Jose Toro Bustos – 20.239.232-5 - Sección A1

Representación de una posible solución.

Una posible solución se representa gráficamente mediante una matriz con 1's y 0's donde los 1's representan cuadrados negros y los 0's representan cuadrados blancos

Como se ve en la imagen

```
las tripletas ingresadas tienen multiples soluciones
Solucion N=0
0 0 1
0 1 0
1 0 0
Solucion N=1
0 0 1
1 0 0
0 1 0
```

Estas dos de las distintas soluciones para las tripletas (1,1,1) y (1,1,1)

Listado de restricciones o conjunto de restricciones que poseen, explicando para cada una:

el código posee una función llamada verificarSolucion, la cual se encarga de revisar si una matriz generada es una solución de las tripletas entregadas por el usuario

```
int verificarSolucion(int a,int b,int c,int d,int e,int f, int ** matriz){
    int fila1,fila2,fila3,columna1,columna2,columna3;
    fila1=matriz[0][0] + matriz[0][1] + matriz[0][2];
    fila2=matriz[1][0] + matriz[1][1] + matriz[1][2];
    fila3=matriz[2][0] + matriz[2][1] + matriz[2][2];
    columna1=matriz[0][0] + matriz[1][0] + matriz[2][0];
    columna2=matriz[0][1] + matriz[1][1] + matriz[2][1];
    columna3=matriz[0][2] + matriz[1][2] + matriz[2][2];
    //ahora compruebo que todo se cumpla ya que un 1 significa negro, por ejemplo si mi "a" es 3 y fila1
    if((a == fila1)&&(b == fila2)&&(c == fila3)&&(d == columna1)&&(e == columna2)&&(f == columna3)){
        return 1;
    }
    else{
        return 0; //No cumple la condicion de las tripletas
    }
}
```

Ya que mi código genera todas las variables de matrices posibles (aproximadamente 500) mi función de verificar solución me entrega un valor 1 o 0 dependiendo si la matriz que estoy generando cumple o no con las tripletas asignadas

Para ello sumo la cantidad de 1's de cada fila y de cada columna y la comparo con su valor correspondiente dentro de cada tripleta, si estas comparaciones resultan ser iguales, entonces esa matriz si es una solución

Criterio de optimización, indicando por qué es un criterio de optimización y justificando a este, o, en caso de no existir, justificar su inexistencia.

Para optimizar la ejecución de mi código utilice la función verificarSolucion la cual en caso de que cumpla todo lo anteriormente mencionado me regresa un valor 1.
Con esto mi programa agrega esa matriz a una lista de soluciones
En caso de que no me regrese el valor 1, hago free de la matriz generada y no hay necesidad de agregarla en soluciones, de esta forma si bien genero todas las posibilidades, solo almaceno las matrices que cumplan las tripletas

```
//verifico que esta matriz cumpla con las tripletas
if(verificarSolucion(a,b,c,d,e,f,matriz) == 1){
    //si la matriz creada cumple, la agrego en soluciones
    conjuntoSoluciones = agregarSolucion(conjuntoSoluciones, &cantidadSoluciones,
}
else{//si la matriz no cumple con los requisitos de verificar solucion entonces l
    free(matriz[0]);
    free(matriz[1]);
    free(matriz[2]);
}
```

El algoritmo de Búsqueda en espacio de soluciones aplicado.
(no supe a que se refiere)

Código que obtiene la solución.

Para generar la solución utilice 9 ciclos for anidados como se ve en la imagen

```
for(casilla1=0;casilla1<=1;casilla1++){
    for(casilla2=0;casilla2<=1;casilla2++){
        for(casilla3=0;casilla3<=1;casilla3++){
            for(casilla4=0;casilla4<=1;casilla4++){
                for(casilla5=0;casilla5<=1;casilla5++){
                    for(casilla6=0;casilla6<=1;casilla6++){
                        for(casilla7=0;casilla7<=1;casilla7++){
                            for(casilla8=0;casilla8<=1;casilla8++){
                                for(casilla9=0;casilla9<=1;casilla9++){
                                    //asigno la memoria a matriz
                                    matriz = crearMatriz();
                                    //lleno las posiciones de matriz
                                    matriz[0][0] = casilla1;
                                    matriz[0][1] = casilla2;
                                    matriz[0][2] = casilla3;
                                    matriz[1][0] = casilla4;
                                    matriz[1][1] = casilla5;
                                    matriz[1][2] = casilla6;
                                    matriz[2][0] = casilla7;
                                    matriz[2][1] = casilla8;
                                    matriz[2][2] = casilla9;
```

Los cuales van variando el valor de cada variable casilla (1,2,3,4,5,6,7,8 y 9)

Haciéndolas variar entre los valores 0 y 1 de esta forma represento las matrices pintadas donde 0 es el color blanco y 1 el color negro, con estos ciclos for genero todas las posibles soluciones y luego gracias a la función verificarSolucion almaceno solo las que cumplen con las tripletas.

Luego dependiendo de cuantas soluciones tenga almacenadas, le entrego al usuario tres posibles mensajes

-“no hay solución para las tripletas” en caso de no hayan soluciones

-“las tripletas ingresadas tienen solución única” en caso de que solo haya una solución

-“las tripletas ingresadas tienen múltiples soluciones” en caso de tener mas de una solución

Además, le muestro al usuario la/s solucion/es de manera grafica como se ve en el punto “***Representación de una posible solución.*** “