UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE CIENCIAS

Práctica 1: esquemas de representación

ALUMNOS

Ulises Rodríguez García - 318042202
Zurisadai Uribe García - 318223197
Javier Alejandro Rivera Zavala - 311288876

PROFESOR

Oscar Hernández Constantino

AYUDANTE

Malinali Gónzalez Lara

ASIGNATURA

Complejidad computacional

Septiembre 19, 2023

Esquemas de Codificación

Considera el siguiente problema:

RUTA INDUCIDA

`EJEMPLAR:` Una gráfica G = (V, E) y 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 ?

1. Propón una versión de optimización, para el problema anterior.

   `Ejemplar`:
   Dada una gráfica G = (V, E), y un entero positivo $K \leq |V|$.
   `Pregunta:`
   ¿Cuál es el mayor valor para K, tal que la subgráfica inducida por un subconjunto $V' \subseteq V$ con $|V'| = K$ sea una ruta simple con K vértices?

2. Describir e implementar un esquema de codificación razonable para ejemplares del problema de **RUTA INDUCIDA** (en su versión de decisión). El esquema de codificación utilizado para gráficas, no puede ser una matriz.

   El sistema de codificación empleado, representa a un ejemplar del problema de decisión mediante una cadena de ceros y unos exclusivamente, dicha cadena sigue estas normas:

   - Los primeros m símbolos en la cadena, con $m > 0$, han de ser ceros y les sigue un 1 como separador. La cantidad m de tales ceros nos indica la longitud de la representación en binario para nuestro parámetro K.
   - Después de los $m + 1$ símbolos antes mencionados, ha de haber h ceros, con $h > 0$ y les sigue un 1 como separador. La cantidad h de tales ceros nos indica la longitud de la representación en binario de la cantidad de vértices de G.
   - Después del último 1 empleado como separador, los siguientes m símbolos representan al parámetro K en binario y los siguientes h símbolos que les siguen, constituyen la representación de $|V| = n$.
   - A todos los símbolos anteriores debe de seguirles una subcadena S de longitud $n * n$, cada subcadena de n símbolos en S representa las adyacencias del vértice que corresponde al número de subcadena (número dado por como aparecen en S) respecto de los vértices que corresponden a la posición de los símbolos en dicha subcadena.
   - NOTA: En esta representación se señala cada adyacencia entre vértices 2 veces, una por cada vértice que pertenece a la arista correspondiente, sin embargo, en este caso eso no representa una gráfica con dirección, la salida de nuestro algoritmo será entonces una matriz de adyacencias simple y en binario donde se señala qué vértice es adyacente con que otro en cada fila, pero aunque se señalen las adyacencias 2 veces, eso no significa que haya doble dirección. Se optó por esta representación por simplicidad, y por qué es más fácil extender(ó reducir) este algoritmo para representar gráficas dirigidas o con lazos con una simple verificación extra o bien quitando una verificación (se deja en el código la verificación correspondiente comentado, en caso de que se quiera probar).

   Esta representación es razonable toda vez que en primer lugar, cualquier número entero positivo (y otros más, desde luego) es representable bajo el sistema binario de forma finita y única, por lo tanto K y $n = |V|$ son representables dentro del sector de la cadena que les corresponde, además, es obvio que sus longitudes m y h respectivamente, se pueden representar con la acumulación de m y h ceros pues se trata de enteros positivos. Apuntamos que además, los unos empleados como separadores nos permiten distinguir el fin de las subcadenas respectivas que definen a m y h, además de que estos últimos nos permiten distinguir claramente la representación de K y n.
   Por último, la representación de G es factible mediante la subcadena S de longitud $n * n$ ya que los unos nos indican con que vértices si es adyacente el vértice i (i el número de la subcadena de S de tamaño n) y los ceros con cuales no. Puesto que tomamos a S de longitud $n * n$ se cubrieron todas las posibles aristas.

3. Describir e implementar un algoritmo para transformar el esquema de codificación propuesto, a uno que utilice codificación de gráficas como matrices.
   El algoritmo propuesto se comporta como sigue:

   - Se recibe el nombre del archivo que contiene la cadena de representación con su extensión, se abre el archivo y se copia su contenido en una cadena y por último se cierra el archivo.
   - Se recorre la cadena obtenida en el paso anterior desde el inicio, se verifica que haya $m > 0$ ceros seguidos de un 1 y después de ellos, $h > 0$ ceros seguidos de un uno.
   - Verificamos que haya al menos $2 * (h + m + 1)$ símbolos en la cadena para que no tengan lugar errores al tratar de conseguir los valores de K y n.
   - Si se pasó la verificación anterior, tomamos los siguientes m símbolos desde donde nos quedamos en el segundo paso, esto para obtener el valor de K al convertir tales dígitos a decimal. Para obtener el valor de n, tomamos los h símbolos que siguen a los m anteriores y procedemos igual. Importante cerciorarnos de que tales dígitos son ceros y unos nada más, esto mientras los tomamos.
   - Nos cercioramos de que el valor de K sea menor o igual a n.
   - Si pasamos la prueba anterior, entonces verificamos que el resto de la cadena después de donde nos quedamos hace 2 pasos, tenga longitud $n * n$ para poder representar a G.
   - Si se cumplió la condición anterior, tomamos grupos de n símbolos en n símbolos del resto de la cadena, el primer grupo de la cadena será la primer fila de la matriz, el segundo grupo será la segunda fila y así sucesivamente. Mientras los leemos, verificamos que dichos grupos estén constituidos únicamente por ceros y unos y contamos cuantos unos hay en total en la subcadena S. También iremos contando cuantas aristas hay, llevando un conteo de la cantidad de unos.
   - Tomamos el valor de K en decimal, el número de aristas (el techo de la cantidad de unos contados en S sobre 2), el número de vértices (en decimal) e imprimimos en pantalla la descripción del ejemplar, finalmente abrimos un archivo para la salida (se crea si no existe) y escribimos en su primer línea la representación de K en binario y después de ello, las lineas siguientes tendrán a la matriz de adyacencias resultante, fila por fila.
   - NOTA: Añadimos una verificación especial que no necesariamente debe emplearse, esto para asegurarnos de que no se aceptan gráficas con lazos. Para ello se verifica que el símbolo en la posición i dentro de la subcadena i de S (mientras se va formando la matriz), sea siempre cero. En una versión que maneje un tipo más amplio de gráficas, tal verificación puede omitirse.

Analicemos la complejidad del presente algoritmo respecto al tiempo:

Consideremos a x como la longitud de la cadena que representa al ejemplar, la lectura del archivo que la contiene se considera como $O(x)$, pues depende del tamaño del texto a leer y las operaciones de apertura y cierre del archivo suelen ser despreciables. Luego de eso tenemos la transformación de la cadena leída, hay operaciones constantes como la asignación de variables y algunas comprobaciones como aquellas de la longitud de K y n, que K no sea mayor que n, etc. aunque las mismas serán despreciables conforme lo que veremos más adelante. Como x ha de ser igual a $2m + 2h + 2 + m \cdot m$ según lo explicado en el punto 2 del reporte, podemos decir que esta parte del algoritmo tiene orden $O(m + h + m \cdot m)$dada al definición del orden de complejidad o-grande y sus propiedades.

El primer bucle while busca la longitud de la secuencia de ceros al principio de la cadena binaria. Aquí un fragmento del código correspondiente:

```
while cadenaBinaria[i] == '0':
    longitudK += 1
    i += 1
```

En el peor caso, este bucle recorrerá los primeros $h + 1$ símbolos, luego el siguiente bucle recorre los subsecuentes $m+1$ símbolos, posterior a ello tomaremos otros h símbolos y otros m símbolos para conseguir los valores de K y n para convertirlos de binario a decimal, todo ello con complejidad $O(m + h)$ pues la conversión desde binario depende de las longitudes de K y n. Hasta el momento, la complejidad del algoritmo es $O(2m + 2h + 2)$ que es equivalente a $O(m + h)$ dada la definición del orden de complejidad o-grande y sus propiedades.

Después de todo lo anterior, recorremos la subcadena S que contiene la representación de G, durante tal recorrido efectuamos una comprobación de pertenencia de los símbolos de S a '01' y construimos una matriz de longitud $m \cdot m$. Veamos un fragmento del código que ilustra este proceso:

```
for x in range(n):
    for y in range(n):
        if cadenaBinaria[i] not in '01':
            print("La cadena ingresada debe consistir únicamente de unos y ceros\n")
            return None
        ...
        i += 1
```

Podemos decir que esa parte del algoritmo tiene una complejidad de $O(m \cdot 2m)$, pues la comprobación se realiza dentro de un bucle for interno sobre m símbolos y requiere de recorrer la lista '01' de longitud 2, ese recorrido se efectúa m veces, por ello $O(m \cdot 2m)$ que es equivalente a $O(m \cdot m)$. Hay otras comprobaciones dentro del bucle pero son despreciables al ser constantes.

Para terminar, la parte del algoritmo que realiza la escritura en el archivo de salida, depende directamente del tamaño de la representación del ejemplar, h símbolos para el valor de K y $m \cdot m$ para la matriz, las demás operaciones son despreciables pues se efectúan en tiempo constante. La complejidad de esta parte del algoritmo es $O(h + m \cdot m)$.

Todo lo anterior nos deja ver que la complejidad general del algoritmo corresponde a $O(h + m + m \cdot m)$, dada la definición y las propiedades del orden de complejidad o-grande, por último, dado que $x = 2h + 2m + 2 + m \cdot m$ podemos afirmar que la complejidad de nuestro algoritmo respecto al tiempo es de orden $O(x)$.

4. **Ejemplares**

- Ejemplar con $|V| \geq 5$, $|E| \geq 10$, $K = 3$ y con respuesta SI

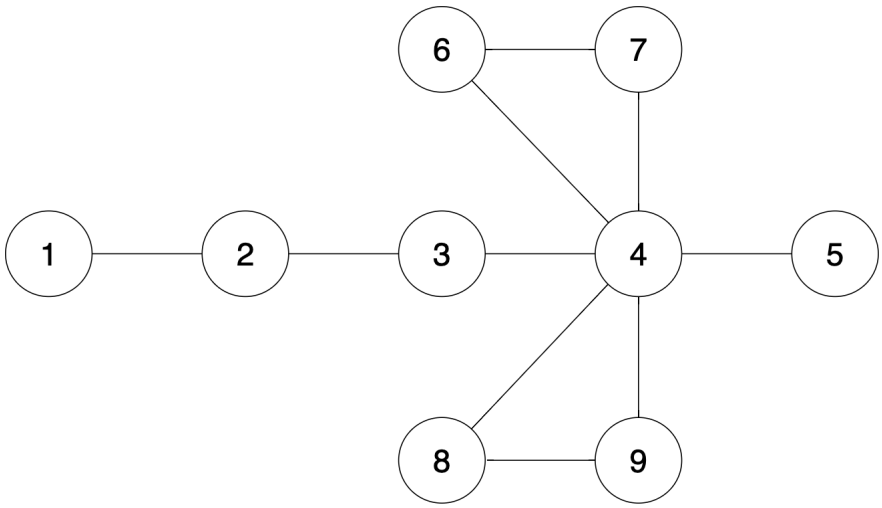
  Cadena que codifica a cada ejemplar

  La cadena que codifica el ejemplar con un grafo que tiene 9 vértices y 11 aristas, con K=3, se presenta a continuación:

  00100001111001010000000101000000000100000001011111000100001000100100000101000000100001000110010

  Representación Visual del Ejemplar (sin codificar)

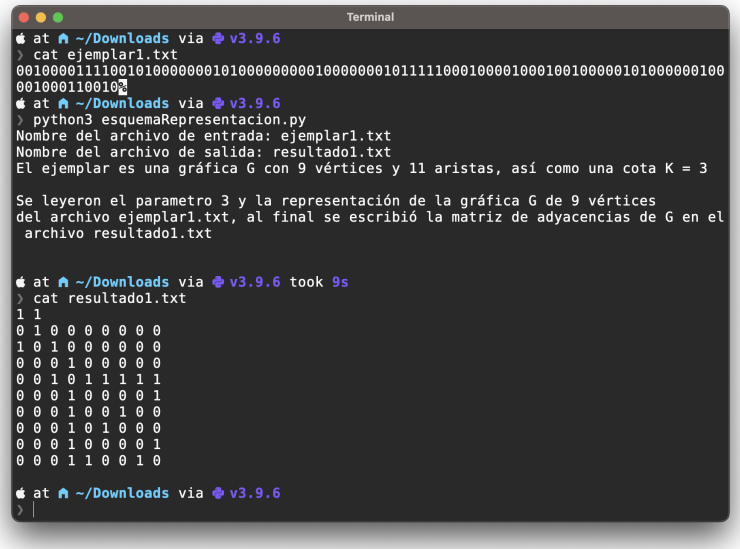
  A continuación, se muestra una representación visual del grafo sin codificar:



  La representación visual del grafo se muestra arriba. Este grafo cumple con las condiciones de tener al menos 5 vértices y al menos 10 aristas.

Resultado de la Ejecución del Programa

El programa fue ejecutado con éxito y generó el siguiente resultado:



Esta imagen muestra el resultado obtenido al procesar el ejemplar codificado, confirmando que se leyeron los parámetros correctamente y se generó la matriz de adyacencias del grafo G en el archivo de salida. Es importante destacar que el subconjunto de vértices que satisface la condición es el conjunto $\{1, 2, 3\}$, entre otros.

- Ejemplar con $|V| \geq 5$, $|E| \geq 10$, $K = 2$ y con respuesta NO

  Para este caso, encontrar un ejemplar que cumpla con las condiciones dadas es imposible debido a la restricción sobre el parámetro K. Un grafo con $|E| \geq 10$ y $|V| \geq 5$ tendría, en promedio, un grado de nodos considerablemente alto y más de una arista, lo que haría inmediato encontrar una ruta inducida de tamaño 1 (i.e., una arista entre 2 vértices adyacentes).

  Para ilustrar esto, consideremos un grafo sencillo con 5 vértices donde cada vértice está conectado a todos los demás, formando un grafo completo K_5 que tiene 20 aristas (mayor o igual que 10). En este grafo, es posible encontrar una ruta inducida de tamaño 1 con facilidad, e incluso en cualquier subgrafo del mismo con entre 10 y 20 aristas; tomamos cualquier arista del gráfico y con ello la condición ya se cumple, de modo que podemos responder SI al problema de decisión.

  Por lo tanto, para $K = 2$, sería prácticamente imposible obtener una respuesta NO a menos que estemos considerando un grafo muy específico y altamente restrictivo, el cual estaría lejos de ser un caso general y tendría que ser diseñado especialmente para no contener ninguna ruta inducida de tamaño 1 con 2 vértices.

  Debido a estas razones, podemos concluir que un ejemplar que cumpla con todas estas condiciones no puede existir.

- Ejemplar con $|V| \geq 6$, $|E| \geq 10$, $K = 3$ y con respuesta SI

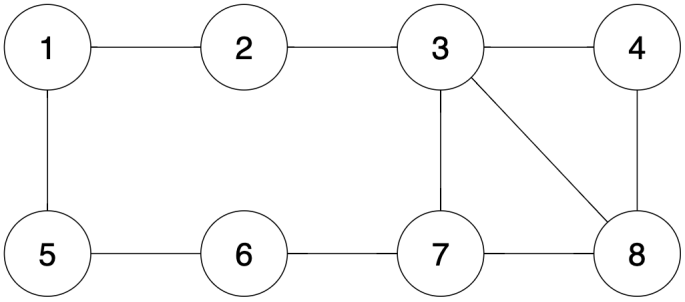
  Cadena que codifica a cada ejemplar

  La representación codificada del ejemplar que describe un grafo con 8 vértices y 10 aristas, con K=3, se presenta a continuación:

  0010000111100001001000101000000101001100100001100001000000010100010010100110010

Representación Visual del Ejemplar (sin codificar)

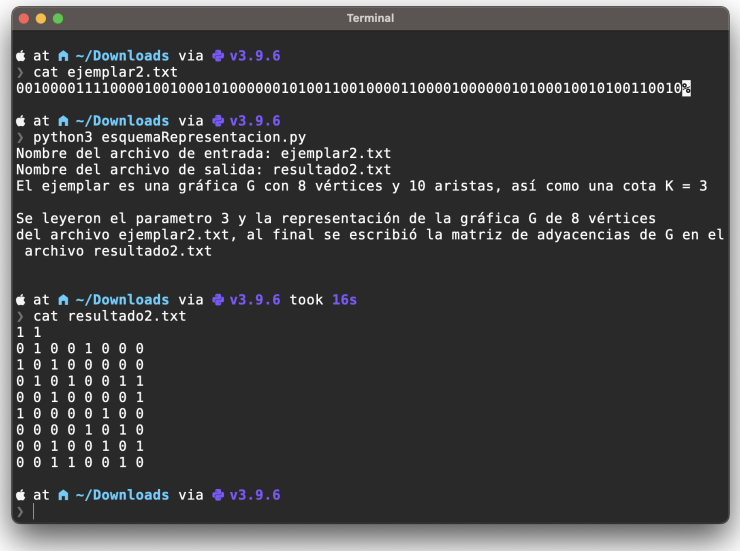
A continuación, se muestra una representación visual del grafo sin codificar:



La representación visual del grafo se muestra arriba. Este grafo cumple con las condiciones de tener al menos 6 vértices y al menos 10 aristas.

Resultado de la Ejecución del Programa

El programa se ejecutó con éxito y generó el siguiente resultado:



Esta imagen muestra el resultado obtenido al procesar el ejemplar codificado, confirmando que se leyeron los parámetros correctamente y se generó la matriz de adyacencias del grafo G en el archivo de salida. Es importante destacar que el subconjunto de vértices que satisface la condición es el conjunto $\{1, 2, 3\}$, entre otros.

Referencias

- Discrete Math: Each positive integer has a unique binary representation, part 1, Japhet Wood, abril de 2020, consultado en septiembre de 2023
- Discrete Math: Each positive integer has a unique binary representation, part 1, Japhet Wood, abril de 2020, consultado en septiembre de 2023
- Graphs and digraphs
  Autores: Gary Chartrand, Linda Lezniak y Ping Zhang
  Editorial: Taylor & Francis group - CRC press
  Año de impresión: 2016
  Idioma: Inglés.
  Se consultaron los capítulos 1 y 2: *Introduction* en el apartado de *Induced graphs* y *Conected graphs and distance* en el apartado de *The adjacency matrix of a graph*.
- Notas del curso de análisis de algoritmos 2023-2
  Presentación 2 - Notación asintótica
  Autor: Pedro Ulises Cervantes González
  Link: