Laboratorio Nro. 2: Fuerza Bruta

Juan Gonzalo Quiroz
Universidad Eafit
Medellín, Colombia
jquiro12@eafit.edu.co

Alejandro Díaz Cano
Universidad Eafit
Medellín, Colombia
adiazc@eafit.edu.co

2) En el código analizamos las diferentes formas de ataque que tiene la reina, Y en una posición marcamos todo horizontalmente, verticalmente y con sus respectivas diagonales para saber en dónde no podríamos poner las demás reinas y empezamos a poner hasta que se ajuste las cantidades de reinas, en caso de que falte una la posición de la primera sube una unidad hasta que se ajusten todas y si no quedan más casillas avanza la segunda una posición y así sucesivamente hasta encontrar las opciones correctas.

2.1) En este punto usamos una variable booleana para saber si es posible poner la reina en caso de no serlo avanza a la siguiente posición hasta encontrar la posición donde se pueda posicionar sin que las otras reinas se ataquen.

3) Simulacro de preguntas de sustentación de Proyectos

1. **Fuerza Bruta, Back tracking, se podría con una solución poner sus otras posibles soluciones como lo dice en el texto con giros cada 90° y en modos espejo cada uno para ahorrar un tiempo de ejecución importante.**
2.

Valor de N	Tiempo de ejecución
4	O(N^2)
8	O(N^3)
16	O(N^4)
32	O(N^5)
N	O(N^M)

3. Antes de posicionar la reina pregunta ¿es una posición válida? (en este caso si no está marcada como no se puede poner, por medio de un arreglo booleano) entonces en las restantes empieza hacer la función normal eliminando las diagonales, verticales y horizontales con el fin de encontrar las posiciones en caso de que alguna no concuerde avanza una unidad y si llega al final se devuelve uno y avanza en uno el anterior y así sucesivamente.

4. El programa antes de poner la reina lee un arreglo booleano donde se encuentran las coordenadas de donde está mal la tabla o donde no se puede poner, y en caso de estar mala avanza una unidad hasta encontrar una posición buena. Y lo pone pasa a la siguiente fila y vuelve a revisar y la pone donde sea válida, y en caso de que ya se tenga todo se retorna la solución, en caso contrario comienza a devolverse y aumentar la fila anterior en la siguiente posición hasta que la encuentre.

5. O(n*m)

6. La cantidad de llamados que hace cada una por que son "for" anidados con el fin de crear un ciclo y recorrer en dos posiciones de un arreglo [n][m]

4) Simulacro de Parcial

1. *a[j+1)!= NULL*
2. *O(n*m)*
3. I,K
4. O(n^2)

5. Lectura recomendada (opcional)

a) Introduction to the design & Analysis of Algorithm's.
b) El plan general es analizar eficientemente el tiempo de ejecución, la eficiencia del algoritmo recursivo.
Indicando el tamaño e identificando las operaciones del algoritmo, revisando el número mínimo de operación, máximo y el promedio.
c) Mapa de Conceptos

6. Trabajo en Equipo y Progreso Gradual (Opcional)

DOCENTE MAURICIO TORO BERMÚDEZ
Teléfono: (+57) (4) 261 95 00 Ext. 9473. Oficina: 19 - 627
Correo: mtorobe@eafit.edu.co

a) Actas de reunión

Nos reunimos todos los jueves y fines de semana trabajos en unión por medio de la Skype, Drive, etc.

b) El reporte de cambios en el código
Se encuentra en GitHub

c) El reporte de cambios del informe de laboratorio
Se trabajó en el informe un jueves y fines de semana en drive.