

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE CIENCIAS

Complejidad Computacional 2024-1



---

***PRÁCTICA 2***

---

**PROFESOR:**

Oscar Hernández Constantino

**AYUDANTE:**

Malinali González Lara

**INTEGRANTES:**

Manjarrez Angeles Valeria Fernanda - 317234785

Pedro Mendez Jose Manuel - 315073120

Sánchez Reza Neider - 317020931

18 de octubre de 2023

## Índice

<b>1. Demostración de pertenencia de Ruta Inducida en <i>NP</i></b>	<b>2</b>
<b>2. Descripción del certificado y esquema de codificación a utilizar</b>	<b>2</b>
2.1. Especificación y codificación del certificado . . . . .	2
2.2. Ejemplo de certificado y esquema de codificación . . . . .	2
2.2.1. Representación visual de ejemplar . . . . .	3
2.2.2. Esquema de codificación . . . . .	3
2.2.3. Certificado . . . . .	3
2.2.4. Representación del certificado en el esquema de codificación	3
<b>3. Descripción del algoritmo de generación aleatoria de certificados</b>	<b>4</b>
<b>4. Descripción del algoritmo de verificación</b>	<b>5</b>
4.1. Descripción del algoritmo . . . . .	5
4.2. Pseudocódigo del algoritmo . . . . .	5
<b>5. Pruebas de ejecución</b>	<b>6</b>
5.1. Ejemplar 1 . . . . .	6
5.1.1. Certificado 1 . . . . .	6
5.1.2. Certificado 2 . . . . .	7
5.1.3. Certificado 3 . . . . .	8
5.1.4. Certificado 4 . . . . .	8
5.1.5. Certificado 5 . . . . .	9
5.2. Ejemplar 2 . . . . .	10
5.2.1. Certificado 1 . . . . .	10
5.2.2. Certificado 2 . . . . .	11
5.2.3. Certificado 3 . . . . .	12
5.2.4. Certificado 4 . . . . .	12
5.2.5. Certificado 5 . . . . .	13
5.3. Ejemplar 3 . . . . .	14
5.3.1. Certificado 1 . . . . .	14
5.3.2. Certificado 2 . . . . .	15
5.3.3. Certificado 3 . . . . .	16
5.3.4. Certificado 4 . . . . .	16
5.3.5. Certificado 5 . . . . .	17

## 1. Demostración de pertenencia de Ruta Inducida en $NP$

Consideremos el siguiente problema:

### RUTA INDUCIDA

- EJEMPLAR: Una gráfica  $G = (V, E)$ , un entero positivo  $K \leq |V|$ .
- PREGUNTA: ¿Existe un subconjunto  $V' \subseteq V$ , con  $|V'| \geq K$ , tal que la subgráfica inducida por  $V'$  es una ruta simple con  $|V'|$  vértices?

Demostraremos que el lenguaje asociado al problema anterior pertenece a la clase de complejidad  $NP$ .

*Demostración.* Para demostrar que este problema pertenece a la clase  $NP$ , debemos mostrar que, dado un certificado podemos verificar en tiempo polinomial si dicha propuesta es una solución válida.

Sea  $G = (V, E)$  una gráfica y  $u$  un subconjunto de vértices de  $G$ , tal que  $|u| = K$ .

Notemos que  $|u| \in O(|V(G)|)$  es polinomial en el tamaño de la entrada.

Proponemos el siguiente algoritmo:

1. Sea  $u_1$  el primer elemento de la lista de vértices  $u$ .
2. Verificamos que solo existe una adyacencia (arista) con alguno de los otros  $u_i$  i-ésimos elementos de la lista.
3. Verificamos que el vértice con el que se tiene la adyacencia no sea un vértice anteriormente visitado.
4. Si se cumplen todas las verificaciones, probamos ahora para el siguiente elemento  $u_2$  y repetimos lo anterior hasta  $u_k$ . Si alguna verificación no se cumple, rechazamos.

Realizar estas verificaciones toma un tiempo polinomial, pues para  $n = |V(G)|$ , se efectúan  $n-i$  verificaciones para el  $i$ -ésimo vértice, con un total de  $n$  vértices, dando un total de  $\frac{n(n-1)}{2}$  verificaciones. Por lo tanto el problema de la ruta inducida pertenece a la clase  $NP$ .

□

## 2. Descripción del certificado y esquema de codificación a utilizar

### 2.1. Especificación y codificación del certificado

De acuerdo con el planteamiento del problema, proponemos como certificado del problema un subconjunto de vértices de la gráfica  $G$ , de tamaño al menos  $K$ .

Al momento de codificar el certificado anterior, usaremos una lista o arreglo ordenado de vértices de  $G$  y de tamaño  $K$ . Para ello, retomaremos el esquema de codificación basado en texto para ejemplares del problema, usado en la práctica anterior, el cual consiste de:

1. La primera línea contiene el valor de  $K$  del ejemplar.
2. La segunda línea contiene los  $n$  vértices de  $G$ , numerados iniciando en 1, separados por coma.
3. Las  $m$  líneas siguientes corresponden con las  $m$  aristas de  $G$ , cada una en una línea en el formato  $a, b$ , donde  $a, b \in V(G)$ .

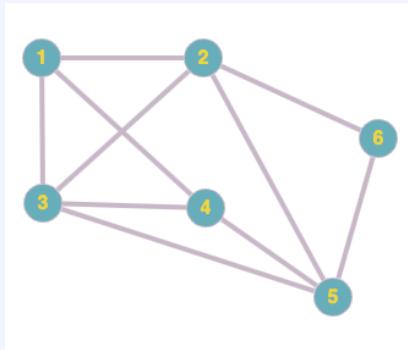
Así, la codificación del certificado corresponde con una lista de vértices similar a la de la segunda línea, en la codificación del ejemplar.

### 2.2. Ejemplo de certificado y esquema de codificación

Detallaremos el caso para un ejemplar con  $|V| \geq 5, |E| \geq 10, k = 3$ .

### 2.2.1. Representación visual de ejemplar

Proponemos el siguiente ejemplar que cumple con las características requeridas de que  $|V| \geq 5$  y  $|E| \geq 10$ .

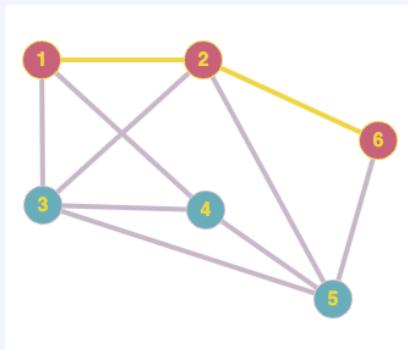


Podemos notar que sí existe un subconjunto de vértices  $V' \subseteq V$  tal que la subgráfica inducida es una ruta simple y el número de vértices de dicha subgráfica es mayor o igual a  $k = 3$ . El cuál será propuesto como nuestro certificado.

### 2.2.2. Esquema de codificación

Retomando el esquema de codificación utilizado para los ejemplares del problema entonces, el primer número (3) indica el valor de  $K$  que representa el número mínimo de vértices requerido para formar la ruta simple. La línea siguiente (1,2,3,4,5,6) representa todos los vértices de la gráfica terminando con las líneas subsiguientes representan las aristas presentes en la gráfica (Estamos hablando de la siguiente imagen).

El certificado propuesto, en este caso, es el subconjunto de vértices 1, 2, 6 que se codificará de manera similar al anterior esquema de codificación, como la segunda línea que representaba los vértices de nuestra gráfica, en este caso representara los vértices de nuestra posible ruta, asumiendo que el orden importa, es decir, si tenemos 1,2,6 nos referimos a que tenemos una gráfica: (1,2), (2,6), esto representa una ruta simple que pasa por los vértices 1,2,6.



### 2.2.3. Certificado

Sea  $G = (V, E)$  una gráfica con  $|V| \geq 5$  y  $|E| \geq 10$  junto con un entero positivo  $K = 3$  (el explicado anteriormente). Un certificado podría ser un subconjunto (en forma de lista(arreglo de números en Java)) de vértices de  $G$  que forman una ruta simple con el número de vértices especificado, dado por  $k$ , en este caso 3. Por lo que nuestro certificado podría ser  $C_1 = \{1, 2, 6\}$ .

### 2.2.4. Representación del certificado en el esquema de codificación

1 , 2 , 6
-----------

Listing 1: Representación de nuestro certificado en un archivo txt.

### 3. Descripción del algoritmo de generación aleatoria de certificados

A grandes rasgos, el algoritmo selecciona aleatoriamente, de la lista de vértices de la codificación de la gráfica (en el esquema mencionado anteriormente),  $K$  vértices para incluirlos en una nueva lista. Lo anterior se logra usando estructuras de datos adecuadas del lenguaje de programación JAVA. Esta última lista (en forma de arreglo de JAVA) corresponderá con el certificado codificado. Los detalles de la implementación se describen a continuación.

En la clase `GraphCertificateGenerator` de JAVA se implementan métodos para leer ejemplares del problema, generar y escribir certificados. Tenemos dos métodos principales:

- `certificateGenerator`:

Lee la representación codificada de un ejemplar (gráfica y valor  $K$ ) desde un archivo de entrada especificado. Se obtiene el número de aristas y los vértices de la gráfica y se extrae el valor de  $K$ . Así, de los vértices en el ejemplar dado, se genera un certificado seleccionando aleatoriamente  $K$  vértices de la gráfica. Enseguida, se guardan en el archivo de salida especificado.

La función aleatoria se encuentra en el siguiente fragmento de código:

```
// Prevents generating a loop
if (kValue <= nOfV){
    while (selectedVertices.size() < kValue) {
        int randomIndex = random.nextInt(v.length);
        selectedVertices.add(v[randomIndex]);
    }
}
```

Figura 1: Parte del código que genera un nuevo certificado.

Finalmente, se imprime el número de vértices, el número de aristas, el valor de  $K$ , el certificado generado y el nombre del archivo de salida.

- `decodeCertificate`:

Lee la representación codificada de un certificado desde un archivo especificado. Divide la línea leída en el archivo en una matriz de cadenas utilizando la coma como delimitador, convertimos cada cadena en un entero y almacena los enteros en un arreglo de enteros.

Además, maneja adecuadamente cualquier excepción que pueda ocurrir durante este proceso e imprime el mensaje de error. Finalmente, devuelve el arreglo de enteros que representa nuestro certificado leído y previamente generado.

El código del método descrito anteriormente es el siguiente:

```
1 public static int[] decodeCertificate(String in){
2     try {
3         File dataG = new File(in);
4         Scanner scanner = new Scanner(dataG);
5
6         String line = scanner.nextLine();
7         String[] strArray = line.split(",");
8         int[] numbers = new int[strArray.length];
9         for (int i = 0; i < strArray.length; i++) {
10             numbers[i] = Integer.parseInt(strArray[i]);
11         }
12     } catch (Exception e) {
13         System.out.println("Error reading certificate file");
14         e.printStackTrace();
15     }
16 }
```

```

11     }
12
13     return numbers;
14 } catch (Exception e) {
15     System.out.println(e);
16     return new int[0]; // Retorna un arreglo vacío si ocurre
17     // alguna excepción.
18 }

```

Listing 2: Código Java de decodeCertificate.

## 4. Descripción del algoritmo de verificación

Como vimos anteriormente, el problema de RUTA INDUCIDA está en la clase *NP*. Así, podemos dar un algoritmo eficiente para verificar si un subconjunto de vértices dado cumple con inducir una ruta simple en una gráfica.

### 4.1. Descripción del algoritmo

El algoritmo consiste en verificar si existen únicamente las aristas necesarias para formar una ruta simple con los vértices especificados en el certificado. Es decir, se toma el primer vértice de la lista y se verifica que exista exactamente una adyacencia con el vértice siguiente del listado. El proceso se repite para cada vértice en la lista.

Si es posible recorrer toda la lista de vértices en el certificado, entonces la lista induce una ruta simple en *G*. En caso contrario, si se encuentran más de una adyacencia o menos de una para algún vértice en el listado, entonces la lista no induce una ruta simple en *G*.

### 4.2. Pseudocódigo del algoritmo

A continuación presentamos el pseudocódigo correspondiente al algoritmo descrito anteriormente:

**Algoritmo VerificaCertificado:**

Parámetros:

grafica: codificación en matriz del ejemplar.  
certificado: codificación del certificado para validación.

```

VerificaCertificado(grafica, certificado):
    for v_i in grafica.vertices:
        verticeOrigenVisitado = false
        for v_j in grafica.vertices - v_i: // Se verifican el resto de vértices
            if v_j == v_i.siguiente:
                if v_i,v_j in grafica.aristas:
                    if verticeOrigenVisitado:
                        return false // El certificado no es válido
                    else:
                        verticeOrigenVisitado = true
            else if not verticeOrigenVisitado:
                return false // No hay aristas a este vértice
    return true // El certificado es válido

```

Nótese que como la gráfica de entrada es no dirigida, entonces basta verificar la arista en un sentido. La variable `verticeOrigenVisitado` indica si para el vértice que estamos analizando ya fue encontrada una adyacencia.

Del pseudocódigo anterior, podemos ver que, para  $n = |V(G)|$ , el ciclo más anidado se ejecuta  $n - 1$  veces para el primer vértice,  $n - 2$  veces para el segundo y así sucesivamente. De este modo, la cantidad de veces que se ejecuta el ciclo más anidado en todo el algoritmo está dado por la expresión

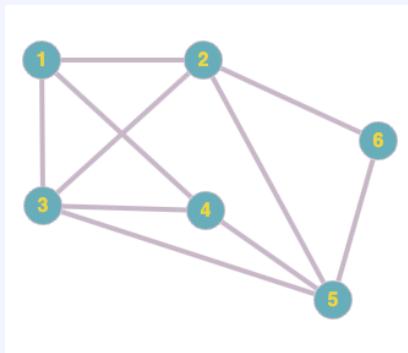
$$(n - 1) + (n - 2) + \dots + 2 + 1$$

Lo cual sabemos es igual a  $\frac{n(n-1)}{2}$ . Así, tenemos que el algoritmo se ejecuta en tiempo  $O(n^2)$ , es decir, el algoritmo de verificación toma tiempo polinomial.

## 5. Pruebas de ejecución

### 5.1. Ejemplar 1

Proponemos el siguiente exemplar con su representación correspondiente en el esquema de codificación



```

1 3
2 1 , 2 , 3 , 4 , 5 , 6
3 1 , 2
4 1 , 3
5 1 , 4
6 2 , 3
7 2 , 5
8 2 , 6
9 3 , 4
10 3 , 5
11 4 , 5
12 5 , 6

```

Listing 3: Representación de nuestra gráfica en el archivo example1.txt

#### 5.1.1. Certificado 1

Primer certificado generado, con su representación en el esquema de codificación y el resultado obtenido al ejecutar el algoritmo de verificación.

```
1 3 , 4 , 6
```

Listing 4: Representación de nuestro certificado en el archivo example1\_cert1.txt

```
[valeriamanjreez@MacBook-Air-de-Valeria src % java induced_path.ValidationAlgorithm
hm .../examples/graphs/example1.txt .../examples/certificates/example1_cert1.txt

Reading the file .../examples/graphs/example1.txt

Read graph:
0 1 1 1 0 0
1 0 1 0 1 1
1 1 0 1 1 0
1 0 1 0 1 0
0 1 1 1 0 1
0 1 0 0 1 0

Value of parameter K: 3

The given certificate 3,4,6 is NOT a valid solution.
valeriamanjreez@MacBook-Air-de-Valeria src % _
```

### 5.1.2. Certificado 2

Segundo certificado generado, con su representación en el esquema de codificación y el resultado obtenido al ejecutar el algoritmo de verificación.

```
1 6 , 4 , 2
```

Listing 5: Representación de nuestro certificado en el archivo example1\_cert2.txt

```
[valeriamanjreez@MacBook-Air-de-Valeria src % java induced_path.ValidationAlgorithm
hm .../examples/graphs/example1.txt .../examples/certificates/example1_cert2.txt

Reading the file .../examples/graphs/example1.txt

Read graph:
0 1 1 1 0 0
1 0 1 0 1 1
1 1 0 1 1 0
1 0 1 0 1 0
0 1 1 1 0 1
0 1 0 0 1 0

Value of parameter K: 3

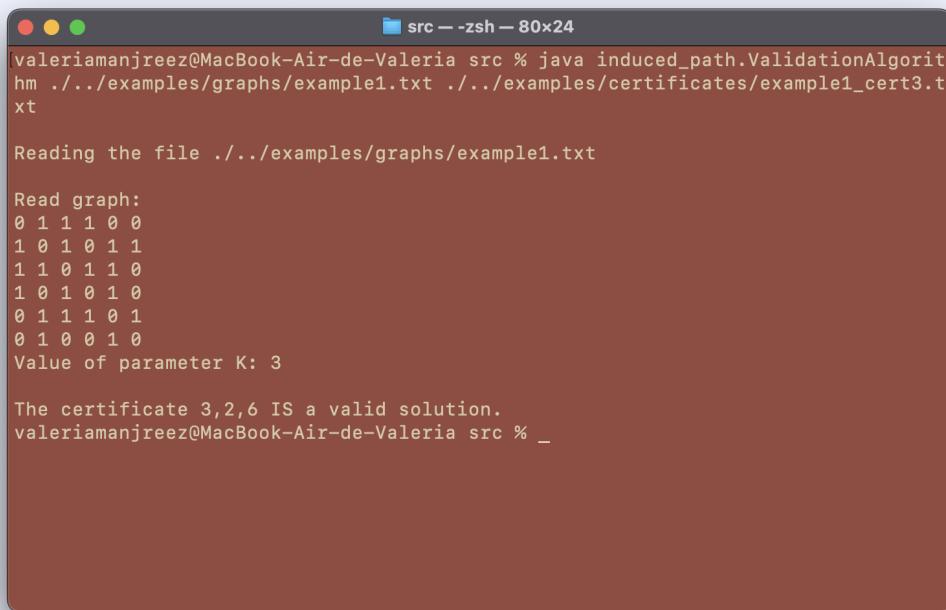
The given certificate 6,4,2 is NOT a valid solution.
valeriamanjreez@MacBook-Air-de-Valeria src % _
```

### 5.1.3. Certificado 3

Tercer certificado generado, con su representación en el esquema de codificación y el resultado obtenido al ejecutar el algoritmo de verificación.

1 3 , 2 , 6

Listing 6: Representación de nuestro certificado en el archivo example1\_cert3.txt

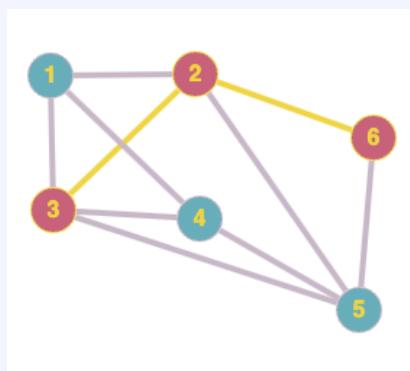


```
src -- zsh -- 80x24
valeriamanjreez@MacBook-Air-de-Valeria src % java induced_path.ValidationAlgorithm ./../examples/graphs/example1.txt ./../examples/certificates/example1_cert3.txt

Reading the file ./../examples/graphs/example1.txt

Read graph:
0 1 1 1 0 0
1 0 1 0 1 1
1 1 0 1 1 0
1 0 1 0 1 0
0 1 1 1 0 1
0 1 0 0 1 0
Value of parameter K: 3

The certificate 3,2,6 IS a valid solution.
valeriamanjreez@MacBook-Air-de-Valeria src % _
```

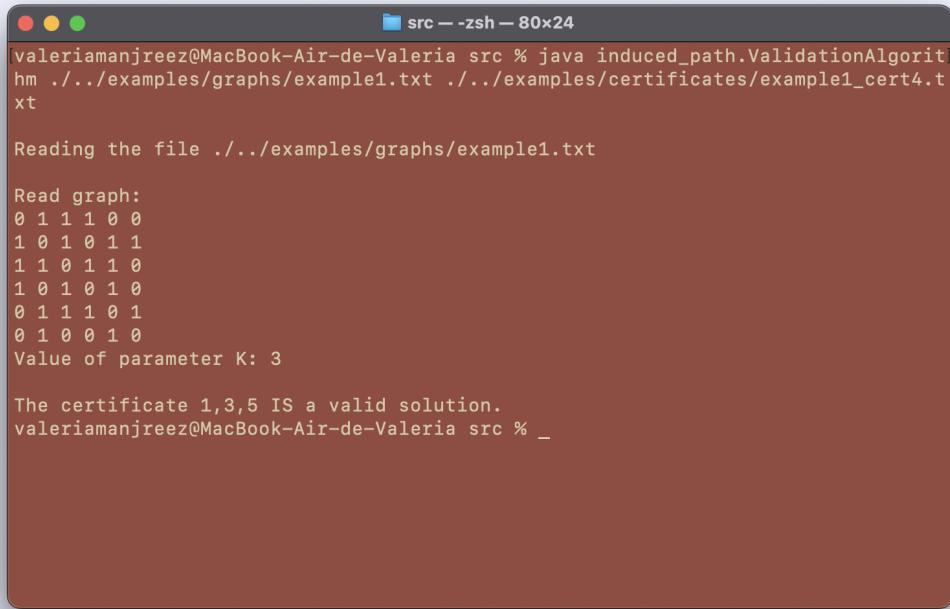


### 5.1.4. Certificado 4

Cuarto certificado generado, con su representación en el esquema de codificación y el resultado obtenido al ejecutar el algoritmo de verificación.

1 1 , 3 , 5

Listing 7: Representación de nuestro certificado en el archivo example1\_cert4.txt



```
[valeriamanjreez@MacBook-Air-de-Valeria src % java induced_path.ValidationAlgorithm ./../examples/graphs/example1.txt ./../examples/certificates/example1_cert4.txt

Reading the file ./../examples/graphs/example1.txt

Read graph:
0 1 1 1 0 0
1 0 1 0 1 1
1 1 0 1 1 0
1 0 1 0 1 0
0 1 1 1 0 1
0 1 0 0 1 0
Value of parameter K: 3

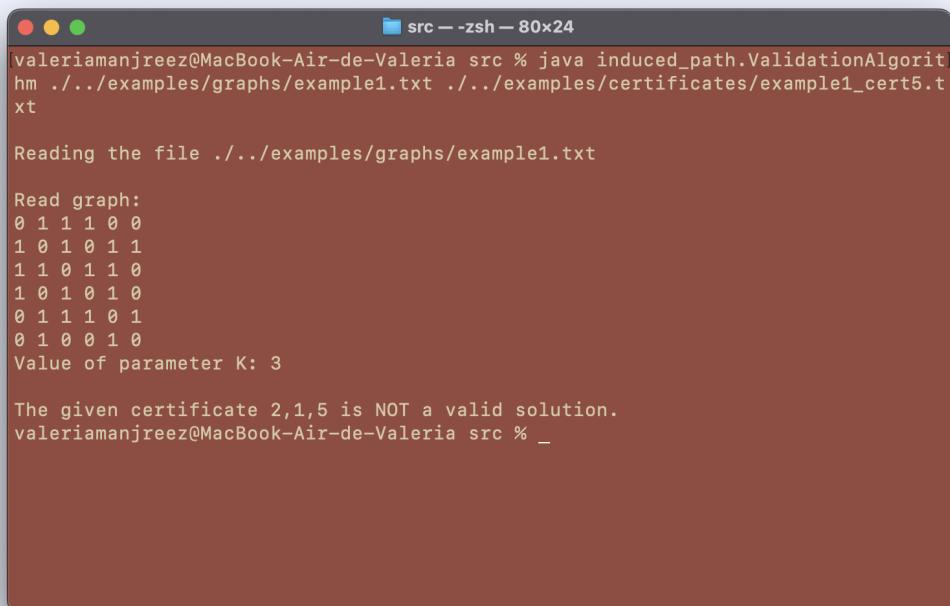
The certificate 1,3,5 IS a valid solution.
valeriamanjreez@MacBook-Air-de-Valeria src % _
```

### 5.1.5. Certificado 5

Quinto certificado generado, con su representación en el esquema de codificación y el resultado obtenido al ejecutar el algoritmo de verificación.

```
1 2 , 1 , 5
```

Listing 8: Representación de nuestro certificado en el archivo example1\_cert5.txt



```
[valeriamanjreez@MacBook-Air-de-Valeria src % java induced_path.ValidationAlgorithm ./../examples/graphs/example1.txt ./../examples/certificates/example1_cert5.txt

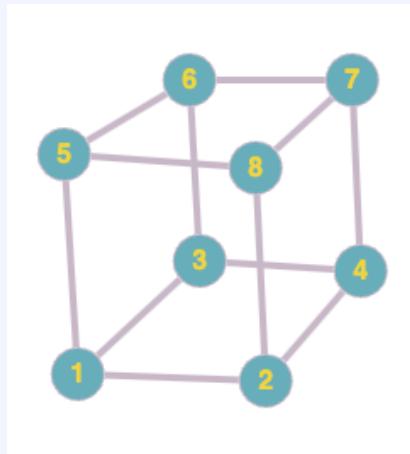
Reading the file ./../examples/graphs/example1.txt

Read graph:
0 1 1 1 0 0
1 0 1 0 1 1
1 1 0 1 1 0
1 0 1 0 1 0
0 1 1 1 0 1
0 1 0 0 1 0
Value of parameter K: 3

The given certificate 2,1,5 is NOT a valid solution.
valeriamanjreez@MacBook-Air-de-Valeria src % _
```

## 5.2. Ejemplar 2

Proponemos el siguiente ejemplar con su representación correspondiente en el esquema de codificación



```
1 4
2 1 , 2 , 3 , 4 , 5 , 6 , 7 , 8
3 1 , 2
4 1 , 4
5 1 , 5
6 2 , 3
7 2 , 8
8 3 , 4
9 3 , 7
10 4 , 6
11 5 , 6
12 5 , 8
13 6 , 7
14 7 , 8
```

Listing 9: Representación de nuestra gráfica en el archivo example2.txt

### 5.2.1. Certificado 1

Primer certificado generado, con su representación en el esquema de codificación y el resultado obtenido al ejecutar el algoritmo de verificación.

```
1 1 , 4 , 8 , 6
```

Listing 10: Representación de nuestro certificado en el archivo example2.cert1.txt

```
[valeriamanjreez@MacBook-Air-de-Valeria src % java induced_path.ValidationAlgorithm
hm .../examples/graphs/example2.txt .../examples/certificates/example2_cert1.txt

Reading the file .../examples/graphs/example2.txt

Read graph:
0 1 0 1 1 0 0 0
1 0 1 0 0 0 0 1
0 1 0 1 0 0 1 0
1 0 1 0 0 1 0 0
1 0 0 0 0 1 0 1
0 0 0 1 1 0 1 0
0 0 1 0 0 1 0 1
0 1 0 0 1 0 1 0
Value of parameter K: 4

The given certificate 1,4,8,6 is NOT a valid solution.
valeriamanjreez@MacBook-Air-de-Valeria src % _
```

### 5.2.2. Certificado 2

Segundo certificado generado, con su representación en el esquema de codificación y el resultado obtenido al ejecutar el algoritmo de verificación.

```
1 7 , 1 , 8 , 3
```

Listing 11: Representación de nuestro certificado en el archivo example2\_cert2.txt

```
[valeriamanjreez@MacBook-Air-de-Valeria src % java induced_path.ValidationAlgorithm
hm .../examples/graphs/example2.txt .../examples/certificates/example2_cert2.txt

Reading the file .../examples/graphs/example2.txt

Read graph:
0 1 0 1 1 0 0 0
1 0 1 0 0 0 0 1
0 1 0 1 0 0 1 0
1 0 1 0 0 1 0 0
1 0 0 0 0 1 0 1
0 0 0 1 1 0 1 0
0 0 1 0 0 1 0 1
0 1 0 0 1 0 1 0
Value of parameter K: 4

The given certificate 7,1,8,3 is NOT a valid solution.
valeriamanjreez@MacBook-Air-de-Valeria src % _
```

### 5.2.3. Certificado 3

Tercer certificado generado, con su representación en el esquema de codificación y el resultado obtenido al ejecutar el algoritmo de verificación.

```
1 2 , 8 , 5 , 6
```

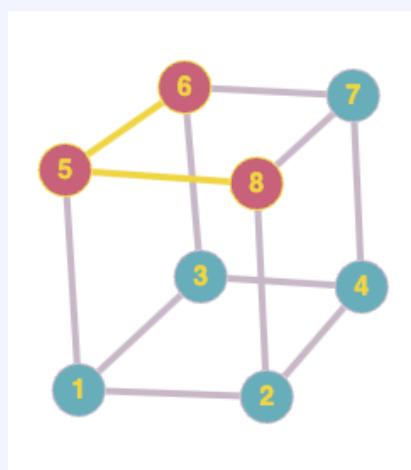
Listing 12: Representación de nuestro certificado en el archivo example2\_cert3.txt

```
src --zsh -- 80x24
valeriamanjreez@MacBook-Air-de-Valeria src % java induced_path.ValidationAlgorithm
./examples/graphs/example2.txt ./examples/certificates/example2_cert3.txt

Reading the file ./examples/graphs/example2.txt

Read graph:
0 1 0 1 1 0 0 0
1 0 1 0 0 0 0 1
0 1 0 1 0 0 1 0
1 0 1 0 0 1 0 0
1 0 0 0 0 1 0 1
0 0 0 1 1 0 1 0
0 0 1 0 0 1 0 1
0 1 0 0 1 0 1 0
Value of parameter K: 4

The certificate 2,8,5,6 IS a valid solution.
valeriamanjreez@MacBook-Air-de-Valeria src % _
```

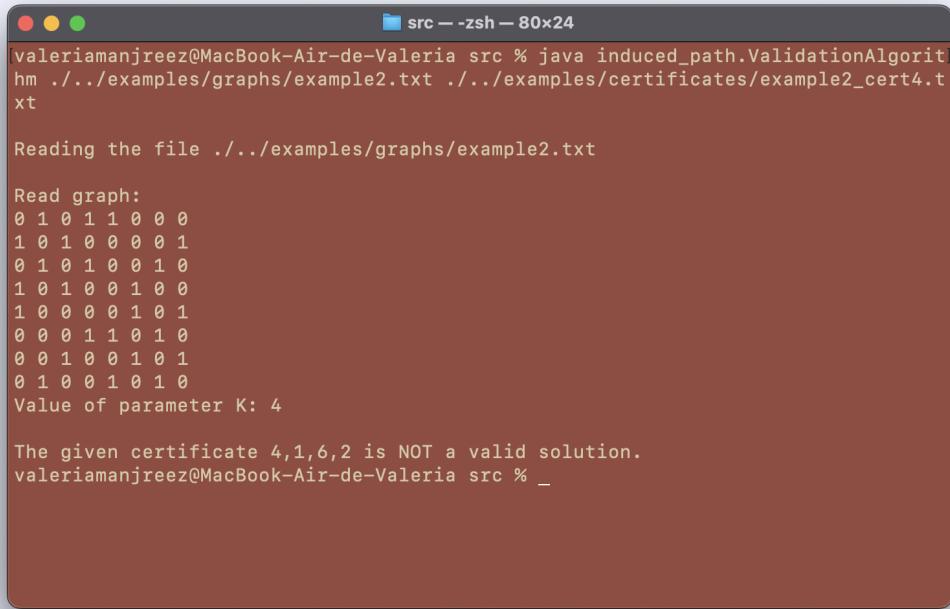


### 5.2.4. Certificado 4

Cuarto certificado generado, con su representación en el esquema de codificación y el resultado obtenido al ejecutar el algoritmo de verificación.

```
1 4 , 1 , 6 , 2
```

Listing 13: Representación de nuestro certificado en el archivo example2\_cert4.txt



```
[valeriamanjreez@MacBook-Air-de-Valeria src % java induced_path.ValidationAlgorithm ./../examples/graphs/example2.txt ./../examples/certificates/example2_cert4.txt

Reading the file ./../examples/graphs/example2.txt

Read graph:
0 1 0 1 1 0 0 0
1 0 1 0 0 0 0 1
0 1 0 1 0 0 1 0
1 0 1 0 0 1 0 0
1 0 0 0 0 1 0 1
0 0 0 1 1 0 1 0
0 0 1 0 0 1 0 1
0 1 0 0 1 0 1 0
Value of parameter K: 4

The given certificate 4,1,6,2 is NOT a valid solution.
valeriamanjreez@MacBook-Air-de-Valeria src % _
```

### 5.2.5. Certificado 5

Quinto certificado generado, con su representación en el esquema de codificación y el resultado obtenido al ejecutar el algoritmo de verificación.

1 7 , 2 , 5 , 4

Listing 14: Representación de nuestro certificado en el archivo example2\_cert5.txt



```
[valeriamanjreez@MacBook-Air-de-Valeria src % java induced_path.ValidationAlgorithm ./../examples/graphs/example2.txt ./../examples/certificates/example2_cert5.txt

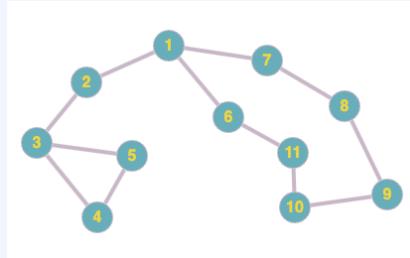
Reading the file ./../examples/graphs/example2.txt

Read graph:
0 1 0 1 1 0 0 0
1 0 1 0 0 0 0 1
0 1 0 1 0 0 1 0
1 0 1 0 0 1 0 0
1 0 0 0 0 1 0 1
0 0 0 1 1 0 1 0
0 0 1 0 0 1 0 1
0 1 0 0 1 0 1 0
Value of parameter K: 4

The given certificate 7,2,5,4 is NOT a valid solution.
valeriamanjreez@MacBook-Air-de-Valeria src % _
```

### 5.3. Ejemplar 3

Proponemos el siguiente ejemplar con su representación correspondiente en el esquema de codificación



```
1 5
2 1 , 2 , 3 , 4 , 5 , 6 , 7 , 8 , 9 , 10 , 11
3 1 , 2
4 1 , 7
5 1 , 6
6 2 , 3
7 3 , 4
8 3 , 5
9 4 , 5
10 6 , 11
11 7 , 8
12 8 , 9
13 9 , 10
14 10 , 11
```

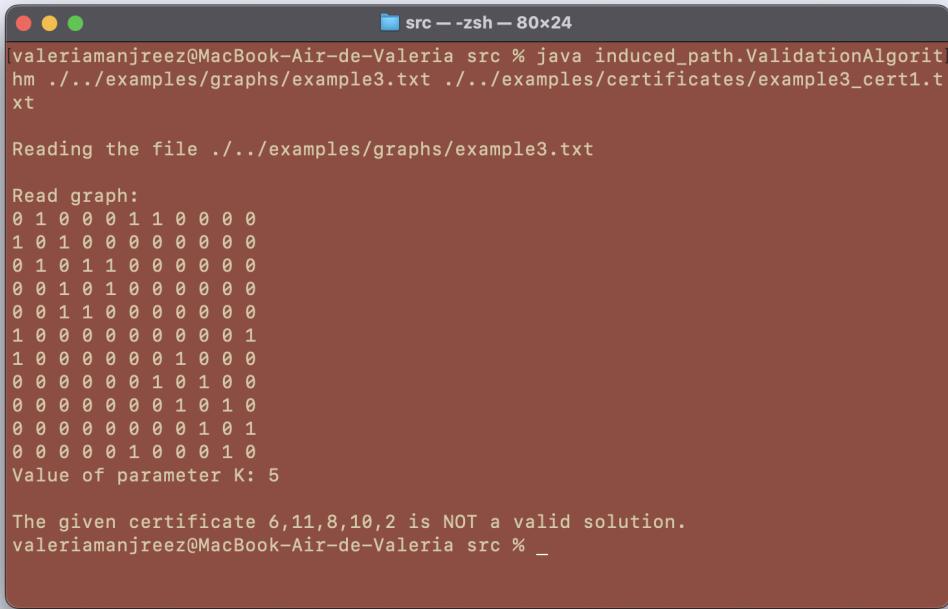
Listing 15: Representación de nuestra gráfica en el archivo example3.txt

#### 5.3.1. Certificado 1

Primer certificado generado, con su representación en el esquema de codificación y el resultado obtenido al ejecutar el algoritmo de verificación.

```
1 6 , 11 , 8 , 10 , 2
```

Listing 16: Representación de nuestro certificado en el archivo example3\_cert1.txt



```
[valeriamanjreez@MacBook-Air-de-Valeria src % java induced_path.ValidationAlgorithm ./../examples/graphs/example3.txt ./../examples/certificates/example3_cert1.txt

Reading the file ./../examples/graphs/example3.txt

Read graph:
0 1 0 0 0 1 1 0 0 0 0
1 0 1 0 0 0 0 0 0 0 0
0 1 0 1 1 0 0 0 0 0 0
0 0 1 0 1 0 0 0 0 0 0
0 0 1 1 0 0 0 0 0 0 0
1 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 1 0 0 0
0 0 0 0 0 0 1 0 1 0 0
0 0 0 0 0 0 0 1 0 1 0
0 0 0 0 0 0 0 0 1 0 1
0 0 0 0 0 1 0 0 0 1 0
Value of parameter K: 5

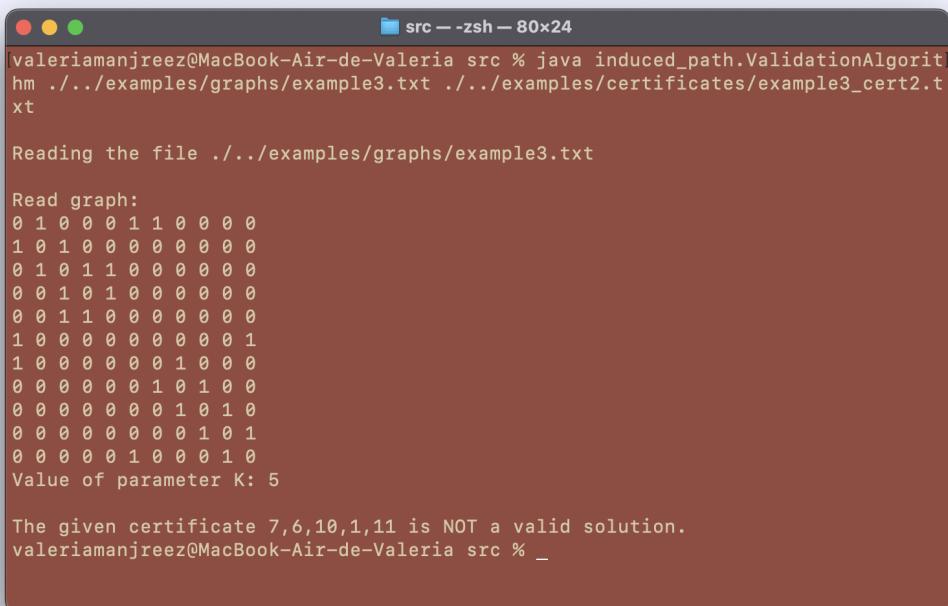
The given certificate 6,11,8,10,2 is NOT a valid solution.
valeriamanjreez@MacBook-Air-de-Valeria src % _
```

### 5.3.2. Certificado 2

Segundo certificado generado, con su representación en el esquema de codificación y el resultado obtenido al ejecutar el algoritmo de verificación.

```
1 7 ,6 ,10 ,1 ,11
```

Listing 17: Representación de nuestro certificado en el archivo example3\_cert2.txt



```
[valeriamanjreez@MacBook-Air-de-Valeria src % java induced_path.ValidationAlgorithm ./../examples/graphs/example3.txt ./../examples/certificates/example3_cert2.txt

Reading the file ./../examples/graphs/example3.txt

Read graph:
0 1 0 0 0 1 1 0 0 0 0
1 0 1 0 0 0 0 0 0 0 0
0 1 0 1 1 0 0 0 0 0 0
0 0 1 0 1 0 0 0 0 0 0
0 0 1 1 0 0 0 0 0 0 0
1 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 1 0 0 0
0 0 0 0 0 0 1 0 1 0 0
0 0 0 0 0 0 0 1 0 1 0
0 0 0 0 0 0 0 0 1 0 1
0 0 0 0 0 1 0 0 0 1 0
Value of parameter K: 5

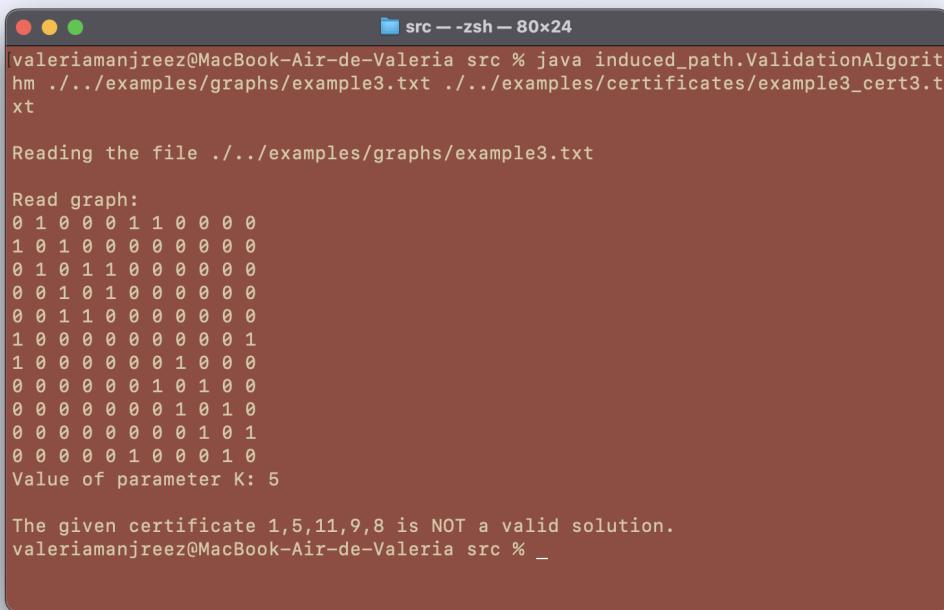
The given certificate 7,6,10,1,11 is NOT a valid solution.
valeriamanjreez@MacBook-Air-de-Valeria src % _
```

### 5.3.3. Certificado 3

Tercer certificado generado, con su representación en el esquema de codificación y el resultado obtenido al ejecutar el algoritmo de verificación.

```
1 1,5,11,9,8
```

Listing 18: Representación de nuestro certificado en el archivo example3\_cert3.txt



The screenshot shows a terminal window titled 'src -- zsh -- 80x24'. The command entered is 'java induced\_path.ValidationAlgorithm ./../examples/graphs/example3.txt ./../examples/certificates/example3\_cert3.txt'. The output shows the program reading the graph from 'example3.txt' and printing its adjacency matrix. It then reads the certificate 'example3\_cert3.txt' and prints its values. Finally, it outputs 'Value of parameter K: 5' and 'The given certificate 1,5,11,9,8 is NOT a valid solution.'

```
valeriamanjreez@MacBook-Air-de-Valeria src % java induced_path.ValidationAlgorithm ./../examples/graphs/example3.txt ./../examples/certificates/example3_cert3.txt

Reading the file ./../examples/graphs/example3.txt

Read graph:
0 1 0 0 0 1 1 0 0 0 0
1 0 1 0 0 0 0 0 0 0 0
0 1 0 1 1 0 0 0 0 0
0 0 1 0 1 0 0 0 0 0
0 0 1 1 0 0 0 0 0 0
1 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 1 0 0
0 0 0 0 0 0 1 0 1 0
0 0 0 0 0 0 0 1 0 1
0 0 0 0 0 0 0 1 0 1
0 0 0 0 0 1 0 0 0 1
Value of parameter K: 5

The given certificate 1,5,11,9,8 is NOT a valid solution.
valeriamanjreez@MacBook-Air-de-Valeria src % _
```

### 5.3.4. Certificado 4

Cuarto certificado generado, con su representación en el esquema de codificación y el resultado obtenido al ejecutar el algoritmo de verificación.

```
1 2,3,5,10,8
```

Listing 19: Representación de nuestro certificado en el archivo example3\_cert4.txt

```
[valeriamanjreez@MacBook-Air-de-Valeria src % java induced_path.ValidationAlgorithm ./../examples/graphs/example3.txt ./../examples/certificates/example3_cert4.txt

Reading the file ./../examples/graphs/example3.txt

Read graph:
0 1 0 0 0 1 1 0 0 0 0
1 0 1 0 0 0 0 0 0 0 0
0 1 0 1 1 0 0 0 0 0 0
0 0 1 0 1 0 0 0 0 0 0
0 0 1 1 0 0 0 0 0 0 0
1 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 1 0 0 0
0 0 0 0 0 0 1 0 1 0 0
0 0 0 0 0 0 0 1 0 1 0
0 0 0 0 0 0 0 1 0 1
0 0 0 0 0 1 0 0 0 1 0
Value of parameter K: 5

The given certificate 2,3,5,10,8 is NOT a valid solution.
valeriamanjreez@MacBook-Air-de-Valeria src % _
```

### 5.3.5. Certificado 5

Quinto certificado generado, con su representación en el esquema de codificación y el resultado obtenido al ejecutar el algoritmo de verificación.

1 11,6,1,2,3

Listing 20: Representación de nuestro certificado en el archivo example3\_cert5.txt

```
[valeriamanjreez@MacBook-Air-de-Valeria Computational-Complexity-main % cd p2
[valeriamanjreez@MacBook-Air-de-Valeria p2 % cd src
[valeriamanjreez@MacBook-Air-de-Valeria src % java induced_path.ValidationAlgorithm ./../examples/graphs/example3.txt ./../examples/certificates/example3_cert5.txt

Reading the file ./../examples/graphs/example3.txt

Read graph:
0 1 0 0 0 1 1 0 0 0 0
1 0 1 0 0 0 0 0 0 0 0
0 1 0 1 1 0 0 0 0 0 0
0 0 1 0 1 0 0 0 0 0 0
0 0 1 1 0 0 0 0 0 0 0
1 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 1 0 0 0
0 0 0 0 0 0 1 0 1 0 0
0 0 0 0 0 0 0 1 0 1 0
0 0 0 0 0 0 0 1 0 1
0 0 0 0 0 1 0 0 0 1 0
Value of parameter K: 5

The certificate 11,6,1,2,3 IS a valid solution.
valeriamanjreez@MacBook-Air-de-Valeria src % _
```

