TP: Q-learning

immediate

20 de noviembre de 2016

1. Introducción

En este trabajo exploramos el algoritmo de Q-learning para entrenar jugadores virtuales del juego 4 en línea. El algoritmo consiste básicamente en explorar la mayor cantidad de configuraciones del juego posibles, penalizando o recompensando las distintas acciones que realiza los jugadores. Al final del entrenamiento, los jugadores virtuales tienen una valorización de las acciones posibles dada una configuración del juego.

1.1. Juego 4 en línea

El juego de 4 en línea es un juego de mesa para dos jugadores que consiste en introducir fichas en un tablero vertical, con el objetivo de colocar 4 fichas consecutivas del color correspondiente, ya sea en forma vertical, horizontal, o diagonal. Gana el primer jugador que alcanza esa configuración, y en caso que el tablero se complete antes de que algún jugador lo logre, se produce un empate. Típicamente el tablero tiene un tamaño de 6 filas por 7 columnas, lo que da un total de 4.531.985.219.092 configuraciones posibles.

2. Algoritmo de Q-learning

El algoritmo consiste en explorar todas las configuraciones posibles del juego. A medida que se produce el entrenamiento, los jugadores actualizan una magnitud $Q(s, a)$ que representa la ganancia o valorización de realizar la acción a en el estado s. Cuando un jugador realiza una acción que lo lleva a lograr un 4 en línea, este recibe una recompensa, y esto lleva a valorizar todas las acciones que lo llevaron a estar cerca de dicho estado.

Al finalizar el entrenamiento, teóricamente la mejor estrategia es, dado un estado, elegir la acción que maximize Q.

2.1. Estructura de datos

Representamos la ganancia $Q(s, a)$ como un diccionario de *python*, donde cada entrada del diccionario es un estado, representado con una variable *string* de 42 caracteres de largo asociada, por ejemplo, la *string* '12000....0' representa el estado donde el jugador 1 tiene una ficha en la primer columna, el jugador 2 en la segunda, y el resto del tablero está vacío. Dado un estado, Q tiene una valorización de cada acción posible, que en principio son alguna de las 7 columnas del tablero, salvo que se encuentre en una configuración donde una de las columnas esté completa. Los valores de $Q(s, a)$ se incializan con un valor aleatorio entre 0 y 1.

2.2. Representación del problema

Decidimos pensar el juego de dos jugadores como la evolución de un único sistema, que puede encontrarse en cualquiera de los estados mencionados en la introducción. En vez de distinguir entre dos jugadores, observamos que dado un estado del sistema, queda inmediatamente determinado cuál de los dos jugadores puede jugar. Así al inicio del juego, con el tablero vacío, las acciones posibles entre las que puede elegir el sistema consisten en colocar una ficha del jugador 1 en cualquiera de las 7 columnas. En el estado siguiente, las acciones posibles son colocar necesariamente una ficha del jugador 2 nuevamente en cualquiera de las 7 columnas, y así siguiendo. Es decir, si el sistema tiene una cantidad par de fichas, necesariamente juega el jugador 1 en alguna de las columnas disponibles.

Por otro lado, consideramos que una correcta actualización de la ganancia $Q(s, a)$ viene dada por la ecuación 1, donde $R(s, a)$ es la recompensa de realizar una acción a en el estado s, s' es el estado obtenido al realizar a en s, y a' es cualquiera de la acciones posibles en el estado s'.

$$Q^{n+1}(s, a) = Q^n(s, a) + \alpha(R(s, a) - \gamma(max_{a'} Q^n(s', a') - Q^n(s, a)) \quad (1)$$

Dado un estado, la valorización de una acción:

- aumenta si al realizar una acción, de esta se obtiene una recompensa (en este caso se gana el juego)
- disminuye si la mejor acción del siguiente estado (que corresponde al otro jugador) está muy valorizada. Es decir cada jugador trata de tomar acciones que no lleven a un estado donde el otro jugador tenga una muy buena jugada.

Pensándolo de esta manera, en el algoritmo solo se introducen recompensas positivas. Cuando un jugador gane y reciba una recompensa, el otro jugador desvaluará la acción que llevó al sistema a ese estado, y valorará más las otras.

2.3. Exploración de los estados: temperatura.

Durante la fase de entrenamiento no hay ninguna restricción al elegir las acciones que el sistema decida explorar, estas pueden ser escogidas totalmente al azar. Sin embargo podemos introducir una magnitud que oficie de temperatura del sistema. Cuando la temperatura del sistema sea alta, dado un estado el sistema elije con equiprobabilidad cualquiera de las acciones posibles. Si la temperatura es exactamente cero, el sistema elije la acción que maximize Q.

Con el parámetro de la temperatura, podemos realizar un entrenamiento que en un principio explora lo más aleatorio posible, pero al disminuir la temperatura el sistema preferirá realizar acciones con valorización alta, de esta manera ponderará las buenas acciones.

2.4. Modificaciones agregadas

ACA IRIAN LAS COSAS QUE LE PUSIMOS DESPUES, DE QUE GANE EL JUEGO SI TIENE 3, ETC...

3. Resultados

4. Conclusiones