectos del desafío a resolver, y de la solución que hay que aplicar, pero no existe una relación clara entre los mismos.	Se han definido los aspectos del problema a resolver, y estos tienen relación a los componentes principales de la solución que se va a aplicar.	Se han definido los aspectos del problema a resolver y cada uno está relacionado a los componentes de la solución y la técnica a realizar.	6
Ejecución de la Técnica y Código Fuente	No existe código fuente, no es ejecutable o no se relaciona con el problema o la solución propuestos.	Existe código fuente ejecutable que tiene algunas nociones de los requerimientos del problema.	Existe código fuente ejecutable que cubre los requerimientos del problema, ejecuta la técnica pedida y muestra algún tipo de resultados.	Existe código fuente ejecutable y fácilmente legible que cubre los requerimientos del problema, ejecuta la técnica pedida y muestra resultados de acuerdo a lo solicitado en la práctica.	4
Obtención de Resultados y Visualización (Act. 1 y 2)	No hay resultados visibles, o sólo se ha mostrado el proceso de ejecución y no los resultados obtenidos.	Hay muestra del proceso de ejecución y de los resultados obtenidos, pero estos no se entienden o no son claros.	Hay muestra del proceso de ejecución y de los resultados según el formato solicitado.	Hay muestra del proceso de ejecución y de los resultados según el formato solicitado, existiendo además una breve discusión sobre los mismos.	5
Análisis Comparativo de Resultados (Act. 2 y 3)	No existe un análisis de los resultados obtenidos, o este no está documentado apropiadamente.	Existe un registro de los resultados obtenidos y una comparación entre los mismos, pero no se hace un análisis con mayor profundidad.	Existe un registro de los resultados obtenidos y un análisis entre los mismos, indicando similaridades y diferencias.	Existe un registro de los resultados obtenidos, y un análisis entre los mismos que indica similaridades, diferencias y el porqué de los resultados obtenidos, indicando también posibilidades de mejora.	5