Memoria Práctica 3: Agentes en un entorno con adversario

El objetivo de la práctica era la implementación de un bot que simulase a un jugador para competir a Mancala.

En esta memoria se van a detallar tanto la implementación del algoritmo utilizado como del diseño del "estado", así como la heurística utilizada.

En primer lugar, cabe reseñar que no se ha modificado la estructura inicial de "estado" y por tanto su implementación es idéntica a la inicialmente propuesta. **No se ha hecho uso** de **estructuras de datos adicionales**.

Por otra parte, se ha elegido el **algoritmo de la poda alpha-beta** para su implementación en el bot, pues se obtiene un beneficio en tiempo y cómputo sensible, permitiendo alcanzar una mayor profundidad a la hora de explorar estados futuros. Se ha optado por un diseño sencillo del algoritmo.

- Como caso base tenemos que el algoritmo llegue a un estado que sea final o que haya alcanzado la profundidad máxima, en cuyo caso evaluamos ese estado y lo devolvemos.
- En otro caso, y en función de si el estado explorado es min o es max, se generan igualmente los hijos simulando el movimiento (previa criba de movimientos válidos, que en general viene determinado porque haya una semilla en el semillero correspondiente), y posteriormente se llama recursivamente al algoritmo, podando en función del valor que devuelva la función y del valor de alpha y beta que haya en ese momento.
- Cabe remarcar que **el primer nivel del árbol se genera fuera del algoritmo** poda alpha-beta siempre, porque se necesita devolver en último término el movimiento más oportuno que debe realizar nuestro bot. De esta forma, no almacenamos el camino durante el algoritmo de la poda alpha-beta y ahorramos en cómputo, pues no necesitamos conocer ni guardar el camino que se ha seguido, **sólo queremos conocer el próximo movimiento**.

Por último, se pasa a detallar la heurística.

En general, se han estimado situaciones que se quieren premiar o que se quieren penalizar a la hora de analizar el estado del tablero en un momento futuro. Los valores que definitivamente tiene cada parámetro son meramente estimativos y basados en la empírica.

I. El primer paso de la heurística de nuestro bot es conocer si actuamos como jugador1 o como jugador2, para poder evaluar **los estados** como **propios** o como los del **adversario**.
II. Una vez dilucidado esto, se estiman unos valores para cada semillero en función del número de semillas que haya en ellos. En el caso de nuestro bot, no le interesa

demasiado tener semilleros vacíos ni acumular demasiadas semillas en un mismo semillero, para **evitar quedarse sin movimientos y ser robado** respectivamente.

III. Para el adversario premiamos lo contrario, nuestro bot prefiere que tenga semilleros vacíos, pero no está especialmente satisfecho con la posibilidad de que el adversario acumule demasiadas semillas, así que lo penalizamos en función del número que acumule, cuantas más acumule el adversario, más penalizado está.

Para reforzar este comportamiento, contamos con cálculos adicionales:

a. Primero, contabilizamos el número de semillas en nuestro poder y en el del adversario, no queremos que el adversario nos saque ventaja y acabemos sin movimientos a largo plazo, así que **premiamos tener más semillas que él en nuestros semilleros**.
b. Segundo, como acumular semillas puede acarrear indeseados robos por parte de nuestro adversario, contabilizamos el número de semillas en el granero en el futuro, penalizando fuertemente el número de semillas que el adversario tiene en su granero, de tal modo que **podamos predecir** en mayor o menor medida **si nuestro bot ha sido víctima de un robo** y castigarlo.

IV. Por último, y básico, nuestro bot se pregunta si **el estado** al que ha llegado **representa una victoria o una derrota** para él, variando la estimación de ese nodo de tal forma que el estado del tablero pase a ser trivial.

V. Cabe destacar que en cierto momento del desarrollo se tuvo en cuenta y se premiaba que nuestro bot repitiera turno, pero se descubrió que en muchos casos era contraproducente y realmente tiene cierta lógica si pensamos que estamos evaluando un tablero a unos 9 movimientos vista y que las posibilidades de que se alcance cierto estado no son lo suficientemente destacables. En una evaluación más a corto plazo probablemente hubiera tenido más sentido tener ese parámetro en cuenta. Por tanto, nuestro bot desecha esta posibilidad y no la evalúa en concreto.