El juego del Número Mágico y su solución con algoritmos de grafos

Daniel Felipe Vargas Ulloa

Junio 1, 2023

1 Introducción

El juego del Número Mágico es un desafiante juego en el que el objetivo es encontrar un camino que sume un número objetivo específico, llamado 'el número mágico'. El juego toma lugar en un tablero cuadrado de tamaño $n \times n$, donde cada casilla contiene un valor numérico entre 1 y 9. En cada turno, el jugador podrá moverse una casilla en cualquier dirección dentro del tablero. Además, el jugador comienza en la casilla superior izquierda con un valor acumulado de 0. El objetivo es encontrar un camino a través del tablero que sume exactamente el número mágico.

2 Solución aplicando conceptos del curso

2.1 Hacer uso de recorridos de grafos

Para resolver este difícil juego, podemos utilizar algoritmos de navegación grafos vistos en el curso. En particular, el algoritmo de búsqueda en profundidad DFS resulta ser una opción adecuada para encontrar un camino que cumpla con el objetivo.

Sin embargo, se requieren algunas modificaciones en el algoritmo de DFS para adaptarlo al juego del Número Mágico. En cada paso del algoritmo, el jugador debe moverse a una casilla vecina en el tablero y acumular el valor de la casilla visitada al valor acumulado. A su vez, se deben considerar algunas restricciones, como no visitar casillas previamente visitadas o fuera del tablero, además de no superar el número objetivo.

La ejecución del algoritmo se realiza de la siguiente manera. Comenzando desde la casilla inicial superior izquierda, se realizan movimientos a las casillas vecinas en las cuatro direcciones (arriba, abajo, izquierda, derecha) de manera recursiva. En cada movimiento, se verifica si se ha alcanzado el objetivo sumando el valor de la casilla actual al valor acumulado. Si se alcanza el objetivo, se

ha encontrado un camino válido. Si no se encuentra un camino válido, regresamos al estado anterior a nuestro último movimiento, y evaluamos los demás movimientos posibles, buscando recursivamente una solución hasta encontrarla.

2.2 Propuestas alternativas para solucionar el problema

Alternativamente, sería posible formular una solución que, en lugar de hacer uso de algoritmos en grafos, haga uso de programación dinámica. Una gran motivación para resolver el problema de esta forma nace del hecho de que para un tablero con un dado número mágico, podemos formular nuestro siguiente movimiento como un subproblema similar, donde ahora queremos alcanzar el mismo número mágico menos el valor de la casilla que acabamos de ocupar.

No obstante, dada la facilidad de implementar un algoritmo DFS, resulta innecesario hacer uso de programación dinámica para este juego. Por otra parte, si el juego nos pidiese llegar al número mágico en el menor número de pasos, no podríamos utilizar nuestro algoritmo DFS como solución a este problema, y tendríamos que buscar alternativas como implementar una búsqueda en anchura como BFS o una solución que haga uso de programación dinámica.

3 Comparación con el enfoque de una persona sin conocimiento de algoritmos

Si una persona sin conocimiento de algoritmos se enfrenta al juego del Número Mágico, probablemente seguirá un enfoque de prueba y error. Esta persona podría realizar movimientos aleatorios en el tablero, sumando los valores de las casillas visitadas y verificando si se alcanza el número objetivo. Sin embargo, este enfoque es altamente ineficiente a la hora de resolver el problema, ya que puede requerir de realizar los mismos cálculos varias veces y en un peor caso tendríamos que evaluar todos los caminos posibles.

El uso de algoritmos de grafos, como el DFS modificado, es esencial para abordar eficientemente el problema del Número Mágico. Estos algoritmos permiten explorar sistemáticamente todas las posibles combinaciones de movimientos en el tablero, evitando movimientos repetidos y descartando caminos que excedan el número objetivo. Con el enfoque algorítmico, se puede encontrar el camino óptimo rápidamente y sin repetir los cálculos realizados, lo cual es muy difícil de lograr con métodos de prueba y error.

4 Casos de ejemplo

4.1 Ejemplo 1: Un tablero fácil

■	6	8	9
7	2	6	1
6	6	7	1
9	6	4	3

Para este tablero, se tiene el objetivo de crear un camino de costo 20. Para este caso sencillo, es fácil encontrar una solución correcta por medio de prueba y error. No obstante, podemos encontrar el camino que resuelva este problema haciendo uso de nuestra versión modificada de DFS, la cual procederemos a explicar a continuación.

Comenzando desde la casilla inicial superior izquierda, evaluamos el camino generado al ir en una de las cuatro posibles direcciones en las que se puede mover el jugador, registrando el valor total del camino a medida que se recorre. Realizamos este proceso recursivamente, continuando siempre y cuando nuestro camino no haya alcanzado el número mágico, y rechazando caminos que se pasen del número mágico o no tengan a donde moverse. En caso de no poder continuar el camino, nos devolvemos al estado antes de agregar esa casilla y probamos con otra las posibles direcciones de movimiento. Este proceso iterativo se detiene cuando se encuentra un camino que sume el número mágico. Nótese que pueden existir varios caminos que cumplan con esta condición.

Para este problema, podemos encontrar la siguiente solución, la cual puede ser encontrada por una persona en un par de minutos, pero encontramos inmediatamente haciendo uso de los algoritmos de grafos vistos en clase. Tenemos entonces la siguiente solución, marcando las casillas visitadas con un '■':

■	6	8	9
■	■	■	■
6	6	7	■
9	6	4	■

4.2 Ejemplo 1: Un tablero regular

■	2	4	6	1	3
6	6	3	4	8	4
2	8	7	3	4	3
8	3	9	3	8	3
9	7	1	6	6	4
8	5	8	1	5	7

Para este tablero, se tiene el objetivo de crear un camino de costo 60. A partir de este ejemplo, podemos ver como este problema comienza a ser difícil para una persona a medida que el tablero escala en tamaño, ya que realmente no existe una mejor estrategia para este juego que nos acerque a la respuesta correcta, por lo que sin hacer uso de algoritmos de recorrido de grafos o programación dinámica resulta realmente complicado encontrar una respuesta correcta, resultando en un puzzle difícil e interesante para aquellos que desconocen los algoritmos de recorrido de grafos. Utilizando nuestro algoritmo de DFS adaptado a este problema, obtenemos la siguiente solución:

■	■	■	■	■	■
6	6	3	4	8	■
2	8	7	3	4	■
8	■	9	3	8	■
9	■	■	■	6	■
8	5	8	■	■	■

4.3 Ejemplo 1: Un tablero difícil

■	8	9	3	3	2	5	4	6	1
9	6	8	1	5	5	3	8	7	7
9	5	2	7	1	5	7	1	3	3
5	7	9	4	8	3	4	9	1	1
4	9	2	3	5	4	1	8	5	2
6	7	5	5	1	3	6	7	4	1
9	1	3	6	3	8	8	9	7	2
7	4	4	9	1	5	1	3	4	1
1	9	3	6	4	8	1	9	4	5
6	9	1	9	9	2	7	4	2	5

Para este tablero, se tiene el objetivo de crear un camino de costo 180. Este tablero presenta un mayor desafío, ya que el camino buscado será mucho más largo que los anteriores. Dado que el tamaño del tablero es considerablemente más grande, la búsqueda de una solución óptima mediante prueba y error resultaría extremadamente laboriosa.

Nuevamente, podemos emplear el algoritmo de búsqueda en profundidad (DFS) modificado para encontrar una solución eficiente. En cada paso del algoritmo, el jugador se mueve a una casilla vecina en el tablero y acumula el valor de la casilla visitada al valor acumulado. Se deben tener en cuenta restricciones como no visitar casillas previamente visitadas y no superar el número objetivo. El algoritmo se ejecuta de manera recursiva, evaluando diferentes movimientos hasta encontrar un camino que sume el número objetivo. Para este tablero difícil, se obtiene la siguiente solución utilizando el algoritmo DFS modificado:

■	■	■	3	3	2	5	4	6	1
9	6	■	■	■	5	3	8	7	7
9	5	2	7	■	5	■	■	■	3
5	7	9	4	■	■	■	9	■	1
4	9	2	3	5	4	1	8	■	2
6	■	■	5	1	3	6	7	■	■
■	■	3	6	■	■	8	9	■	■
■	4	4	9	■	■	■	■	■	1
■	9	3	6	■	■	■	9	4	5
■	■	■	■	■	2	7	4	2	5

4.4 Ejemplo 1: Un tablero muy difícil

■	4	4	7	7	5	2	4	4	7	6	8	4	3	3
3	9	2	1	3	1	3	4	4	1	8	9	4	8	4
7	3	4	8	4	6	2	7	5	8	4	4	4	3	2
5	1	8	5	4	9	1	9	3	2	4	3	1	6	7
6	8	9	8	6	3	8	2	7	6	5	4	5	2	8
9	1	4	8	8	8	9	4	6	8	8	1	3	8	1
4	2	8	3	8	6	8	9	3	8	9	8	8	9	4
6	4	3	3	4	7	7	8	5	8	8	1	7	4	8
7	8	4	5	9	3	2	1	5	5	9	9	9	2	7
4	5	4	8	4	7	7	8	5	7	4	3	4	6	9
5	2	4	5	2	3	7	1	3	7	1	3	2	1	5
5	9	6	6	2	1	7	2	7	4	1	2	8	3	7
1	9	6	3	7	4	2	9	8	9	3	5	7	1	4
7	2	9	4	7	8	3	3	6	8	5	4	2	9	5
6	6	2	9	6	7	3	5	1	6	9	6	9	6	9

Para este tablero, se tiene el objetivo de crear un camino de costo 250. Si bien este tablero solo creció en 5 casillas con respecto al anterior, el número posibles caminos crece con crece con el cuadrado de la distancia del tablero. Llegado este punto, encontrar una solución correcta a mano podría ser cuestión de horas, por lo que puede que este tablero ya sea demasiado grande para ser utilizado en un 'scape room' (El mejor tablero para este propósito quizá siendo el difícil). A pesar de esto, nuestro algoritmo puede fácilmente encontrar una solución a este tablero y a tableros más grandes en cuestión de milisegundos. Para este caso particular, nuestro algoritmo desarrollado encuentra la siguiente solución:

■	■	4	7	7	5	2	4	4	7	6	8	4	3	3
3	■	■	■	■	1	3	4	4	1	8	9	4	8	4
7	3	■	■	■	6	2	7	5	8	4	4	4	3	2
5	1	8	■	■	9	1	9	3	2	4	3	1	6	7
6	■	■	■	6	3	8	2	7	6	5	4	5	2	8
■	■	4	8	8	8	9	4	6	8	8	1	3	8	1
■	■	■	3	8	6	8	9	3	8	9	8	8	9	4
6	■	■	3	4	7	7	8	5	8	8	1	7	4	8
7	■	4	5	9	3	2	1	5	5	9	9	9	2	7
■	■	4	8	4	7	7	8	5	7	■	■	■	■	9
■	2	4	5	2	3	7	1	3	7	■	■	■	1	5
■	9	6	6	2	1	7	2	7	4	■	■	8	3	7
■	9	6	■	■	4	2	9	8	9	3	■	7	1	4
■	■	■	■	■	8	■	■	■	■	■	■	2	9	5
6	6	2	9	■	■	■	5	■	■	9	6	9	6	9