

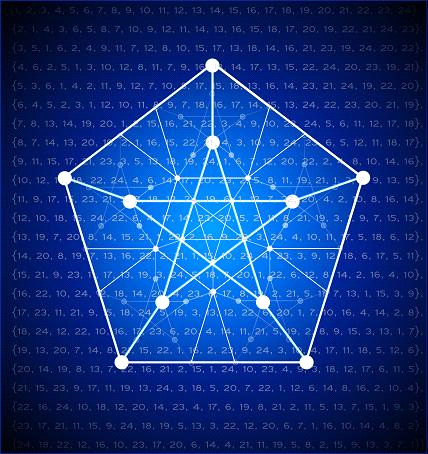
**COLORACIÓN DE GRAFOS**

**Integrantes:**

Moreno Gutiérrez David

Pérez Mendoza Luis Alfredo

Sarahi Soto Palafox



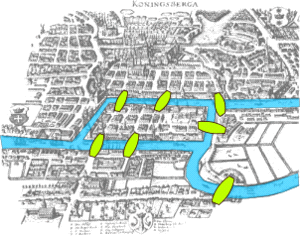
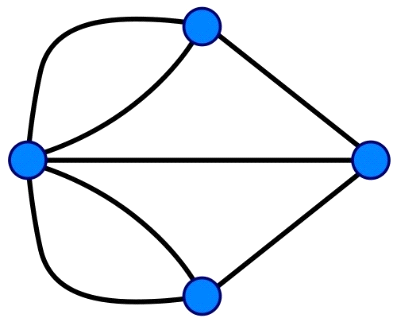
**RESUMEN**

Uno de los problemas más conocidos de Teoría de Grafos es el problema de coloración de grafos. Tal y como indica su nombre, se trata de asignar etiquetas denominadas colores a los elementos de un grafo. Hay varios tipos de problemas de coloración de grafos, siendo el más conocido el problema de coloración de los vértices de un grafo. El objetivo en estos problemas es asignar colores a los vértices de un grafo, de tal forma que los vértices adyacentes no compartan el mismo color. Suele ser habitual además buscar el número cromático, que sería el mínimo número de colores necesario para la coloración del grafo. Los problemas de coloración de grafos tienen múltiples aplicaciones. De hecho, se pueden utilizar para resolver muchos problemas de asignación.

**INTRODUCCIÓN**

La Teoría de Grafos surge como concepto por primera vez en el siglo XVIII, cuando el matemático suizo Leonhard Euler (1707-1783), planteando el conocido Problema de los siete puentes de Königsberg, (en la actualidad Kaliningrado, Rusia), problema que resuelve en el año 1736. Dicho problema consistía en encontrar un camino tal que, pasando una única vez por cada uno de los puentes, se pudiera regresar al punto de partida. La respuesta a dicho problema es que no existe ningún camino cumpliendo esas características.

En su demostración, representó cada puente como una línea (arista) uniendo dos puntos (vértices), cada uno de los cuales se correspondía con una región diferente. Sentenció la demostración indicando que los puntos intermedios de un posible recorrido tienen que estar necesariamente conectados a un número par de aristas, de forma que los puntos inicial y final serían los únicos que podrían estar conectados por un número impar de líneas, lo cual no puede suceder pues este debe ser el mismo.

.  

Posteriormente, en el año 1847, Gustav Kirchhoff utiliza la Teoría de Grafos para analizar redes eléctricas, publicando las Leyes de Kirchhoff, permitiendo calcular así el voltaje y la corriente en circuitos eléctricos.

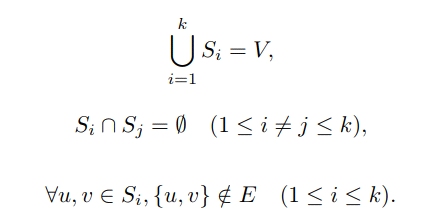
Pasaron años de investigación en torno al problema, centrados en demostraciones sobre los vértices de grafos planos sin lazos, es decir, los grafos duales asociados a un mapa.

Finalmente, en el año 1976, los matemáticos Kenneth Appel y Wolfgang Haken demostraron que no puede existir ninguna configuración en un contraejemplo mínimo del Teorema de los Cuatro Colores. Basaron su demostración en la deducibilidad, utilizando las Cadenas de Kempe. Redujeron el número de configuraciones del conjunto inevitable a 1.936, los cuales fueron comprobados individualmente con ayuda del ordenador. Utilizaron 1200 horas de tiempo computacional para obtener la prueba final. Podemos ver un ejemplo de cómo se puede aplicar el Teorema de los Cuatro Colores sobre las provincias de España.

El Teorema de los Cuatro Colores fue el primer gran teorema demostrado utilizando un ordenador, presentando una demostración que no pudo ser verificada directamente por otros matemáticos. Con el paso del tiempo, la demostración ha sido aceptada por la mayoría, proporcionando incluso refinamientos en cuanto al número de configuraciones (hasta 633), aunque siempre queda quien se hace preguntas del tipo ¿Cómo podemos garantizar la fiabilidad de los algoritmos y del hardware?

**Dificultad del problema**

Resulta también útil para comprender el problema de coloración de grafos, plantearlo como un tipo de problema de particiones sobre el conjunto de vértices V, donde una solución S es representada por un conjunto de k clases de colores (o colores), S = {S1, . . . , Sk}. Para que la solución sea factible, es necesario que se cumplan las siguientes restricciones a la vez que se minimiza el número de colores k:



Las restricciones (1.1) y (1.2) establecen que S debe ser una partición del conjunto de vértices V, mientras que la restricción (1.3) obliga a que ningún par de vértices adyacentes sean asignados a la misma clase (o color), es decir, que todas las clases de colores en la solución deben ser conjuntos independientes.

**Dificultad del problema:** NP Completo.

**CONCEPTUALIZACIÓN**

**Definición.** Un grafo G es un par ordenado G= (V,E), donde V es un conjunto de vértices y E es un conjunto de aristas.

**Definición. La arista** a la **línea resultante del cruce de dos superficies o planos**. Las aristas también son los segmentos de una [**recta**](https://definicion.de/recta/) que marcan el límite de los lados de una figura plana.

**Definición**. El vértice es el nombre que recibe el punto que marca la unión entre los segmentos que originan un ángulo o donde se fusiona un mínimo de tres planos.

En la **teoría de grafos**, cada vértice está considerado como la unidad fundamental que compone a los [**grafos**](https://definicion.de/grafos/). Los grafos no dirigidos están compuestos por vértices y aristas (es decir, pares desordenados de vértices), mientras que los grafos dirigidos abarcan vértices y arcos (pares ordenados de vértices).

La **coloración de grafos** es un caso especial de etiquetado de grafos es una asignación de etiquetas llamadas *colores* a elementos del grafo.

**Definición.** Una coloración se dice factible si y sólo si es completa y apropiada, es decir, que todos los vértices están asignados a algún color y que ningún par de vértices adyacentes (dados u,v∈V{u,v}∈E) esa asignado al mismo color.

En palabras coloquiales es una coloración de los vértices de un grafo tal que ningún vértice adyacente comparta el mismo color es llamado vértice coloración.

**Definición.** Dado un grafo G=(V,E), una matriz de adyacencia es una matriz A∈Rn×n para la cual Aij= 1 si y sólo si los vértices vi y vj son adyacentes (es decir, existe una arista que los une), y Aij= 0 en cualquier otro caso.

**Definición**. Una Triangulación es una configuración es parte de una triangulación (disposición de un grafo plano en el que las caras están delimitadas por tres aristas, formando así triángulos) contenida en un circuito, es decir, planteando un grafo plano como un circuito eléctrico, siendo este un ciclo (camino cerrado contenido en el grafo). Un conjunto inevitable es un conjunto de configuraciones con la propiedad de que cualquier triangulación debe contener una de las configuraciones del conjunto.

**Tipos de grafos**

**K-Regular**

Un grafo regular es un [grafo](https://es.wikipedia.org/wiki/Grafo) donde cada vértice tiene el mismo grado o valencia. Un grafo regular con vértices de grado *k* es llamado grafo *k*-regular o grafo regular de grado k.

Por ejemplo, un grafo 2-regular consiste en un ciclo.

Los grafos 0-regualres son grafos donde no hay vértices adyacentes, es decir, se trata de grafos G=(V,E), con E=∅.

Consiste en un grafo con vértices desconectados.

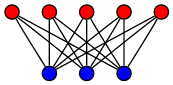
Un grafo es 0-regular si y sólo si su número cromático es 1, es decir, se asignan todos los vértices al mismo color.

**Completos**

Los grafos completos con n vértices, denotados por Kn, son grafos que presentan una arista entre cada par de vértices del grafo, lo que proporciona un conjunto E de m = n (n − 1) 2 aristas. Puesto que todos los vértices del grafo son adyacentes entre sí, cada vértice debe ser asignado a un color distinto, por lo que el número cromático de un grafo completo es X( Kn ) = n

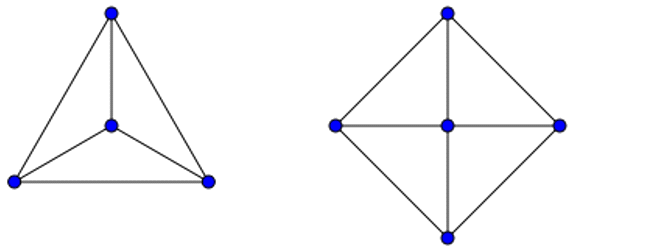
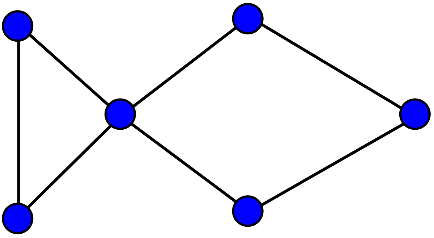
**Bipartitos**

Los grafos bipartitos, denotados por G = (V1 V2, E), son grafos cuyos vértices pueden ser particionados en dos conjuntos V1 y V2 tales que sólo existen aristas entre vértices de V1 y vértices de V2. En consecuencia, V1 y V2 son conjuntos independientes, por lo que los grafos bipartitos pueden ser coloreados utilizando sólo 2 colores, asignando todos los vértices de V1 a un color y todos los vértices de V2 al otro. Resulta evidente que X(G) = 2 si y sólo si G es bipartito.



**Grafos Planos**

Los grafos planos son aquellos que pueden ser representados en el plano, sin que ninguna de las aristas del grafo corte a otra. Puede ser representado de manera que sus aristas se intersectan únicamente los vértices del grafo. Estos se ajustan a lo que hemos visto en cuanto al Teorema de los cuatro colores (si un grafo es plano entonces puede ser coloreado de forma factible utilizando 4 o menos colores), aunque el recíproco no es necesariamente cierto.



**APLICACIONES**

* Para colorear cualquier mapa de países de tal forma que dos países vecinos nunca tengan el mismo color.

Este problema fue resuelto utilizado solo cuatro colores por Kenneth Appel y Wolfgang Haken en 1976, puede ser considerado como el nacimiento de la teoría de grafos.

* En la enumeración de los isómeros, compuestos químicos con idéntica composición (fórmula) pero diferente estructura molecular.

Problema resuelto por Arthur Cayley en 1857, donde represento cada compuesto mediante un grafo árbol en el cual los vértices representan átomos y las aristas la existencia de enlaces químicos.

* Se utiliza para diferentes áreas: Dibujo computacional, en todas las áreas de Ingeniería.
* Los grafos se utilizan para modelar trayectos. Como el de una línea de autobús a través de las calles de una ciudad, así obtener caminos óptimos para el trayecto.
* Para la administración de proyectos: como técnica de revisión y evaluación de programas (PERT) en las que se modelan los mismos utilizando grafos y optimizando los tiempos para concretar los mismos.
* En las ciencias sociales para desarrollar un concepto no metafórico de red social que sustituye los nodos por los actores sociales y verifica la posición, centralidad e importancia de cada actor dentro de la red. Por ejemplo, una red social puede representar la estructura de poder dentro de una sociedad al identificar los vínculos (aristas), su dirección e intensidad y da idea de la manera en que el poder se transmite y a quiénes.
* Se emplea en problemas de control de producción, para proyectar redes de ordenadores, para diseñar módulos electrónicos modernos y proyectar sistemas físicos con parámetros localizados (mecánicos, acústicos y eléctricos).
* Se usa para la solución de problemas de genética y problemas de automatización de la proyección (SAPR). Apoyo matemático de los sistemas modernos para el procesamiento de la información. Acude en las investigaciones nucleares (técnica de diagramas de Feynman).
* En el estudio de la biología y hábitat. El vértice representa un hábitat y las aristas (o "edges" en inglés) representa los senderos de los animales o las migraciones. Con esta información, los científicos pueden entender cómo esto puede cambiar o afectar a las especies en su hábitat.
* Plano de estaciones del metro
* Plano de autopistas circuito eléctrico.
* Sociogramas de una red social.
* Topología de red de computadores.
* Organigramas.
* Arquitectura de redes de telefonía móvil.
* Draws de eliminación directa.

Ejemplo de aplicación sencillo:

**Programación de exámenes finales**

Sirve para programar los exámenes sin que ningún estudiante tenga dos exámenes al mismo tiempo. Idea: los vértices son las asignaturas y existe una arista entre un par de vértices, si hay un estudiante matriculado en ellas.

**MÉTODO DE SOLUCIÓN CON ALGORITMOS GENÉTICOS**

**A) REPRESENTACIÓN DE LAS SOLUCIONES.**

Cada individuo será representado como un vector (o arreglo) de tamaño del número de regiones que contiene el grafo, para este caso 9, donde cada cromosoma corresponderá a un numero de color que puede adquirir dependiendo del número que escriba en pantalla el usuario.

**GRAFICO:**

|  |  |  |
| --- | --- | --- |
| 1 | 1 | 0 |
| 1 | 0 | 1 |
| 1 | 0 | 0 |

**Individuo:**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 |

Primera Columna Segunda Columna Tercera Columna

Representación de la solución final:

La solución final desplegada en pantalla será una matriz de “n x m” donde ningún color pueda ser adyacente a otro.

|  |  |  |
| --- | --- | --- |
| 1 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 1 |

**B) FUNCIÓN DE APTITUD**

Para la función de aptitud de cada individuo, se hace una comparación de cada cromosoma con su cromosoma vecino (en el grafo) para comparar si el valor que tiene cada uno (su color) es igual o diferente.

Para el siguiente caso la función de aptitud del individuo es la siguiente:

(NOTA: Para el ejemplo se hará representación gráfica del individuo (en forma de matriz), pero las comparaciones (dentro del programa) se hacen en base al grafo de adyacencias, ya que la verdadera representación de individuos es lineal (en forma de arreglo) y no en forma de matriz.

Representación del individuo dentro del programa:

Cromosoma Vecino Vecino

Ver las imágenes de origen Ver las imágenes de origen Ver las imágenes de origen

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 |

Representación grafica del individuo:

|  |  |  |
| --- | --- | --- |
| 1 | 1 | 0 |
| 1 | 0 | 1 |
| 1 | 0 | 0 |

[F(0)=2],[F(1)=1],[F(2)=0],[F(3)=2],[F(4)=1],[F(5)=0],[F(6)=1],[F(7)=2],[F(8)=1]

2+1+0+2+1+0+1+2+1= 10 (Fitness/Aptitud del individuo)

Como se ve en el ejemplo anterior para la función F(n), donde n es la casilla en donde nos encontramos situados, está dada por el número de vecinos semejantes que tenemos, y la función de aptitud está dada por la suma de dichas funciones.

|  |  |  |
| --- | --- | --- |
| 1 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 1 |

[F(0)=0],[F(1)=0],[F(2)=0],[F(3)=0],[F(4)=0],[F(5)=0],[F(6)=0],[F(7)=0],[F(8)=0]

0+0+0+0+0+0+0+0+0= 0 (Fitness/aptitud de la solución).

**C) POBLACIÓN INICIAL (CON TAMAÑOS DIFERENTES: 20, 50, ETC**.).

La población inicial se generó usando números aleatorios irrepetibles para sus cromosomas (donde cada número representa un color distinto).

Para este ejemplo:

La población inicial es de 20 individuos.

Se utilizará un grafo de 3x3.

Solo se utilizan dos colores (0,1).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Ind[0] | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| Ind[1] | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 |
| Ind[2] | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 |
| Ind[3] | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 |
| Ind[4] | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 |
| Ind[5] | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 |
| Ind[6] | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| Ind[7] | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| Ind[8] | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| Ind[9] | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| Ind[10] | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 0 |
| Ind[11] | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 |
| Ind[12] | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 |
| Ind[13] | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 |
| Ind[14] | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 |
| Ind[15] | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 |
| Ind[16] | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Ind[17] | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| Ind[18] | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 |
| Ind[19] | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 |

**D) SELECCIÓN**

**Se utiliza el método de la ruleta**

Calculamos la probabilidad con distribución uniforme para cada individuo.

Pi = F[ind(i)] / FT, donde FT= Σ i=0 F[ind(i)] con i=0, …, 19.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Probabilidad |  | Probabilidad acumulada |
| P0 | 14/256 = 0.0546875 | q0 | 0.0546875 |
| P1 | 8 /256 = 0.0312500 | q1 | 0.0859375 |
| P2 | 14/256 = 0.0546875 | q2 | 0.1406250 |
| P3 | 14/256 = 0.0546875 | q3 | 0.1953125 |
| P4 | 8 /256 = 0.0312500 | q4 | 0.2265625 |
| P5 | 16/256 = 0.0625000 | q5 | 0.2890625 |
| P6 | 12/256 = 0.0468750 | q6 | 0.3359375 |
| P7 | 20/256 = 0.0781250 | q7 | 0.4140625 |
| P8 | 18/256 = 0.0703125 | q8 | 0.4843750 |
| P9 | 10/256 = 0.0390625 | q9 | 0.5234375 |
| P10 | 10/256 = 0.0390625 | q10 | 0.5625000 |
| P11 | 14/256 = 0.0546875 | q11 | 0.6171875 |
| P12 | 12/256 = 0.0468750 | q12 | 0.6640625 |
| P13 | 8 /256 = 0.0312500 | q13 | 0.6953125 |
| P14 | 8 /256 = 0.0312500 | q14 | 0.7265625 |
| P15 | 10/256 = 0.0390625 | q15 | 0.7656250 |
| P16 | 20/256 =0.0781250 | q16 | 0.8437500 |
| P17 | 14/256 = 0.0546875 | q17 | 0.8984375 |
| P18 | 14/256 = 0.0546875 | q18 | 0.9531250 |
| P19 | 12/256 = 0.0468750 | q19 | 1.0000000 |

**Se gira la ruleta 20 veces**, es decir, se genera aleatoriamente 20 números en [0,1]

Y se selecciona siguiente población que cumpla: qi < rk < qj con i,j,k=0,…,19.

|  |  |  |
| --- | --- | --- |
|  | Números aleatorios | Seleccionamos |
| r0 | 0.951346 | Ind[18] |
| r1 | 0.561743 | Ind[10] |
| r2 | 0.718657 | Ind[14] |
| r3 | 0.972346 | Ind[19] |
| r4 | 0.122234 | Ind[2] |
| r5 | 0.353523 | Ind[7] |
| r6 | 0.708534 | Ind[14] |
| r7 | 0.522356 | Ind[9] |
| r8 | 0.119536 | Ind[2] |
| r9 | 0.302164 | Ind[6] |
| r10 | 0.682364 | Ind[13] |
| r11 | 0.885046 | Ind[17] |
| r12 | 0.413423 | Ind[7] |
| r13 | 0.913253 | Ind[18] |
| r14 | 0.061285 | Ind[1] |
| r15 | 0.214553 | Ind[4] |
| r16 | 0.699323 | Ind[14] |
| r17 | 0.561234 | Ind[10] |
| r18 | 0.434361 | Ind[8] |
| r19 | 0.863398 | Ind[17] |

**E) CRUZAMIENTO**

El método de cruzamiento solo se puede efectuar en parejas, por lo que el número total de la población debe ser par y funciona de la siguiente manera.

Se selecciona un punto de cruzamiento aleatorio entre 1 y (número de regiones-1) esto es para que al menos una parte de los individuos se pueda intercambiar.

Suponemos que para este caso el punto de cruza es 2.

(Punto de cruzamiento)

Ver las imágenes de origen

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |

(Estos son los individuos originales.)

Los datos que se encuentran del punto de cruzamiento en adelante (hacia la derecha) son intercambiando entre individuos para formar dos individuos nuevos.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 |

Después se hace la evaluación de las aptitudes (fitness) de los nuevos individuos y se comparan con la de su generación pasada, si la aptitud de los “hijos” supera (en este caso minimizando) o iguala la aptitud de los padres ahora estos ocuparan su lugar en la población, para poder así variar información genética en las nuevas generaciones, si por el lado contrario su aptitud es menor que la de los padres, los hijos no serán parte de la población.

Para la siguiente generación es posible que pasen (Los dos padres, los dos hijos o uno y uno dependiendo sea el caso).

**F) MUTACIÓN**

Para el caso de la mutación, este es un método más sencillo y que además solo afecta a un individuo cada vez que este sucede.

Para este caso las probabilidades de mutación por generación son del 10% o menos (por cada 100 individuos mutaran entre 0 y máximo 10).

Como se observa los porcentajes de mutación son bastante menores que lo de cruzamiento tratando de asemejarse un poco a la naturaleza donde de igual manera las mutaciones son menos frecuentes.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 |

-Para la mutación se elige un individuo al azar.

-Luego se genera un numero aleatorio entre y (número de regiones-1), este número nos sirve para determinar la casilla del cromosoma que de mutara

-En el caso de que solo haya dos colores (0 y 1) si en la casilla se encuentra un 0 cambiara a 1 y viceversa

-En el caso de que el ejemplo utilice más colores, se generará otro número aleatorio con los diferentes colores, y la casilla seleccionada será cambiada por el nuevo color generado aleatoriamente.

Para este caso suponiendo que el número aleatorio para la selección de casilla fue 4 y se empieza a contar desde 0 el nuevo individuo es el siguiente.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 |

Al igual que en el método de cruzamiento después de la mutación de evaluará la aptitud del individuo, si el nuevo individuo tiene una mejor aptitud o igual aptitud, pasara a formar parte de la población, en caso contrario no.

**G) CRITERIO DE PARO**

Se tomó como criterio de paro, cuando el programa encuentre a un individuo con una aptitud (fitness) igual a cero que para este caso es la solución del problema, en caso contrario se optó por detener el programa cuando el número de generaciones sea igual a 2000 (Dos Mil generaciones).

**IMPLEMENTACIÓN DEL ALGORITMO**

**Lenguaje de programación:**

El lenguaje utilizado para la implementación de este algoritmo será “c”.

**Entradas y salidas del programa:**

El algoritmo consta de una interfaz que requiere los siguientes elementos:

1. Tamaño de la población.
2. Número de regiones.
3. Número de colores.
4. Nombre del archivo extensión txt donde se encuentra escrito el grafo.
5. Probabilidad de mutación.
6. Probabilidad de cruzamiento.

El algoritmo despliega como resultado lo siguiente:

1. El proceso de lectura del archivo.
2. La población inicial con la que se trabajara.
3. El proceso de cálculo de fitness de cada generación.
4. El número de generaciones que necesitó y el individuo con la solución óptima.
5. La solución óptima mostrada como una matriz
6. El tiempo de ejecución a partir de que empieza el algoritmo genético.

**FUNCIONAMIENTO:**

El algoritmo realiza en orden las siguientes funciones:

* **Obtención y validación de parámetros de entrada**

En este apartado se valida que los datos ingresados por el usuario no causen conflictos en el algoritmo y se crean las variables para operar con ellas.

* **Lectura del archivo que contiene el grafo a trabajar y desplegado**

En el archivo el primer dato siempre se tomará como el número de regiones, después cada renglón se diferencia con un –1 para seguir con la siguiente línea, después el primer elemento de cada renglón se lee, y los siguientes representan una arista adyacente.

* **Generación de la población inicial y desplegado en pantalla**

Gracias a la función malloc se generan número aleatorios que no se repiten para llenar una matriz del tamaño del individuo por el tamaño de la población e imprimirlo en pantalla.

* **Calculo de fitness**

Se usa la función de aptitud para calcular el valor de cada individuo en una nueva matriz y se conserva el mejor guardándolo en una variable por si no se vuelve a encontrar en la siguiente generación.

* **Selección y cruzamiento**

Se lee la cantidad introducida por el usuario que necesaria mente debe ser menor al 10 por ciento, se seleccionan los elementos que serán usados para el cruzamiento y se realiza el proceso de cruzamiento.

* **Mutación**

Se lee la cantidad introducida por el usuario que necesaria mente debe ser menor al 10 por ciento y se seleccionan los elementos que serán usados para la mutación.

* **Recalculo cíclicamente.**

Se repite cíclicamente el proceso de selección, cruzamiento y mutación mientras no se tenga un fitness igual a cero o no se hayan llegado a 2000 generaciones.

Una vez encontrado el resultado se muestra un pequeño análisis de éste.

**Código principal (Main):**

int main ()

{ clock\_t t\_in,t\_fin;

double secs;

srand(time(NULL));

int \*\*pop,\*\*aux\_pop,n\_reg,n\_ind,n\_colour,best,M,N;

int Nreg,bestInd;

float probc, probm;

FILE \*input;

chain filename;

do

{

printf("INGRESE EL TAMAÑO DE LA POBLACIÓN\n");

scanf("%d",&n\_ind);

}while(n\_ind%2!=0);

printf("INGRESE EL TAMAÑO DEL COLOR DE REGIONES MxN\n");

scanf("%d %d",&M,&N);

printf("INGRESE EL NUMERO DE COLORES\n");

scanf("%d",&n\_colour);

printf("NOMBRE DEL ARCHIVO DEL GRAFO\n");

scanf("%s",filename);

printf("INSERTE PROBABILIDAD DE CRUZA (ENTRE 0.2 Y 0.8)\n");

scanf("%f",&probc);

printf("INSERTE PROBABILIDAD DE MUTACION(ENTRE 0.0 Y 0.1)\n");

scanf("%f",&probm);

printf("\n");

n\_reg = M\*N;

struct Node \*admat[n\_reg]; //Una estructura para guardar el arreglo de la lista ligada para la matriz de adyacencia

int fitness[n\_ind],mejor\_ind[n\_reg],aux\_fitness[n\_ind]; // Un arreglo para almacenar el fitness de cada individuo

t\_in = clock();

get\_ad\_matrix(admat,n\_reg,filename); //Obtiene la matriz de adyacencia

disp\_ad\_matrix(admat,n\_reg);// Muestra la matriz de adyacencia

pop = create\_matrix(n\_ind,n\_reg); //Guarda la memoria para la matriz poblacion

aux\_pop = create\_matrix(n\_ind,n\_reg);

generate\_rand\_pop(pop,n\_ind,n\_reg,n\_colour); //Crea aleatoriamente la poblacion inicial

display\_pop(pop,n\_ind,n\_reg); // Despliega la matriz con la poblacion

get\_fitness(pop,admat,fitness,n\_ind,n\_reg);// calcula el fitness

best = get\_best(pop,fitness,mejor\_ind,n\_ind,n\_reg,&bestInd);

int N\_gen=0;

while(best!=0)

{

cruzamiento(pop,aux\_pop,n\_ind,n\_reg);

get\_fitness(aux\_pop,admat,aux\_fitness,n\_ind,n\_reg);

compara\_cruz(aux\_fitness,fitness,pop,aux\_pop,n\_ind,n\_reg);

best = get\_best(pop,fitness,mejor\_ind,n\_ind,n\_reg,&bestInd);

if(best==0)

{

break;

}

clonacion(pop,aux\_pop,n\_ind,n\_reg);

mutation(aux\_pop,n\_ind,n\_reg,n\_colour);

get\_fitness(aux\_pop,admat,aux\_fitness,n\_ind,n\_reg);

compara\_mut(aux\_fitness,fitness,pop,aux\_pop,n\_ind,n\_reg);

best = get\_best(pop,fitness,mejor\_ind,n\_ind,n\_reg,&bestInd);

N\_gen++;

}

printf("SOLUCION ENCONTRADA DESPUÉS DE %d GENERACIONES Y FUE EL INDIVIDUO %d\n",N\_gen,bestInd);

int \*\*sol;

sol = create\_matrix(M,N);

int k=0;

for(int i=0; i<M; i++)

{

for(int j=0; j<N;j++)

{

sol[i][j] = mejor\_ind[k];

k++;

printf("%d \t",sol[i][j]);

}

printf("\n");

}

t\_fin = clock();

secs = (double)(t\_fin - t\_in)/CLOCKS\_PER\_SEC;

printf("EL TIEMPO TOTAL DE EJECUCIÓN ES %lf segundos \n", secs);

return 0;

}

**EJECUCIÓN DE VARIAS INSTANCIAS DEL PROBLEMA**

**Primera ejecución.**

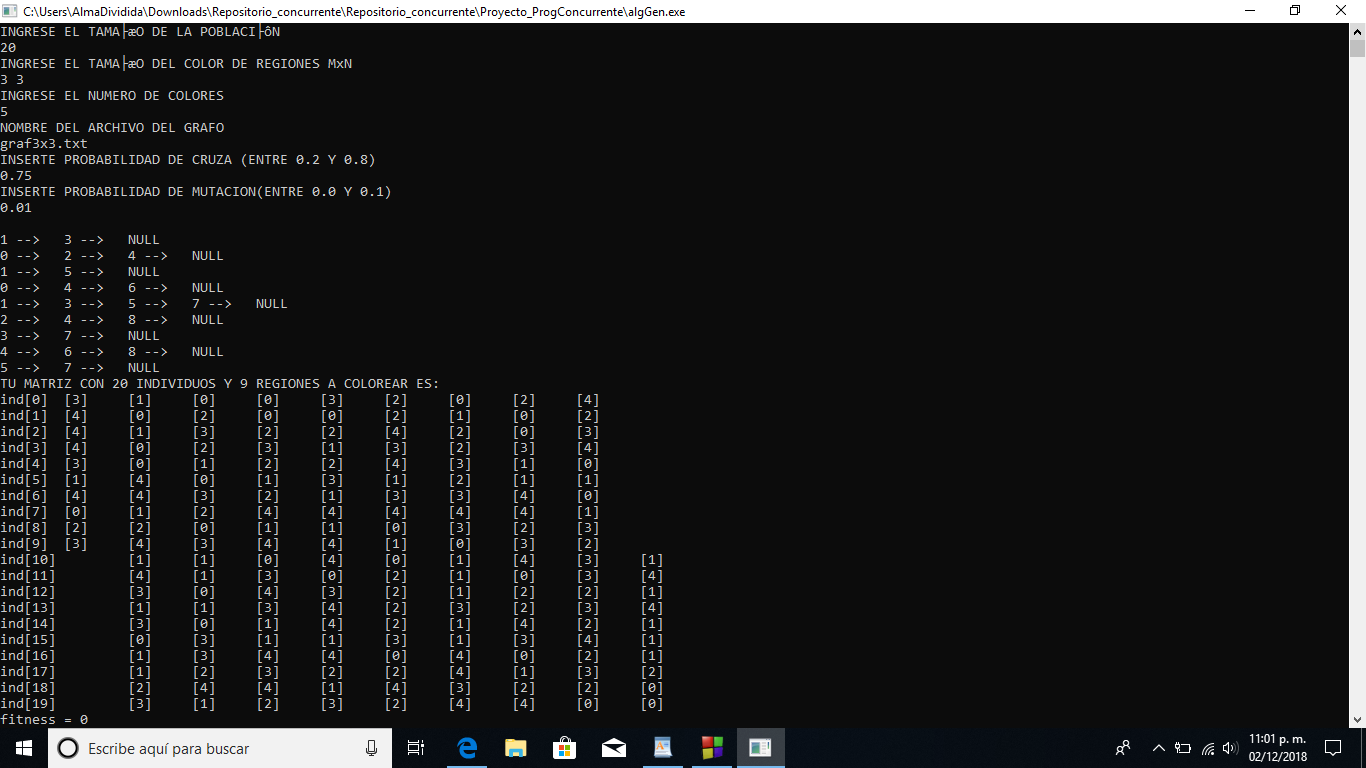
Tamaño de la población inicial: 20

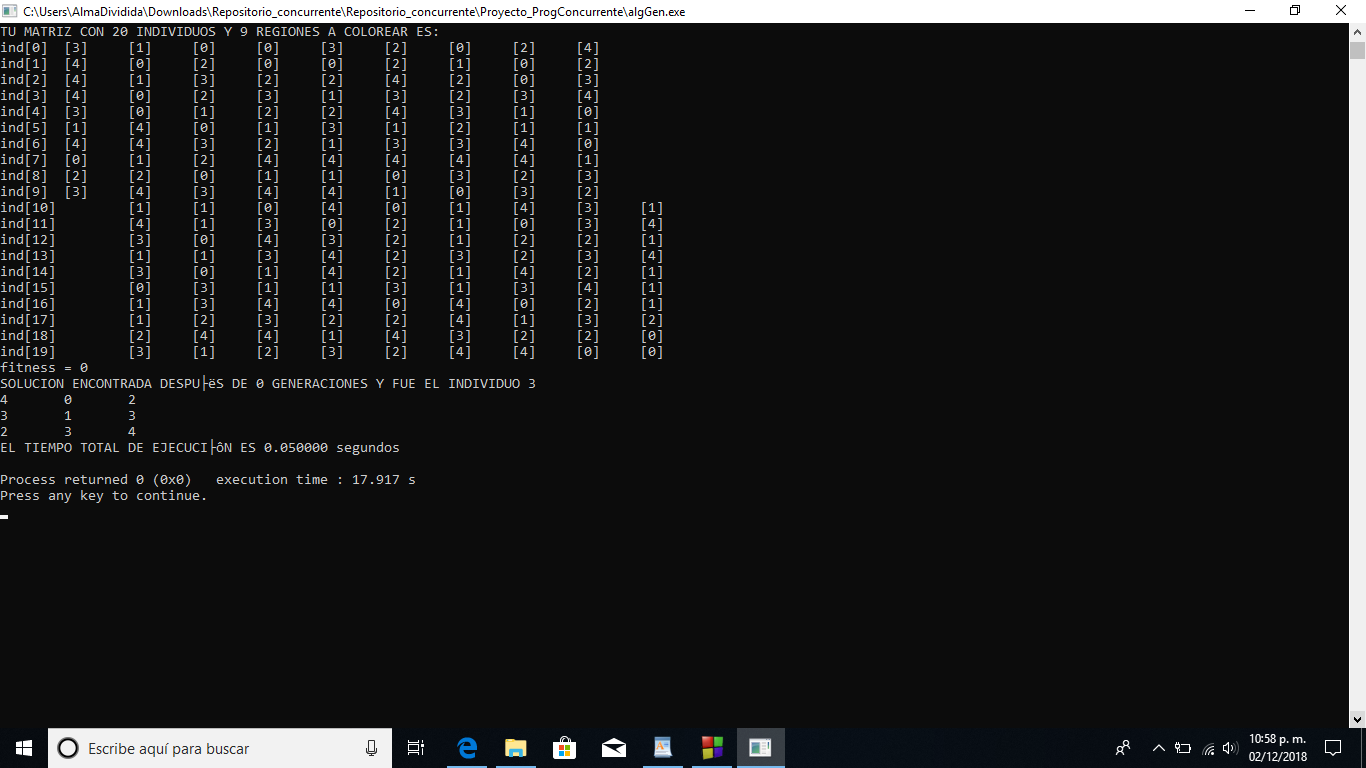
Número de colores: 5

Grafo: 3 x 3

Probabilidad de cruza: 0.75

Probabilidad de mutación: 0.01





**Segunda ejecución.**

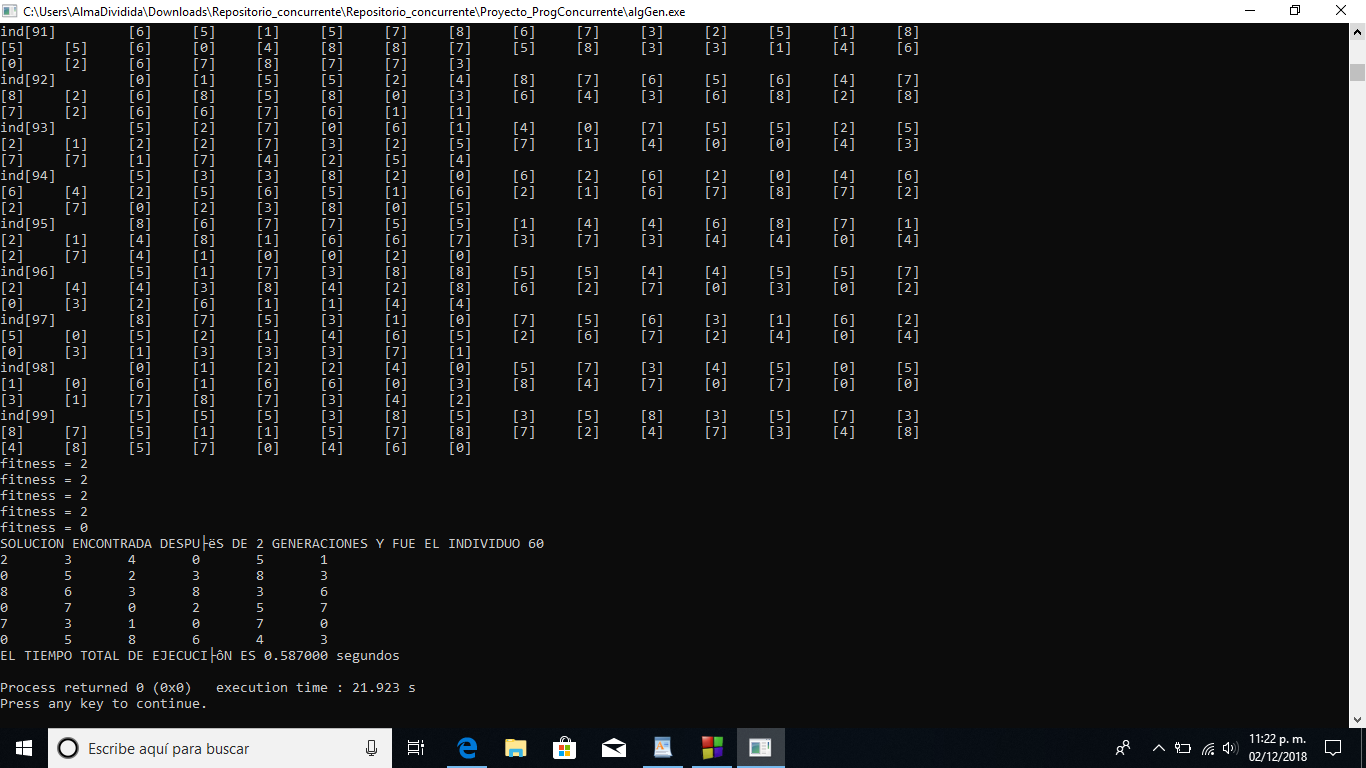
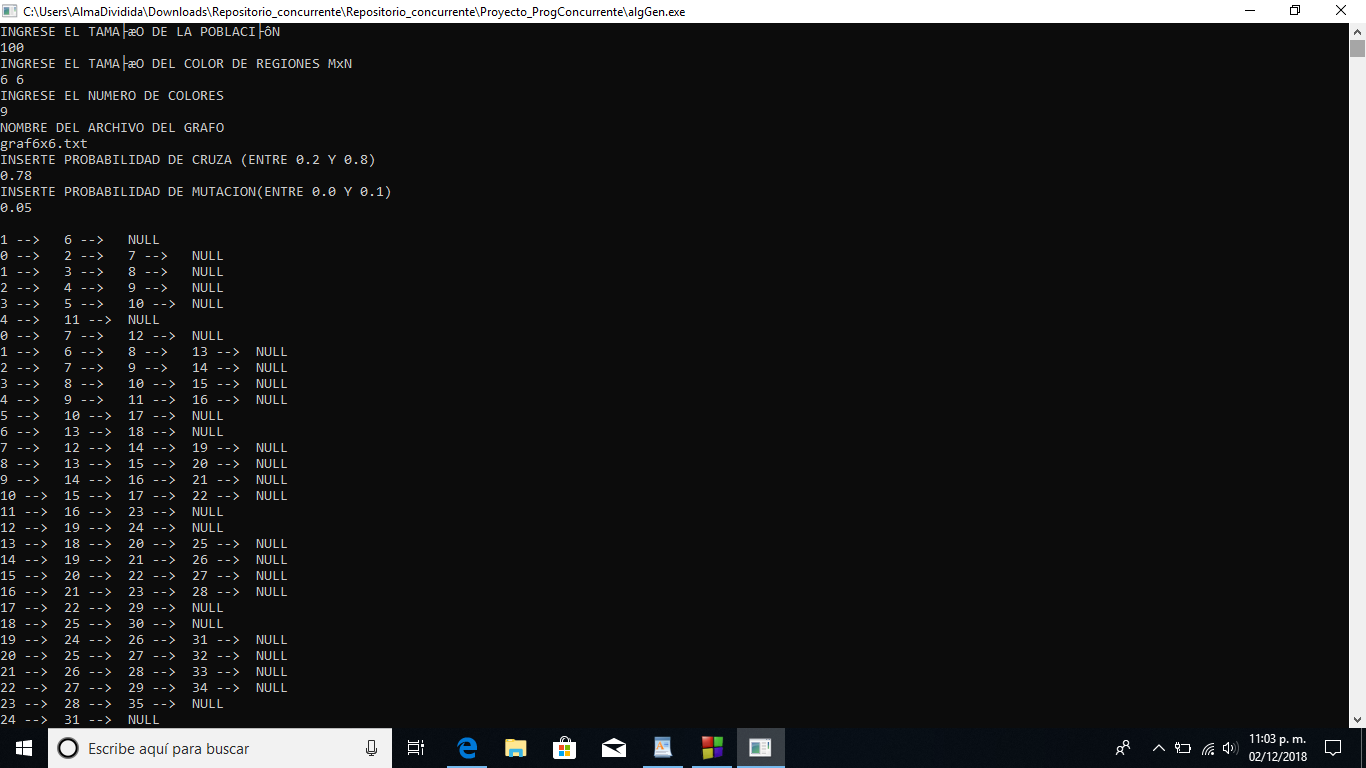
Tamaño de la población inicial: 100

Número de colores: 9

Grafo: 6 x 6

Probabilidad de cruza: 0.78

Probabilidad de mutación:0.05



**Tercera ejecución.**

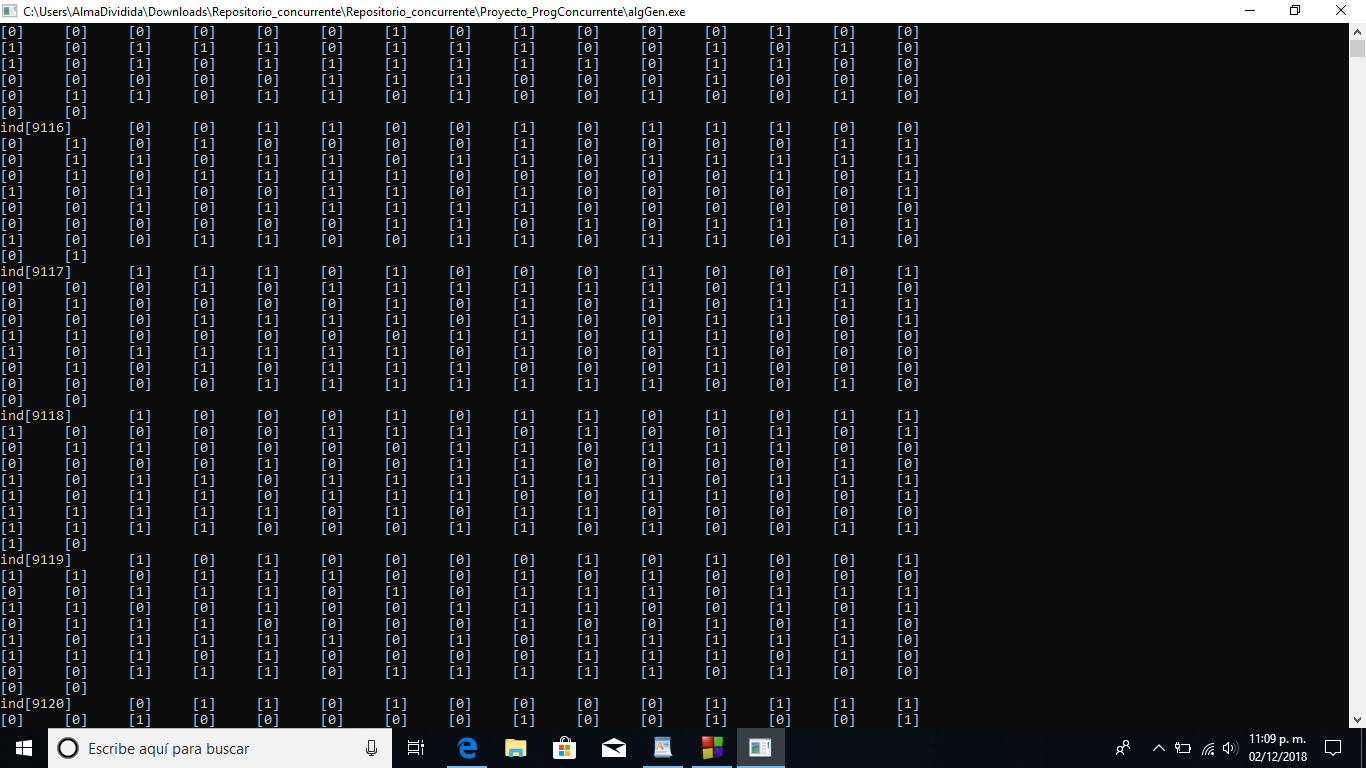
Tamaño de la población inicial: 10000

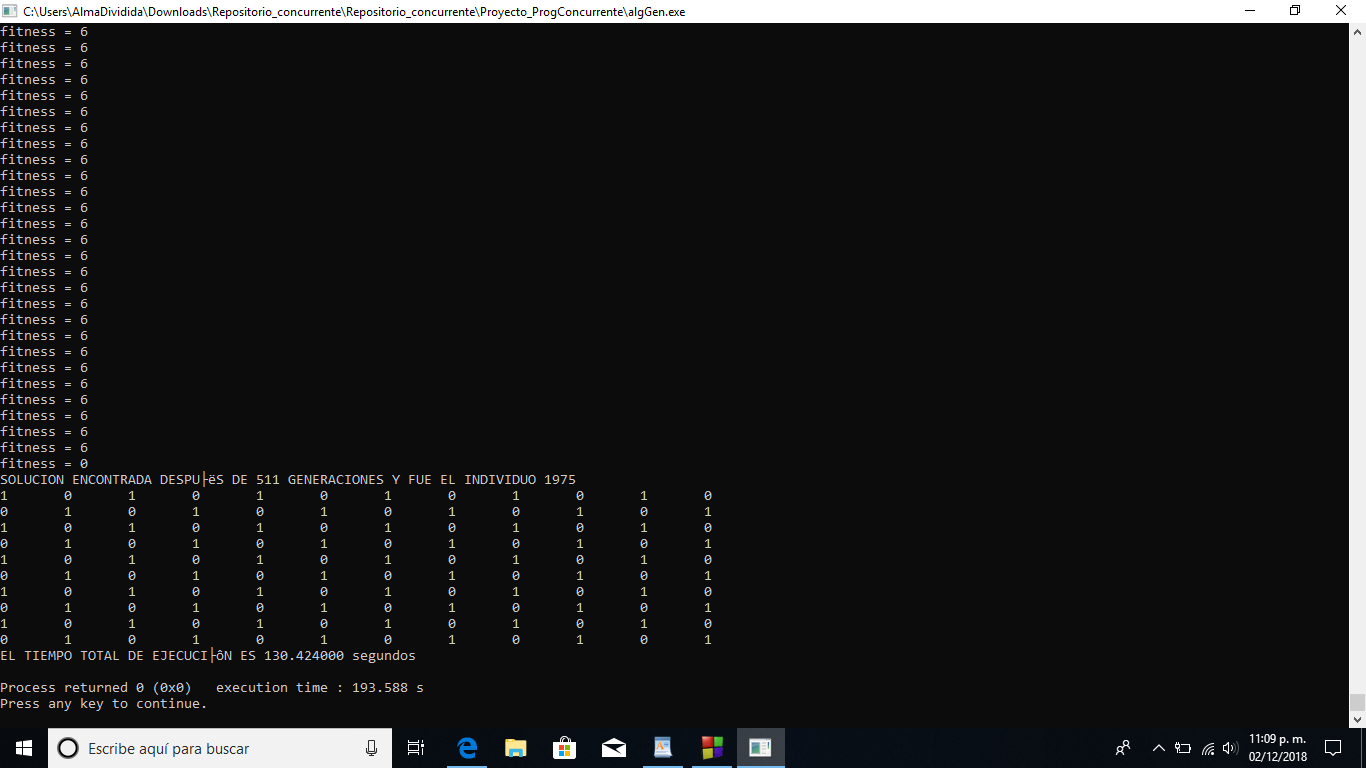
Número de colores: 2

Grafo: 10 x 12

Probabilidad de cruza: 0.80

Probabilidad de mutación: 0.1





**ANÁLISIS DE LOS RESULTADOS.**

**Primera ejecución.**

La primera generación encontró a la mejor solución (con un fitness de 0) en la población inicial por lo que no fue necesario hacer cruce y mutación.

Generaciones: 0

Mejor fitness: 0

Mejor individuo: 14

Tiempo de ejecución: 0.05 s

Solución encontrada:

|  |  |  |
| --- | --- | --- |
| 4 | 0 | 2 |
| 3 | 1 | 3 |
| 2 | 3 | 4 |

**Segunda ejecución.**

El algoritmo obtuvo el resultado óptimo después de 2 generaciones, por lo que, si realizo el proceso de cruzamiento y mutación dos veces, después de esto se encontró el mejor individuo (número 60).

Generaciones: 2

Mejor fitness: 0

Mejor individuo: 60

Tiempo de ejecución: 0.587 s

Solución encontrada:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 2 | 3 | 4 | 0 | 0 | 1 |
| 0 | 5 | 2 | 3 | 8 | 3 |
| 8 | 6 | 3 | 8 | 3 | 6 |
| 0 | 7 | 0 | 2 | 5 | 7 |
| 7 | 3 | 1 | 0 | 7 | 0 |
| 0 | 5 | 8 | 6 | 4 | 3 |

**Tercera ejecución.**

El algoritmo requirió 511 generaciones, por lo que se repitió 511 veces el proceso de mutación y cruzamiento para llegar al resultado óptimo, el individuo número 1975.

Generaciones: 511

Mejor fitness: 0

Mejor individuo: 1975

Tiempo de ejecución: 130.1424 s

Solución encontrada:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |

**BIBLIOGRAFÍA**

<https://es.wikipedia.org/wiki/Grafo_regular>

<http://eio.usc.es/pub/mte/descargas/ProyectosFinMaster/Proyecto_1463.pdf>

[https://es.wikipedia.org/wiki/Coloración\_de\_grafos](https://es.wikipedia.org/wiki/Coloraci%F3n_de_grafos)

<https://es.wikipedia.org/wiki/Teor%C3%ADa_de_grafos>

<https://es.wikipedia.org/wiki/NP-completo>