ReporteComplejidad

Víctor Federico Torres Trejo
Alejandro Maldonado Vázquez

12 de abril de 2024

1 K-coloracion es NP

Para demostrar que es NP, sea la entrada una grafica G y entero k, se propone un certificado es una funcion la cual $f : V \to N$, o sea mapea cada vertice a un numero natural, el cual es menor o igual a k, el cual va a representar el color de cada vertice.
Ahora proponemos la maquina de turing verificadora, para cada vertice $v \in V$, para cada vecino de el, verificara que no hay algun vecino u de v tal que $f(v) = f(u)$, es decir no hay dos vecinos con la misma coloracion. Como a lo mas checa todas las aristas en G, entonces podemos acotar la complejidad a $O(V^2)$.

2 Codificacion de la grafica

Se usara la misma codificacion que la practica anterior:
Para la codificacion, solicitaremos que la entrada sea de la forma:
k
n
$(r, v), (v, u), (u, t), \ldots$
Donde $k >= 2, n >= 1$ y donde $r, v, u, t \in V$ lo cual quiere decir que el numero que los denota tiene que ser menor a n, escrito con tuplas delimitadas por parentesis, cada tupla se separa por una coma. **Certificado** Tenemos nuestra grafica G con vertices nombrados desde 1 hasta n, entonces como certificado tomamos una propuesta C', y para representarlo usaremos una lista de numeros entre 1 y k, donde cada la posicion denota el vertice y el numero en la posicion denota el color, esta lista sera nuestro certificado de tamaño n, que es tamaño polinomial los numeros dentro de ella estan acotados de 1 a k para que sea un certificado valido, tomemos el ejemplar:
$k = 2$ y para la grafica $V = \{v_1, v_2, v_3, v_4\}$, $E = \{(1, 2), (2, 3), (3, 4), (4, 1)$ con la codificacion:
2
4
(1,2),(2,3),(3,4),(4,1)

Y tomamos al certificado: f(1)=1, f(2)=2, f(3)=1, f(4)=2, entonces el certificado sera representado como $1, 2, 1, 2$ que se codificara:
(1,2,1,2)
Si ahora ponemos el certificado: f(1)=2, f(2)=1, f(3)=1, f(4)=2 entonces el certificado sera representado como $2, 1, 1, 2$ que se codificara:
(2,1,1,2)

3 Algoritmo de certificación

El algoritmo de certificación consiste en leer los documentos input donde viene la codificación de la gráfica, esto es, su coloración k, n su número de vértices y las adyacencias.
Usando la k coloración se genera de forma aleatoria una lista con n elementos (para cada vértice) en donde sus elementos es u número de 1 a k. este se genera y se escribe en el nuevo documento dentro de un for para al final generar un archivo cuyo nombre se especifica en consola de la siguiente manera: certificado# donde al final tu especificas el número de certificado que quieres crear (excepto si ya existe un certificado con ese número) pero si se utiliza el número 0 se crea un certificado el cuál si cuenta con una coloración.

4 Algoritmo de Verificacion

```
def verifica_k_coloracion(g:Grafica, certificado:list):
    for i in range(1, g.get_n()):
        for vecino in g.get_neighbors(i):
            if certificado[i - 1] > k or certificado[i - 1] < 1:
                return False
            if certificado[i - 1] == certificado[vecino - 1]:
                return False
    return True
```

Basicamente lo que el codigo hace es que para cada vertice en V, obtiene sus vecinos y revisa si el color del vertice no es igual al de sus vecinos, en el caso de que si entonces regresamos falso, en el caso de que no entonces si terminamos de revisar todos los vertices en V entonces decimos que el certificado si soluciona el problema. Certificado esta codificado como ya lo describimos y la grafica se construye sabiendo su numero de vertices y la lista de aristas que la conforman. El if solo es para verificar que los colores sean los de acorde a la k coloracion.O sea no haya colores no validos. En el caso de que los haya, no es una solucion. Por lo tanto no soluciona

5 Algoritmos extra

Para leer las entradas solo leemos linea por linea de input.txt, el certificado es una lista ya especificada por lo que solo eso lo convertimos a lista, la clase grafica

se crea a partir de la lista de aristas, donde a cada vertice le dice que vecino le va a tocar, por ulitmo en el main, solo leemos k,n,aristas y el certificado y corremos el algoritmo de verificacion descrito arriba, dependiendo de eso decimos si es o no solucion

6 Pruebas

Ejemplos de ejecución

Ejemplar K = 2

El ejemplar es el siguiente ya con codificación:

K = 2
V = 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22
A = (1,2), (1,3), (1,4), (1,5), (1,6), (1,7), (1,8), (1,9), (1,10), (2,18), (3,18), (4,18), (5,11), (6,11), (7,11), (8,22), (9,22), (10,22), (11,12), (11,13), (11,14), (11,15), (11,16), (11,17), (12,18), (13,18), (14,18), (15,22), (16,22), (17,22), (18,19), (18,20), (18,21), (19,22), (20,22), (21,22)

Certificado 0

Se genera con la siguiente codificación:

(2,1,1,1,1,1,1,1,1,2,1,1,1,1,1,1,2,1,1,1,2)

Ejemplar K = 3

El ejemplar es el siguiente ya con codificación:

K = 3
V = 1,2,3,4,5,6,7,8,9,10,11,12,13
A = (1,2), (1,3), (1,4), (2,3), (2,5), (3,6), (3,7), (3,8), (4,8), (4,9), (5,6), (5,10), (6,10), (6,11), (6,7), (7,11), (7,8), (8,11), (8,9), (9,12), (10,13), (11,12), (11,13), (12,13)

Certificado 0

Se genera con la siguiente codificación:

(1,2,3,2,3,1,2,1,3,2,3,2,1)

Ejemplar K = 4

El ejemplar es el siguiente ya con codificación:

K = 4
V = 1,2,3,4,5,6,7,8,9,10,11
A = (1,2), (1,3), (1,4), (1,5), (1,6), (2,8), (2,11), (3,7), (3,9), (4,8), (4,10), (5,9), (5,11), (6,7), (6,10), (7,8), (8,9), (9,10), (10,11)

Certificado 0

Se genera con la siguiente codificación:

(1,2,2,2,2,2,3,4,3,4,3)

7 Consola Certificado

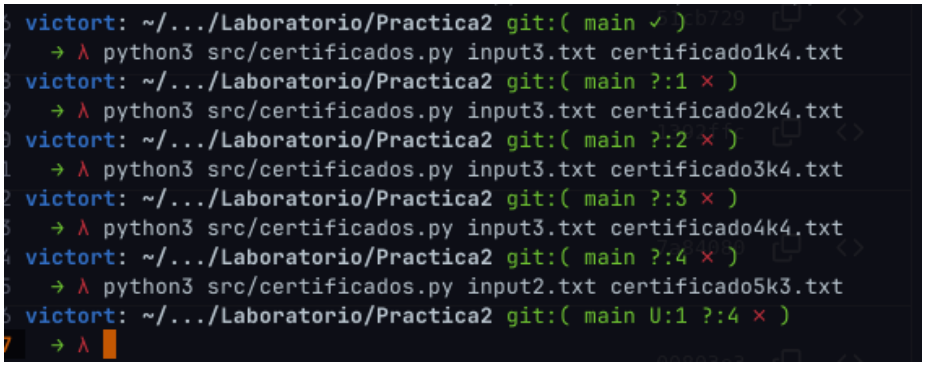

Figure 1: Generando certificados para k=4

8 Consola verificacion

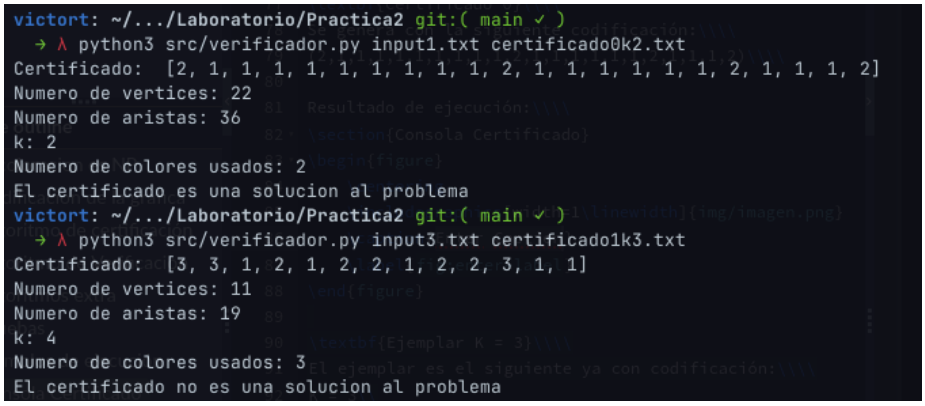

Figure 2: Ejecución de verificación con uno correcto y uno generado aleatoriamente

9 Ejecucion

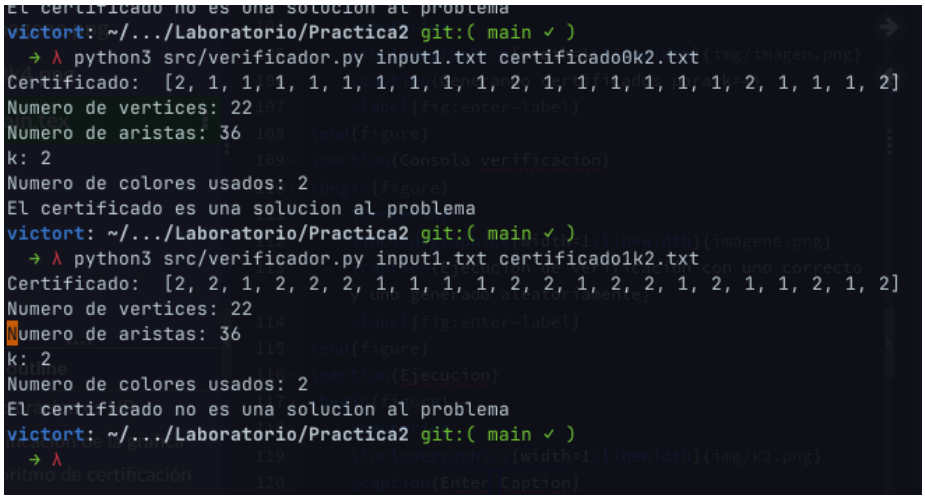

Figure 3: Ejecucion k=2

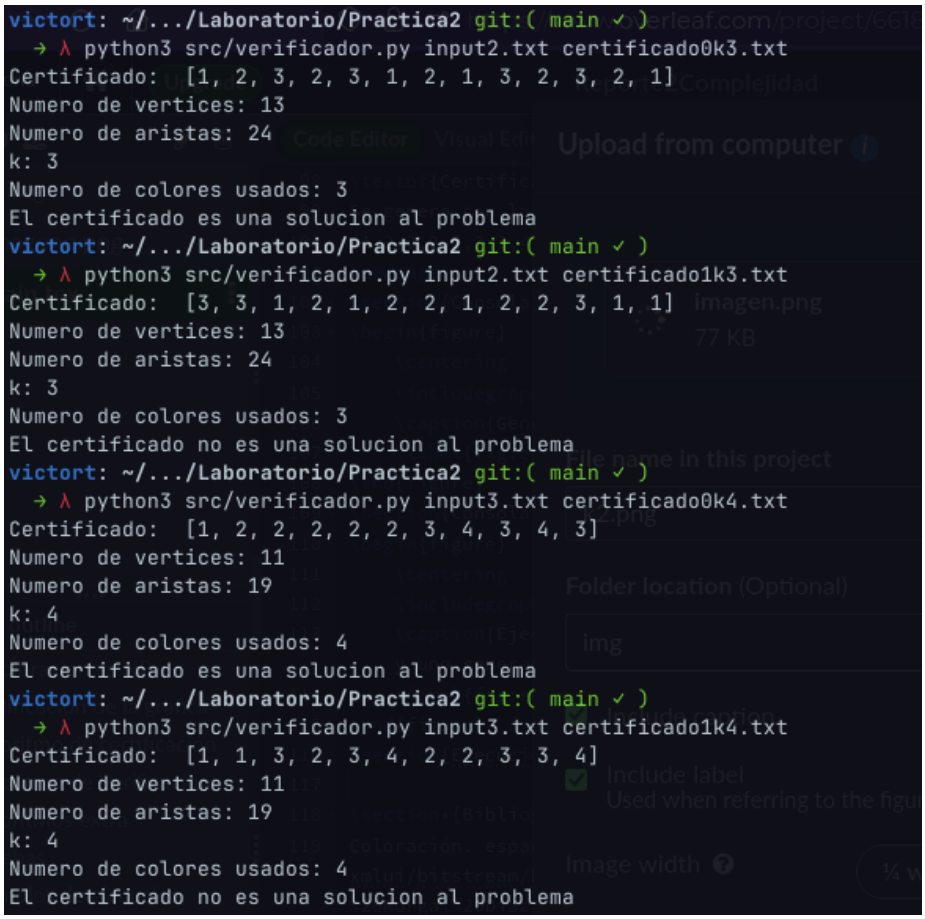

Figure 4: Ejecucion k=3,4

Bibliografía

Coloración. español. url: https://repositorio.unican.es/xmlui/bitstream/handle/10902/3099/Maria%20Rosa%20Murga%20Diaz.pdf?sequence=4