Universidad de Buenos Aires

Facultad de Ciencias Exactas y Naturales

Departamento de Computación

Algoritmos y Estructuras de Datos III

Trabajo Práctico 3

Autores:
Nicolás Chehebar, mail: *nicocheh@hotmail.com*, LU: 308/16
Matías Duran, mail: *mato_ fede@live.com.ar*, LU: 400/16
Lucas Somacal, mail: *lsomacal@gmail.com*, LU: 249/16

Índice

1. El Juego

1.1. Descripción

El juego es una generalización del popular 4 en línea [1]. Consiste en una grilla de M filas y N columnas en la cual dos jugadores colocan alternadamente una ficha propia (identificada con rojo las de un jugador y azul las del otro). Las fichas se pueden colocar en cualquier columna de la grilla y una vez elegida la columna, esta determina el movimiento, ya que irá a la fila de "más abajo" (la de numeración más baja) que esté desocupada. El objetivo de un jugador será lograr tener un línea recta (diagonal, vertical u horizontal) de C fichas propias. Cuando esto suceda, el jugador ganará el partido. Además, cada jugador dispone de P fichas. En caso de que ambos se queden sin fichas y ninguno haya ganado, la partida finaliza en empate. También se da un empate si la grilla queda llena (y ninguno había ganado). Se trata de una generalización del 4 en línea ya que si tomamos parametros $M = 6, N = 7, C = 4, P = 21$ se replicarían las condiciones iniciales del juego.

1.1.1. Ejemplos

2. Jugador Óptimo

2.1. El algoritmo

El algoritmo del jugador del punto 1.a brinda un jugador óptimo. Nos asegura que este jugador hará la estrategia ganadora si hubiera. En caso de que esta no exista, realizará una de empate. Y si tampoco existiera esa, jugará indistintamente sabiendo que perderá. Este análisis se realiza jugada a jugada. Para ejemplificar esto, podría suceder que el oponente tenga la estrategia ganadora y en ese caso nuestro jugador hará cualquier movida indistintamente (pues sabe que perderá), pero si en la próxima jugada el oponente no realiza la correspondiente a su estrategia ganadora y da un nuevo estado del tablero en el que esta vez nuestro jugador tiene estrategia ganadora, nuestro jugador jugará y ganará ya que ahora sí tiene estrategia ganadora.

Para lograr esto, utilizamos una técnica algorítmica similar al Backtracking en el sentido de que exploramos todas las soluciones posibles y nos quedamos con la óptima. Pero esta vez tenemos dos jugadores interviniendo en la situación donde lo que uno busca es todo lo contrario a lo que busca el otro. Podemos decir que un tablero finalizado tiene 3 puntajes posibles, 1 si ganamos nosotros, 0 si es empate, -1 si ganó el otro (podríamos sino generalizarlo para todo tablero y que haya un cuarto valor que sea inválido si aún no hemos calculado el valor de dicho tablero). Así, lo que sabemos es que turno a turno, uno quiere maximizar el puntaje y el otro minimizarlo. Es por esto que dicha técnica algorítmica se llama Minimax.

De esta manera, igual que en Backtracking tenemos un arbol de ejecución donde cada nodo es un estado del tablero y la raiz es el tablero vacío. Cada nodo (que no sea hoja) tendrá N hijos donde cada uno representará que la próxima jugada fue en alguna de las N columnas. De esta forma, recorremos todos los tableros posibles. Según quién comience, en el primer nivel trataremos de maximizar o minimizar, en el siguiente lo contrario y así sucesivamente. Todos los niveles impares minimizarán si empieza el contrincante y maximizarán si empieza nuestro jugador. El que maximiza le asignará a su nodo un puntaje que será el máximo de los puntajes de todos sus hijos. Análogamente el que minimiza le asignará a su nodo un puntaje que será el mínimo de los puntajes de todos sus hijos.

Así, ejecutando dicho algoritmo la raíz tendrá la información de quién tiene la estrategia ganadora, o que ambos pueden asegurar el empate según quién empiece y haya un 1, 0 o -1. Esta es la idea general del algoritmo, lo veremos más claro en el siguiente pseudocódigo

[1] $https://es.wikipedia.org/wiki/Conecta_4$

2.1.1. El Pseudocódigo

2.2. La poda alfa-beta

2.2.1. El algoritmo

2.2.2. El Pseudocódigo

2.3. Complejidad

2.4. Experimentación

2.4.1. Contexto

2.4.2. Sin poda

En principio, para verificar experimentalmente que el jugador era óptimo se jugo en tableros pequeños (particularmente de $2\times2, 3\times3, 4\times4, 2\times3, 3\times2, 2\times4, 4\times2, 3\times4, 4\times3, 5\times3$) con un $c = 2, 3, 4$ siempre que cumpla la condicion del juego ($c \leq minN, M$) contra el jugador random. En todos los casos la cantidad de iteraciones fue de $2 * N^(N * M)$ por lo que todo tablero posible tenía una probabilidad considerable de ser jugado (notar que la cantidad de juegos posibles es menor que $N^(N * M)$ ya que a cada paso lo que se elije es una de N -o menos si hay columnas llenas- opciones donde jugar y esto se repite hasta que termine el juego -a lo sumo se lena la grilla $N * M$ veces-) y como el jugador es random, la distribución respecto de los tableros es uniforme. Más aún, se puede notar que solo queremos contar las opciones que puede dar el segundo jugador (en este caso el aleatorio), lo que nos provee una cota menor de la cantidad total de jugadas distintas que se le pueden hacer al jugador optimo, acotandolo por $N^(N * M/2)$.

Estos experimentos se repitieron en dos casos cada uno, cuando comenzaba el minimax y cuando comenzaba el random. Además para fijado el jugador que empieza, se repitió con valores de fichas que fueron $p = N * M/2, N * M/3, N * M/4$ donde el primer valor de p aseguraba que se pueda jugar sin limite de fichas (siempre se podría llegar a llenar el tablero) y los otros dos sí imponian un limite de fichas. Luego de cada experimento, nos fijamos en el archivo .log devuelto por dichas iteraciones y en todos se observo el mismo comportamiento:

- O bien siempre ganaba (lo que nos indica que había estrategia ganadora para el que empieza). Al invertir la situación (cambiar el jugador que empieza), se daba el item 3.
- O bien siempre ganaba o empataba (lo que nos indicaria que la mejor estrategia para ambos resulta en un empate, pero como el jugador random no siempre juega lo mejor, le daba la posibilidad a nuestro jugador de ganar). Al invertir la situación (cambiar el jugador que empieza) se obtenía este mismo item.
- O bien siempre ganaba o empataba o perdía (lo que nos indicaría que el que no empieza tiene estrategia ganadora, pero como el jugador random no siempre juega lo mejor, le daba la posibilidad a veces a nuestro jugador de ganar). Al invertir la situación (cambiar el jugador que empieza), se daba el item 1.

Como se explico, estas situaciones, en todos estos tableros reforzaron fuertemente la idea de que se trataba de un jugador óptimo. Cabe aclarar que se realizo solo con tableros pequeños debido a que para tableros muy grandes el jugador demoraba demasiado tiempo en decidir que jugar (como se vio en la complejidad teórica exponencial, por lo que crecía brutalmente al crecer el tamaño del tablero y la cantidad de columnas) y por ende resultaba inviable realizar una alta cantidad de iteraciones para recorrer una gran cantidad de tableros posibles lo que nos permita reforzar la idea de optimalidad del jugador.

Pero en tableros pequeños hemos podido comprobar que en todos los tableros que se dieron, cumplieron que el jugador era óptimo ya que si tenía la posibilidad de ganar lo hacía, en caso de no existir esta, si tenía la posibilidad de empatar lo hacía y recién en caso de no existir esta, jugaba cualquier cosa sabiendo perdería. Pero como hemos visto, esto se comprobaba paso a paso, por lo que el jugador podía jugar

creyendo que perdería y como el otro no jugo optimamente luego, pasar a poder ganar o empatar (y efectivamente hacerlo pues es optimo); o sea, asume optimalidad del rival.

Esta experimentación nos permitió reforzar nuestra idea (al menos para estos tableros pequeños y con una probabilidad muy alta) de que el jugador era efectivamente óptimo.

MEDIR EL TIEMPO Y VER QUE CUMPLE CON LA COMPLEJIDAD PROPUESTA.

2.4.3. Con poda

En el caso con poda se realizo la misma experimentación para comprobar la optimalidad. Como el algoritmo tardaba un poco menos (por la poda realizada), se realizó (además de con los ya mencionados) con tableros de tamaño $4 \times 5, 5 \times 4, 6 \times 3, 3 \times 6, 7 \times 3 y 3 \times 7$, con los mismos valores de c que antes (y siempre y cuando cumpliera las condiciones del juego). También se varió la cantidad de fichas de la misma forma.

Nuevamente, los resultados y conclusiones sobre estos fueron las mismas, lo que nos permitió reforzar nuestra idea (al menos para estos tableros pequeños y con una probabilidad muy alta) de que el jugador era efectivamente óptimo.

MEDIR EL TIEMPO Y VER QUE CUMPLE CON LA COMPLEJIDAD PROPUESTA IGUAL PERO QUE MEJORA CLARAMENTE AL SIN PODA.

3. Jugador parametrizable

4. Grid Search

5. Algoritmo Genético

6. Análisis comparativo con paper