Estrella de Steiner en grafos
Prueba de la NP-completitud

Sofía Almeida Bruno,
Pablo Álvarez Cabrera
Universidad de Granada

7 de junio de 2019

Descripción del problema

El problema de la estrella de Steiner en grafos, al que nos referiremos como ES, se define de la siguiente forma:

Dado un grafo $G = (V, E)$, un subconjunto $R \subseteq V$ y un entero positivo $K \leq V - 1$, ¿existe un subárbol de G que contiene todos los vértices de R y que no contiene más de K arcos?

NP-completitud

En primer lugar veremos que está en NP. Después, definiremos una reducción del problema 3-SET al ES para probar que es NP-completo.

ES está en NP

El algoritmo sería el siguiente:

- Seleccionar de forma no determinista un subgrafo, esto es, elegir un subconjunto de nodos y otro de aristas.
- Comprobar que el subgrafo es un árbol. Esto se puede hacer viendo que no contiene ciclos y que es conexo, lo cual se puede hacer en tiempo polinómico.

- Verificar que contiene a los nodos de R.
- Ver que el subgrafo no contiene más de K arcos.

Si existe una estrella de Steiner en el grafo, el algoritmo anterior devuelve "SÍ" para un cierto subgrafo. En caso de no existir, el algoritmo devuelve siempre "NO".

Reducción 3-SET $\propto$ ES

Para la reducción vamos a considerar el problema del cubrimiento por conjuntos de tamaño tres (3-SET).

Una instancia de este problema vendrá dada por:

- Un conjunto finito $X = \{x_1, \ldots, x_{3q}\}$.
- Un subconjunto $C = \{C_1, \ldots, C_n\}$ de subconjuntos de X de 3 elementos.

La reducción se hará de la siguiente forma:

Queremos construir una entrada $(G = (V, E), R, K)$ para el problema ES a partir de una entrada (X, C) para el problema 3-SET.

- En el conjunto de vértices habrá un vértice por cada elemento de X, otro por cada elemento de C y un vértice al que llamaremos v.

$$V = \{x_1, \ldots, x_{3q}\} \cup \{c_1, \ldots, c_n\} \cup \{v\}$$

- Habrá una arista uniendo v con cada uno de los c_i y otra uniendo cada c_i con los elementos que contiene.

$$E = \{vc_1, \ldots, vc_n\} \cup \big(\bigcup_{x_j \in C_i} \{c_i x_j\} \big)$$

- Consideremos $R = \{v, x_1, \ldots, x_{3q}\}$.
- Por último, $K = 4q$.

Veamos que si 3-SET tiene solución positiva, entonces ES también la tiene.

Sea C' un cubrimiento por conjuntos de X. Como $|X| = 3q$, C' tendrá $3q/3 = q$ subconjuntos. Supongamos que estos son $c_1, \ldots, c_q$. Consideramos el subgrafo dado por (V, E), donde $V = \{v, c_1, \ldots, c_q, x_1, \ldots, x_{3q}\}$ y $E = \{vc_1, \ldots, vc_q\} \cup (\bigcup_{x_j \in c_i} \{c_i x_j\})$ con $i = 1, \ldots, q$. Esta es la estrella de Steiner que resuelve el problema, ya que se trata de un árbol que contiene a los nodos de R y el número de aristas es $q + 3q = 4q = k$.

Finalizaremos viendo que si ES tiene una solución positiva, entonces 3-SET también la tiene.

Supongamos que existe un grafo de Steiner G con al menos $4q$ lados. Como G es un árbol no puede tener más de $4q + 1$ vértices. G contiene a v y $x_1, \ldots, x_{3q}$, luego no puede incluir más de q vértices c_i. Ahora observamos que para conectar con todos los nodos x_i debe haber al menos $4q + 1$ vértices, luego G contiene exactamente $4q$ aristas y q nodos c_i $(c_1, \ldots, c_q)$. La solución al problema 3-SET vendrá dada por los subconjuntos $C_1, \ldots, C_q$ asociados a los vértices $c_1, \ldots, c_q$.

Espacio logarítmico

Veamos que esta reducción se puede hacer en espacio logarítmico. Supongamos que la entrada tiene el siguiente formato:

$$x_1 \ldots x_{3q} \# c_{11} c_{12} c_{13} \ldots c_{n1} c_{n2} c_{n3}$$

La salida estará formada por el listado de vértices seguido por el listado de aristas, el valor de R y el valor de K separados por #.

Para los vértices, basta con escribir v, ir copiando en la salida los elementos de la entrada hasta llegar a # y a partir de ahí añadir c_i por cada tres elementos. Por tanto, será necesario un contador $i = 1, \ldots, n$ y otro $j = 1, 2, 3$ que se pueden almacenar en espacio logarítmico (en binario).

Para las aristas, volvemos hasta el símbolo # en la entrada y vamos avanzando escribiendo las aristas $vc_i,\ c_i c_{ij}, \quad i = 1, \ldots, n \quad j = 1, \ldots, 3$ (usamos los mismos contadores).

Para R, volvemos hacia el comienzo de la entrada y añadimos v seguido de $x_1 \ldots x_{3q}$. Guardamos en otro contador el valor $3q$ (también es espacio logarítmico).

Por último, escribimos $K = 4q$. Para conseguir este valor hay que dividir el número de elementos $3q$ entre 3, para hacerlo en espacio logarítmico basta ir contando los valores de la entrada antes de llegar a # de tres en tres. Este número q se multiplica por 4, como es en binario solo hay que añadir dos 0 al final, luego es espacio logarítmico.

El funcionamiento del algoritmo queda ilustrado en el código que hemos implementado en Python:

```
entrada = input() # aquí es donde está la entrada
salida = ""       # aquí es donde está la salida

# Reducción:
# Contadores:
p = 0  # puntero de lectura
a = 0  # posición de '#'
i = 1  # subconjunto
j = 1  # elemento del subconjunto
k = 0  # número de elementos

# Vértices
salida += "v␣"
while(entrada[p] != '#'):
    salida += entrada[p] + "␣"
    p+=1
a = p
p+=1
while(p < len(entrada)):
    salida += "c" + str(i) + "␣"
    for j in range(1,4):
        p+=1
    i+=1

# Aristas
salida += "␣#␣"
p = a+1
i = 1
while(p < len(entrada)-1):
    salida += "vc" + str(i) + "␣"
    for j in range(1,4):
        salida += "c" + str(i) + entrada[p] + "␣"
        p+=1
    i+=1

# R
salida += "␣#␣v␣"
p = 0
while(entrada[p] != '#'):
```

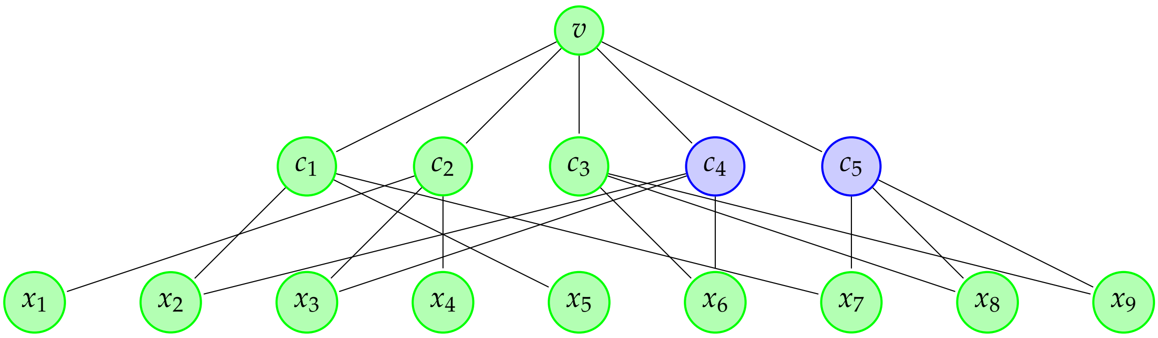
```
        salida += entrada[p] + "␣"
        p+=1
        k+=1

    # K
    salida += "␣#␣"
    salida += str(int((k/3)) * 4)

    print(salida)
```

Ejemplo

Veamos cómo funciona la reducción en un caso concreto.

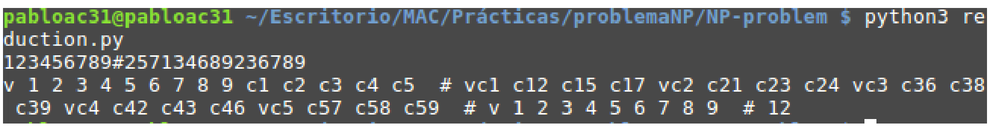
Partimos del problema 3-SET dado por$X = \{x_1, \ldots, x_q\}$ y $C = \{C_1, \ldots, C_5\}$, donde $C_1 = \{x_2, x_5, x_7\}, C_2 = \{x_1, x_3, x_4\}, C_3 = \{x_6, x_8, x_9\}, C_4 = \{x_2, x_3, x_6\}$, $C_5 = \{x_7, x_8, x_9\}\}$. Este problema admite como solución $C' = \{C_1, C_2, C_3\}$.

La entrada de la reducción serían los conjuntos X, C y la salida, el grafo mostrado a continuación, $R = \{v, x_1, \ldots, x_9\}$, $K = 12$.



Una solución a este problema es el subárbol formado por los nodos coloreados en verde y las aristas que los unen.

Ejecutamos nuestro programa pasándole como entrada este mismo ejemplo y obtenemos:


pabloac31@pabloac31 ~/Escritorio/MAC/Prácticas/problemaNP/NP-problem $ python3 reduction.py
123456789#257134689236789
v 1 2 3 4 5 6 7 8 9 c1 c2 c3 c4 c5 # vc1 c12 c15 c17 vc2 c21 c23 c24 vc3 c36 c38 c39 vc4 c42 c43 c46 vc5 c57 c58 c59 # v 1 2 3 4 5 6 7 8 9 # 12