Entrega Final Proyecto: "Taquin"

Cristian Camilo Contreras Borja[1] Kevin Andres Garzon Ospina [2]
Nicolas David Gil Hernandez[3]

Pontificia Universidad Javeriana, Bogota D.c, Colombia.
{[1]contreras cristian, [2]Ka garzon , [3]nicolas gil }@Javeriana.edu.co

2 de junio de 2022

Resumen

En este docuemnto se presentara el analisis,diseño e implementacion del algortimo para jugar de manera autonoma el juego "Taquin"dado un tamaño del tablero de juego.

Keywords: Arboles,Taquin

1. Parte I Análisis y diseño del problema

1.1. Análisis

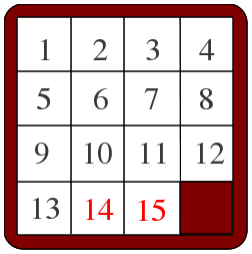

Figura 1: Ejemplificacion Juego Taquin.

El problema informalmente se puede definir como: implementar el juego de TAQUIN el cual consiste en el deslizamiento de piezas que presentan un determinado orden inicial dentro de una caja cuadrada (matriz cuadrada) de las cuales solo una de sus casillas no esta ocupada. Los numeros dentro de la matriz estan desordenados, el juego consiste en maniobrar todas las fichas para corregir el orden de estas, de manera que estas queden en orden consecutivo, se habla de una matriz de numeros:

$$M =< X1, X2, X3.....Xn >< Y1, Y2, Y3.....Yn >=< Xi; Yi \in Z \wedge 0in >$$

donde n es el tamaño que tiene la matriz e i es el indice de cada elemento, en este caso se inicializa en 0 debido al uso de dicho numero en la matriz.

Varias preguntas surgen a traves de este contexto dado:

- ¿Solamente pueden ordenarse numeros enteros?
- ¿Que criterio se toma para ordenar esta matriz?

Se conoce que los numeros enteros no son los unicos elementos ordenables, en varios contextos se pueden ordenar otro tipos de numeros tales como los naturales o los reales, pero a su vez se pueden ordenar elementos ya sea alfabeticamente, imagenes u otro orden especifico, aclarando esto se puede decir que la definicion de matriz para este problema cambiaria:

donde T es un conjunto de elementos que se pueden ordenar.

De igual manera se sabe que la forma mas logica de comparar dos elementos sea a traves de un simbolo como lo es lo cual hace que podamos distinguir cual de dos elementos tienen su orden respectivo.

Algunos datos para tener en cuenta para el diseño de este algoritmo son los siguientes:

- La matriz ques sera trabajada durante el problema tendra como valores datos numericos enteros para facilitar la expliacacion.
- La matriz contara con n dimensiones como se habia aclarado anteriormente.
- Los datos numericos de la matriz seran dados de manera aleatorea por el algortimo

2. Diseño

Con base en las observaciones presentadas en el análisis anterior, podemos escribir un diseño de algoritmo para resolver el problema de jugar TAQUIN. Se evidencia ciertas variables que cambian en la entrada del algoritmo.

2.1. Entradas

1. Un numero n $\in Z, estedatohacereferenciaaltamañoqueseledaraalamatrizenlaquesetrabajara.$

2.2. Salidas

1. Una numero n $\in Z, elcualrepresentalacantidaddepasosparallegaralasolucionSecuencias$S= ¡Xi $\in T \wedge 0in >$ donde las secuencias que se muestran son la solucion paso a paso del tablero

3. Parte II Algoritmo

3.1. Algoritmo

Algorithm 1 *main*

```
1: procedure Daux(A,I,V)
2:     if V = 0 then
3:         return 0
4:     else
5:
6:         if I = 0 then
7:             return I
8:         end if
9:
10:
11:        if A[I] > V then
12:            return Daux(A, I − 1, V)
13:        end if
14:
15:
16:        if A[I] <= V then
17:            return Daux(A, I, V − A[I]) + 1
18:        end if
19:    end if
20: end procedure=0
```

Algorithm 2 *solution*

```
    procedure SOL(S)
2:      iniciarTablero()

4:      while cola do
            nodo ← cola.pop()
6:          if nodo.resuelto then
                z ← list(nodo.camino)
8:              final(z)
            end if
10:         for movimiento, dir ← nodo.acciones do
              hijos ← nodo(movimiento(), nodo, dir)
12:         if hijos.estado not in arbol then
                cola.appendleft(hijos)
14:             arbol.add(hijos.estado)
            end if
16:         end for

18:
```

3.2. Complejidad

El algoritmo *Algoritmo para jugar "Taquin.* tiene orden de complejidad

$$O(log2(n))$$

Este algoritmo que se diseño para calcular los nodos de cada instancia de tablero e ir calculando el menor de cada uno de ellos por lo tanto recorre la matriz completa, donde N hace referencia al tamaño de la matriz esto dado en su peor caso.

3.3. Invariante

La invariante del algoritmo propuesto consiste en la comparacion del nodo que se esta revisando con el nodo de estado final para valdiar si encontro la solucion.