Aprendizaje por refuerzo

Santiago Pérez Moncada

30 de marzo de 2022

1. ¿Qué es aprendizaje por refuerzo?

Es un área del aprendizaje automático inspirada en la psicología conductista, cuya ocupación es determinar qué acciones debe escoger un agente de software en un entorno dado con el fin de maximizar alguna noción de recompensa o premio acumulado. El problema, por su generalidad, se estudia en muchas otras disciplinas, como la teoría de juegos, teoría de control, investigación de operaciones, teoría de la información, la optimización basada en la simulación, estadística y algoritmos genéticos.

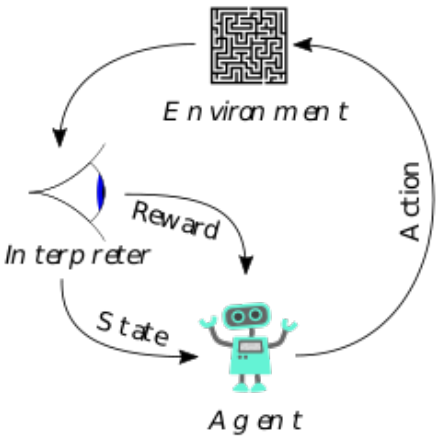

Figura 1: Escenario de aprendizaje por refuerzo

En la Figura 3 observamos un ejemplo de un robot(agente) que debe aprender a salir de un laberinto(entorno), queda claro que el agente hace diferentes acciones para salir del entorno, según la acción que tome el agente se le dará una recompensa, algunas acciones darán una recompensa alta, otras baja y quizás algunas ninguna recompensas, además el agente después de hacer una acción su estado cambiara y en base a este nuevo estado el agente toma la siguiente acción es decir el agente interpreta su entorno para tomar decisiones este proceso se

repite hasta que el agente logre salir del entorno, cabe mencionar que el modelo de la Figura 3 se puede abstraer para cualquier otro problema de aprendizaje por refuerzo, es decir que es un modelo genérico.

2. Ecuación de Bellman determinista

La ecuación de Bellman también se conoce como ecuación de programación dinámica. Richard Bellman descubrió la ecuación que también se llama método variacional moderno debido a la idea del método variacional.

La ecuación de Bellman es una condición necesaria para la optimización de varios métodos de optimización matemática, como la programación dinámica. Esta ecuación expresa 'cómo el valor del problema de decisión de un momento especifico' tiene la forma de 'el valor del problema de decisión derivado de la decisión inicial por la relación de recompensa de la elección inicial'. De esta manera, el problema de optimización dinámica se convierte en un subproblema simple, y estos subproblemas cumplen con el 'principio de optimización-reducción' o 'principio de optimalidad de Bellman' propuesto por Bellman.

2.1. Principio de optimalidad de Bellman

Principio aplicado en programación dinámica que consiste en que una secuencia óptima de decisiones que resuelve el problema debe cumplir la propiedad de que cualquier sub-secuencia de decisiones, que tenga el mismo estado final, debe ser también óptima respecto al subproblema correspondiente.

$$\mathrm{V}(s) = \max_{a}(\mathrm{R}(s,a) + \gamma\,\mathrm{V}(s'))$$

Figura 2: Ecuación de Bellman determinista

Notación de variables

- s - estados
- a - acciones
- R - recompensas
- γ - factor de descuento

3. El plan

Una vez que el agente ya calculo los valores $\mathrm{V}(s)$ para cada uno de los estados con la ecuacion de Bellman $\mathrm{V}(s) = \max_a(\mathrm{R}(s,a) + \gamma\,\mathrm{V}(s'))$ podemos hablar

ahora del 'plan de actuación', el plan de actuación es la serie de acciones que haran que consigamos el objetivo, por ejemplo que el robot salga del laberinto vea la Figura 3, Es decir que cada V(s) lo remplazaremos por la acción que me lleva al siguiente estado s con mejor o mayor V(s)

4. Búsquedas deterministas y no deterministas

- **Búsqueda determinista:** Es un modelo donde las mismas entradas o condiciones iniciales producirán invariablemente las mismas salidas o resultados, no contemplándose el azar o incertidumbre en el proceso.
- **Búsqueda no determinista:** Es un modelo en el que la salida no se puede predecir porque hay múltiples resultados posibles para cada entrada.

Los modelos no deterministas o estocásticos son muy importantes pues con el azar digámoslo así el agente puede ser mas inteligente pues de esta manera realmente puede aprender de su entorno es decir que sus acciones sean más abiertas por ejemplo que el agente tenga una probabilidad de 10 % de ir a la izquierda, de 80 % de ir hacia abajo y de 10 % de ir a la derecha, este escenario es mas interesante, pues hay factores en el entorno son aleatorios que repercuten en las decisiones que el agente deba tomar, un modelo determinista seria por ejemplo en un estado tener la probabilidad de 100 % de ir hacia abajo pero quizás en un momento no siempre sea mejor ir hacia abajo, por eso los modelos deterministas no siempre son la mejor opción y sobre todo no hay nada que pueda aprender el agente.

5. Procesos estocásticos y cadenas de Markov

Revisaremos sistemas estocasticos que serán modelados haciendo uso de **Cadenas de Markov**

5.1. Proceso de Markov

Proceso aleatorio donde el valor de la variable aleatoria en el instante n depende solamente de su valor pasado en el instante $n-1$. En un proceso de Markov, la variable aleatoria representa el estado del sistema en un instante dado n

Los procesos de Markov son ejemplos de procesos que salen en situaciones de la vida real:

- Protocolos de telecomunicación y sistemas de maquinaria
- Llegadas y salidas de clientes en bancos
- Pasar por caja en los supermercados

- Mutación de un virus o molécula de ADN
- Paseo aleatorio como por ejemplo el movimiento Browniano
- Llegada de coches a una intersección
- El estado del tiempo diario

5.2. Cadenas de Markov

Si el espacio de estados (espacio muestral) de un proceso de Markov es discreto, el proceso de Markov se denomina **Cadena de Markov**.En este caso, los estados se etiquetan con los números $0, 1, 2, \ldots$

En este tema, estudiaremos cadenas de Markov de tiempo discreto, ya que son las que salen en la mayoría de sistemas de comunicación y análisis de datos.

5.3. Tiempo de espera

En una cadena de Markov, es el tiempo que estamos en un estado determinado.Dependiendo de como midamos este tiempo de espera, tenemos dos tipos de cadenas de Markov:

- **Cadenas de Markov de tiempo discreto:** Los tiempos de espera toman valores enteros. Por tanto, los cambios de estados pasan en tiempos discretos $t = T_0, T_1, T_2, \ldots$, Si estos tiempos están equiespaciados, tenemos $T_n = nT$, con $n = 0, 1, 2, \ldots$
- **Cadenas de Markov de tiempo continuo:** En este caso, los cambios de estado pueden tener lugar en cualquier instante de tiempo.

Ejemplo 1

Consideremos un buffer donde los paquetes llegan en cada paso de tiempo con una probabilidad a y se van con probabilidad c.Veamos que se trata de una cadena de Markov de tiempo discreto, lo de dicimos por que dice a cada paso de tiempo. El paso de tiempo sería el tiempo que se necesita para recibir o transmitir un paquete.Suponiendo que el buffer tiene un tamaño B, la tabla de estados sería la siguiente.

Estado	Significado
0	Buffer vacío
1	Hay un paquete en el buffer
2	Hay 2 paquetes en el buffer
$\vdots$	$\vdots$
B	El buffer está lleno

5.4. Falta de memoria de las cadenas de Markov

En una cadena de Markov de tiempo discreto el valor de la variable aleatoria $\mathrm{S}(n)$ representa el estado en el tiempo n.La variable aleatoria $\mathrm{S}(n)$ solo depende de la variable aleatoria $\mathrm{S}(n-1)$. Esta propiedad se denomina **falta de memoria de las cadenas de Markov**.La probabilidad de que $\mathrm{S}(n)$ tome el valor s_i solo depende de los valores s_j que toma $\mathrm{S}(n-1)$. Matematicamente.

$$p\{\mathrm{S}(n) = s_i\} = \mathrm{f}(s_j)$$

$\forall s_i, s_j \in \mathrm{S}$ donde S es el conjunto de estados del sistema (espacio muestral).

Ejemplo 2

Consideremos el buffer de un cierto dispositivo de comunicaciones. Supongamos que este dispositivo tiene una capacidad máxima de $B = 4$ paquetes. Los estados del buffer y las posibles transiciones entre ellos se pueden ver en el siguiente gráfico.

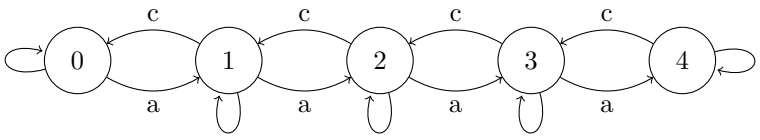

Figura 3: Estados del Buffer y las posibles transiciones entre ellos, Ejemplo 2

5.5. Matriz de transición

Definimos $\mathrm{p_{ij}}(n)$ la probabilidad de que en el tiempo actual n, nuestro sistema se encuentre en el estado i suponiendo que en el tiempo anterior estábamos en el estado j(tiempo $n-1$). Matematicamente

$$\mathrm{p_{ij}}(n) = p\{\mathrm{S}(n) = i|\, \mathrm{S}(n-1) = j\}$$

En el caso de que la probabilidad anterior $\mathrm{p_{ij}}(n)$ sea independiente del tiempo n, diremos que la cadena de Markov es **homogénea**. En este caso, podemos escribir

$$\mathrm{p_{ij}} = p\{\mathrm{S}(n) = i|\, \mathrm{S}(n-1) = j\}$$

Definimos $\mathrm{s_i}(n)$ la probabilidad de que en el tiempo n estemos en el estado i

$$\mathrm{s_i}(n) = p\{\mathrm{S}(n) = i\}$$

Haciendo uso del **teorema de las probabilidades totales** y suponiendo que la cadena de Markov es homogénea, podemos escribir

$$\mathrm{s_i}(n) = \sum_{j=1}^{m} \mathrm{p_{ij}} \cdot \mathrm{s_j}(n-1)$$

Suponiendo que los estados de nuestra cadena de Markov son $\{1, 2, 3..., m\}$, Matricialmente, podemos escribir la expresión anterior como

$$\mathrm{s}(n) = P \cdot \mathrm{s}(n-1)$$

Donde P es la llamada **matriz de transición** de la cadena de Markov homogénea.

$$P = \begin{pmatrix} \mathrm{p}_{11} & \mathrm{p}_{12} & \cdots & \mathrm{p}_{1\mathrm{m}} \\ \mathrm{p}_{21} & \mathrm{p}_{22} & \cdots & \mathrm{p}_{2\mathrm{m}} \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{p}_{\mathrm{m}1} & \mathrm{p}_{\mathrm{m}2} & \cdots & \mathrm{p}_{\mathrm{mm}} \end{pmatrix}$$

Cada elemento de la matriz P es la probabilidad de que tan probable es que yo termine en el estado i sabiendo que vengo del estado j, por ejemplo tomemos el elemento p_{21}, quiere decir la probabilidad de terminar en el estado 2 sabiendo que vengo del estado 1.

Definiremos $\mathrm{s}(n)$ el vector de probabilidades de la cadena de Markov en el tiempo n:

$$\mathrm{s}(n) = (\mathrm{s}_\mathrm{n}(1), \mathrm{s}_\mathrm{n}(2), \ldots, \mathrm{s}_\mathrm{n}(m))^t$$

Como que $\mathrm{s}(n)$ es un vector de probabilidades, se debe verificar

$$\sum_{i=1}^{m} \mathrm{s}_\mathrm{n}(i) = 1$$

El estudio de la matriz de transición P y más concretamente sus valores y vectores propios, nos determinará el comportamiento de nuestra cadena de Markov.

Cada columna j-ésima de la matriz de transición P representa las probabilidades de ir hacia cada uno de los estados i desde el estado j, para $i = 1, ..., m$. Por tanto,

$$\sum_{i=1}^{m} \mathrm{p}_{\mathrm{ij}} = 1$$

Las matrices que cumplan que la suma de sus columnas valen 1, se llaman **matrices estocásticas**.

5.6. Ejemplos de procesos de Markov

5.7. Ejemplo 3

Una fuente «On-Off» se utiliza frecuentemente en los sistemas de comunicacón para simular el tráfico de vos.Supongamos que esta fuente tiene dos estados; silencio(s_1), donde no se envía nungún dato; y activo(s_2), donde el sistema envía un paquete de datos por unidad de tiempo.

Si la fuente está en el estado s_1, hay una probabilidad s de que continúe en este estado y, si está en el estado s_2, hay una probabilidad a de que continue en este estado.

Se trata de una cadena de Markov con dos estados: s_1 y s_2 con matriz de transición.

$$P = \begin{pmatrix} s & 1-a \\ 1-s & a \end{pmatrix}$$

El **diagrama de transición** de la cadena de Markov es el siguiente.

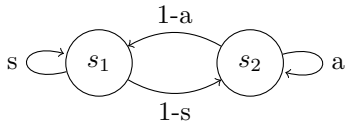

Figura 4: Diagrama de transición del Ejemplo 3

5.8. Ejemplo 4

Supongamos que la probabilidad de que un camión de reparto se mueva de una ciudad a otra al principio de cada dia se muestra en la figura siguiente. 1-Colwood, 2-Langford, 3-Sooke:

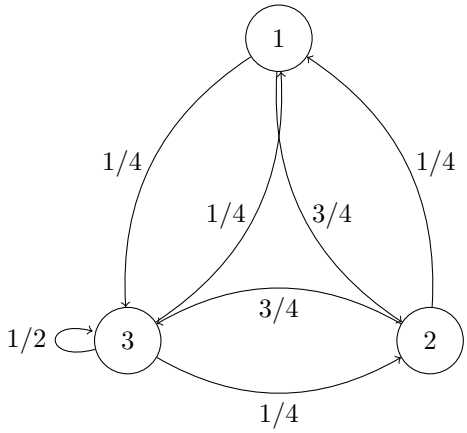

Figura 5: Diagrama de transición del Ejemplo 3

Vayamos a simular el movimiento diario del camión con una cadena de Markov. Esta cadena tendrá 3 estados que corresponderán a cada una de las 3 ciudades. La matriz de transición será

$$P = \begin{pmatrix} 0 & \frac{1}{4} & \frac{1}{4} \\ \frac{3}{4} & 0 & \frac{1}{4} \\ \frac{1}{4} & \frac{3}{4} & \frac{1}{2} \end{pmatrix}$$

Supongamos que al principio el camión se encuentra en la ciudad de Langford(Nodo 2).El vector de probabilidades iniciales s(0) será

$$\mathrm{s}(0) = (0, 1, 0)^t$$

Cada posición del vector pertenece a la probabilidad de estar en cada ciudad la primer posición del vector seria la pobabilidad de estar en 1-Coolwood, la segunda en 2-Langford, y la tercera en 3-Sooke, dado que decimos que en $n = 0$ tiempo inicial estamos en Langford es decir 1 % de probabilidad de estar ahi por eso vale 1 la pocisión del vector correspondiente a Langford y las demas en 0.

6. Matrices estocásticas

Diremos que una matriz P es estocástica o matriz de Markov si verifica las siguientes propiedades:

- La matriz P es cuadrada
- Los elementos $\mathrm{p_{ij}}$ de P están entre 0 y 1, $0 \leq \mathrm{p_{ij}} \leq 1$
- La suma de cada coluna de P vale 1: Si P es una matriz de orden m, entonces

$$\sum_{i=1}^{m} \mathrm{p_{ij}} = 1 \qquad \forall j = 1, \ldots, m$$

- Tienen el valor propio 1
- Todos los valores propios λ_i verifican $|\lambda_i| \leq 1$

7. Las diagonales de P

Vayamos a interpretar las diagonales de la matriz P estudiando un caso particular de Markov: las **colas**. En una cola, los clientes llegan y se van y los estados de la cadena de Markov son el numero de clientes que hay en la cola en cada momento. Una cola se caracteriza por el número medio de clientes que llegan por unidad de tiempo, el número de servidores, el tamaño de la cola y el número media de clientes que abandonan la cola por unidad de tiempo.

Las diagonales de P reflejan las características de la cola.

Diagonal	Significado
Principal	Probabilidades de que la cola tenga el mismo tamaño
Superior 1	Probabilidades de que la cola disminuya en un cliente
Superior 2	Probabilidades de que la cola disminuya en dos clientes
$\vdots$	$\vdots$
Inferior 1	Probabilidades de que la cola aumente en un cliente
Inferior 2	Probabilidades de que la cola aumente en dos clientes
$\vdots$	$\vdots$

8. Valores y vectores propios de P

8.1. Teorema

Sea P una matriz estocástica de orden m. Entonces,

- Cualquier valor propio de λ de P verifica $|\lambda| \leq 1$
- Si las componentes de la matriz P son estrictamente positivas ($p_{ij} > 0$) y $\lambda \neq 1$, entonces $|\lambda| < 1$

8.2. Teorema

Sea P una matriz estocástica y regular. Si P es diagonalizable y todas sus componentes son estrictamente positivas, $p_{ij} > 0$, entonces el valor propio tiene multiplicidad 1.

8.3. Teorema

Sea P una matriz estocástica. Sea x un vector propio del valor propio $\lambda \neq 1$. Entonces la suma de las componentes de x es 0: $\sigma(x) = 0$

8.4. Teorema

Sea P una matriz estocástica y diagonalizable de orden m. Sea x un vector propio de valor propio 1. Entonces, la suma de las componentes de x no es 0:$\sigma(x) \neq 0$

8.5. Colorario

Sea P una matriz estocástica y diagonalizable. Siempre podemos escoger un vector propio x de valor propio 1 tal que la suma de sus componentes sea 1: $\sigma(x) = 1$

8.6. Comportamiento transitorio

Recordemos que s_n representa el vector de probabilidades de la cadena de Markov en el intante n, Por tanto, si el vector s_1 será en función del **vector inicial de probabilidades** $s_0 : s_1 = P\, s_0$

Del mismo modo, $s_2 = P\, s_1 = P^2\, s_0$ y en general, tendremos que el vector de probabilidades en el instante n será

$$s_n = P^n\, s_0 \qquad n = 1, 2, ...$$

8.7. Proceso de decisión de Markov - MDP

Proporciona un marco matemático para modelar la toma de decisiones en situaciones en las que los resultados son en parte aleatorios y en parte están bajo el control de quien toma las decisiones.

Un **MDP** modela un problema de decisión sequencial en donde el sistema evoluciona en el tiempo y es controlado por un agente. La dinamica del sistema esta determinada por una función de transición de probabilidad que mapea estados y acciones a otros estados.

Es decir son la forma idealizada matemáticamente del problema de aprendizaje por refuerzo, para la cualse podria encontrar un enunciado teórico preciso que pueda describirla, en otras palabras **MDP** describen formalmente el medio ambiente en el cual se desarrolla el **aprendizaje por refuerzo**, donde el medio ambiente es completamente observable, esto da como consecuencia que la mayoria de problemas de **aprendizaje por refuerzo** se puedan formalizar como **MDPs**. Con los **MDP** se introducen varios elementos clave para la descripción del problema como el retorno, funciones de valor y las **ecuaciones de Bellman**. Simplificando, un proceso de Markov es un proceso sin memoria y aleatorio, en otras palabras **secuencia de estados aleatorios que posee la propiedad de Markov**, podemos definir el proceso como la siguiente tupla.

$$MDP =< S, P >$$

- S Lista de estados
- P Matriz de transición de estados

8.8. Procesos de Recompensa de Markov

Los procesos de recompensa de Markov se pueden ve como procesos de Markov normales con valores que juzgan que tan positivo es estar en un estado, estro traería un cambio a la definición de procesos de Markov que tuvimos previamente agregándole dos nuevas variables.

$$MDP =< S, P, R, \gamma >$$

- S Lista de estados
- P Matriz de transición de estados
- R Es la recompensa inmediata en el estado donde nos encontraríamos
- γ Es un factor de descuento, $0 \leq \gamma \leq 1$

Los procesos de desición de Markov poseen la misma forma que un procesos de recompensa de Markov pero en este caso se le agregan decisiones que van a ser tomadas por nuestro agente, es decir que ya no se moverá libremente en el sistema.

$$MDP =< S, A, P, R, \gamma >$$

- S Lista de estados
- A Es una lista de acciones
- P Matriz de transición de estados
- R Es la recompensa inmediata en el estado donde nos encontraríamos
- γ Es un factor de descuento, $0 \leq \gamma \leq 1$

En este caso se agregará una lista de acciones que se tomará, debido a que no sera un problema en el cual el agente se moverá aleatoriamente en la cadena de Markov si no mas bien será una serie de decisiones que nos llevarán a hallar la mayor recompensa.

8.9. Métodos de solución básicos

Existen tres formas principales de resolver **MDPs**

- Usando métodos de programación dinámica - Ecuaciones de Bellman.
- Usando métodos de Monte Carlo.
- Métodos de diferencias temporales o de aprendizaje por refuerzo.

9. Ecuación de Bellman con procesos estocásticos

$$\mathrm{V}(s) = \max_a(\mathrm{R}(s,a) + \gamma \cdot \sum_{s'} P(s,a,s') \cdot \mathrm{V}(s'))$$

Ahora multiplicamos los valores esperados por las probabilidades así conseguimos el valor esperado de entrar en cualquier estado como ponderación de las posibilidades. Es decir que el factor $\sum_{s'} P(s,a,s')$ es el ponderado de las sumas de las probabilidades sobre todas las acciones y estados posibles, con este factor

conseguimos que no sabremos cual es el valor de entrar en algún estado, en la primer forma de la ecuación de Bellman era fijo, ahora es una ponderación de lo que obtendré si tomo una decisión a partir del estado actual, así pasamos de un proceso determinista a un proceso estocástico.

Es decir que la probabilidad $P(s, a, s')$ de ir de un estado s con una acción a a un estado s' influirá o modificara el valor final.

10. Política o directivas

En el contexto del aprendizaje por refuerzo, es el conjunto de reglas o que utiliza un agente para decidir qué acciones llevar a cabo. Es más, ya que la política viene a ser como el cerebro del agente a menudo se sustituye la palabra política por agente.

10.1. Tipos de políticas

- **Políticas deterministas o plan:** En las cuales la acción llevada a cabo por el agente en cualquier estado vendrá siempre dada de forma determinista como la salida de la politica. Si no se tiene claro el significado de determinista, quiere decir que, dadas unas entradas concretas, es posible calcular o predecir un resultado concreto. Estas políticas se suelen denotar mediante μ.

  $$\alpha_t = \mu(s_t)$$

  La ecuación significa que, una acción α en un momento temporal concreto t, viene dada por la politica μ del estado s en ese momento.

- **Políticas estocásticas:** Vienen dadas por distribuciones de probabilidad condicionadas y se denotan con el simbolo π. En ese caso, la probabilidad de llevar acabo una acción α en un momento temporal concreto t, estando en el estado s, viene dada por:

  $$\alpha_t \sim \pi(\cdot|s_t)$$

En una política determinista, está clara la acción llevada a cabo en un momento dado y para un estado concreto. Por el contrario, en una política estocástica, las acciones llevadas a cabo no están completamente determinadas, sino que tienen una probabilidad asociada. Además en **aprendizaje por refuerzo** se habla de políticas parametrizadas.

- **Políticas parametrizadas:** Son aquellas que tienen como salida una función que depende de un conjunto de parámetros, los cuales podemos ajustar mediante algún algoritmo de optimización. Por ejemplo, una política parametrizada podría depender de los pesos de una red neuronal.

Normalmente, denotamos los parámetros de una política con la letra θ o con la letra ϕ, es decir las ecuaciones quedan de la siguiente manera:

$$\alpha_t = \mu_\theta(s_t)$$
$$\alpha_t \sim \pi_\theta(\cdot|s_t)$$

11. Factor de Penalización

Es un sistema de recompensa intermedio, anteriormente el agente solo recibía una recompensa positiva al terminar el laberinto o negativa al caer en un estado donde el agente se quema, pero en todos los demás estados intermedios no recibía ninguna recompensa, con este sistema de recompensa los que conseguimos es que la **IA** juegue y se arriesgue. Imaginemos que en cada estado intermedio definimos una recompensa chica y negativa como $R = -0{,}04$, tratara de ir por el camino más rápido en lugar de ir al estado menos peligroso.

Si suponemos que para cada estado intermedio definimos una $R(s) = -2$ y en el estado final una recompensa de $R = 1$ y en el estado donde el agente se quema una $R = -1$, en los estados contiguos al estado donde el agente se quema, la política de acción seria ir directamente al fuego pues sale mas caro seguir viviendo para ir al estado final que simplemente morir. Por esta razón al factor de penalización también se le conoce como **Living Penalty**. Así vemos pues que como dependiendo del factor de penalización que tomemos las políticas de actuación van cambiando según el factor que decidamos.

12. Q-learning

El Q-learning es un algoritmo de aprendizaje basado en valores y se centra en la optimización de la función de valor según el entorno o el problema. La Q en el Q-learning representa la calidad con la que el modelo encuentra su próxima acción mejorando la calidad. El proceso puede ser automático y sencillo. Esta técnica es increíble para comenzar con aprendizaje por refuerzo. El modelo almacena todos los valores en una tabla, que es la Tabla Q. En palabras simples, se utiliza el método de aprendizaje para la mejor solución.

Anterior mente para cada estado s calculábamos el valor $\mathrm{V}(s)$ con la ecuación de Bellman, sin embargo el valor Q interactúa sobre estados s y acciones a es decir $\mathrm{Q}(s, a)$, es decir a partir de un estado s llevo a cabo la acción a, $\mathrm{Q}(s, a)$ mide la calidad de la acción a partir del estado.$\mathrm{Q}(s, a)$ podemos tomarla como una medida de cuantificación de la acción a para poder compararlas y saber cual es mejor.$\mathrm{V}(s)$ y $\mathrm{Q}(s, a)$ quedan de alguna manera relacionados de la siguiente forma.

$$\mathrm{Q}(s,a) = \mathrm{R}(s,a) + \gamma \cdot \sum_{s'} \mathrm{P}(s,a,s')(s')$$

La calidad de una acción a a partir de un estado s, tiene que ver con la recompensa directa $\mathrm{R}(s,a)$ mas el factor de descuento por el valor esperado de llevar acabo dicha acción a.

Ahora podemos reescribir la ecuación de Bellman de la siguiente manera:

$$\mathrm{V}(s) = \max_a(\mathrm{R}(s,a) + \gamma \cdot \sum_{s'} P(s,a,s') \cdot \mathrm{V}(s')) = \max_a \mathrm{Q}(s,a)$$

$$\mathrm{V}(s) = \max_a \mathrm{Q}(s,a)$$

La ecuación de Bellman de la forma $\mathrm{V}(s) = \max_a(\mathrm{R}(s,a)+\gamma\cdot\sum_{s'} P(s,a,s')\cdot \mathrm{V}(s')$ puede ser algo molesto pues es recursiva asi pues reescribiendola con $Q(s,a)$ simplificamos la notación y es mas significativo pues una cosa es el valor $\mathrm{V}(s)$ y otra es la calidad $\mathrm{Q}(s,a)$, ahora teniendo en cuenta que:

$$\mathrm{V}(s) = \max_a \mathrm{Q}(s,a)$$

podemos definir $\mathrm{Q}(s,a)$ de la siguiente forma:

$$\mathrm{Q}(s,a) = \mathrm{R}(s,a)+\gamma\cdot\sum_{s'} \mathrm{P}(s,a,s')(s') = \mathrm{R}(s,a)+\gamma\cdot\sum_{s'}(\mathrm{P}(s,a,s')\cdot\max_{a'} \mathrm{Q}(s',a'))$$

$$\mathrm{Q}(s,a) = \mathrm{R}(s,a) + \gamma \cdot \sum_{s'}(\mathrm{P}(s,a,s') \cdot \max_{a'} \mathrm{Q}(s',a'))$$

Al hacer ese reemplazo lo que hacemos es que las acciones que tomamos desde un estado al siguiente solo dependan de la calidad de las mismas, al final vemos que el valor esta relacionado con la calidad y la validad esta relacionada con el valor. Así tenemos una definición recursiva de $V(s)$ y una definición recursiva de $Q(s,a)$ parece que no hemos arreglado nada pero la realidad es que hemos arreglado muchísimo por que ahora el agente puede pensar cual es el valor $V(s)$ del estado actual y también puede pensar cual es la calidad de estar en ese estado y tomar la acción, puede tomar en cuenta las cosas por separado lo cual es muy interesante para poderlo aplicar a nuestro tema de inteligencia artificial.