Problema K-minimum spanning tree

Roberto Juan Cayro Cuadros, Gabriel Alexander Valdivia Medina, Giulia Alexa Naval Fernández, Rodrigo Alonso Torres Sotomayor

Universidad Católica San Pablo

Resumen

---

1. Introducción al problema

El k-MST o k-minnimum spanning tree problem, árbol de expansión de péso mínimo k en español es un problema computacional que pide un árbol de mínimo costo con exactamente k vértices que forme un subgrafo del grafo original. [**?**]

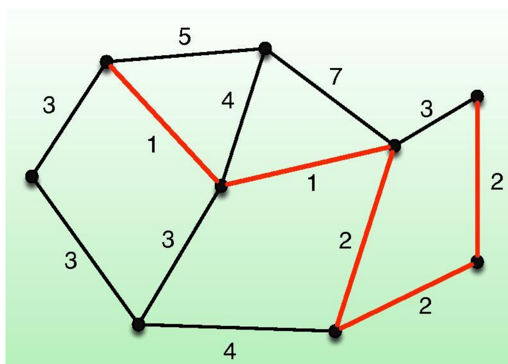

Figura 1: 6-MST del grafo G. Fuente: Wikipedia Commons

2. Demostración NP-completo

No es posible suponer la naturaleza del problema, y establecer que es NP-hard o NP-completo, sin la evidencia correspondiente, para probrarlo este debe pertenecer a NP, ademas que un problema NP-completo pueda reducirse al mismo.

2.1. Demostrar que k-MST $\in$ NP

k-MST (G,k)

1. x $\leftarrow$ arbol.
2. **for** t $\leftarrow$ 0, **to** k
3.   **do** u $\leftarrow$ESCOGER(G)
4.   **if** u **is not in** x
5.   **do** x . add(u)

2.2. Transformación NP-completo α k-MST

2.2.1. Steiner problem

Es un problema NP-completo de los 21 problemas de Karp, usado en problemas de optimización y mayormente enfocado en estructuras de grafos aunque tambien visto en aplicaciones de modelación de redes con más de 2 terminales. El problema consiste en que, dado un grafo no-dirigido de aristas con peso, generar un arbol dado un *Sub-set* de vertices los cuales formarán este arbol. Además, pueden añadirse nuevos vertices para lograr las conexiones entre estos, llamados *Steiner-vertices.*

La decisión asociada al problema será averiguar si existe un árbol que una todos los vértices de un *sub-set* R, usando máximo M aristas. Los vertices deberán ser exactamente los dados en el Sub-set, pero podrán ser añadidos los Steiner-vertices fueran necesarios. Esta decisión es conocida por ser del grupo de los NP-completos. La principal diferencia con el k-MST es que aquí recibimos un conjunto específico de vectores para conformar nuestro árbol, pudiendo usar vértices fuera de la relación para conectarlos. El k-MST no recibe esta relación, sólo el número de vértices exactos que necesita.

2.3. Entradas y salidas

Steiner-problem

Entrada:

- Grafo no-dirigido G con aristas de peso.
- Sub-set de vertices R.
- Número M.

Salida:

- Arbol de menor peso con los vertices de S y los Steiner-vertices si fueran necesarios.

k-MST

Entrada:

- Grafo no-dirigido G con aristas de peso.
- Número k de vértices.

Salida:

- Arbol de menor peso con k-vertices y k-1 aristas.

2.3.1. Transformación

Dada la entrada G para Steiner, se puede tomar el mismo grafo para k-MST, puesto que tiene las aristas pesadas y un número determinado de vertices. De esta forma aseguramos la transformación y no afectara la salida porque siempre busaremos el arbol de menor peso , se usará el tamaño del sub-set de vertices siendo este igual a k.

Por consiguiente logramos reducir el problema de Steiner tree a k-MST, al tranformar su entrada en los parámetros de k-MST. Pero resta confirmar que la solución de k-MST también podría solucionar a Steiner-tree. La salida de k-mst será un arbol con el menor peso posible, sin embargo, debemos tener en cuenta que el número k denota la cantidad de vertices necesarios más no que vertices, por lo tanto el numero total de permutaciones de los vertices de G serán combinaciones de el número total de vertices(n) en k vertices:

$$\text{Total de permutaciones (Tn)} = \binom{n}{k} = \frac{n!}{k!(n-k)!}.$$

Esto es importante porque sabiendo que k es el size of sub-set (S) , y hablamos de todas las permutaciones de ese tamaño, podemos concluir que el S debe estar incluido en Tn. Por ello:

$$S \subseteq Tn$$

Sin embargo, calcular todas las permutaciones de una cantidad n de elementos es un proceso con una complejidad $O(!n)$, que no entra dentro de complejidad polinomial. Es necesario otro tratamiento para que el k-MST opere con los vértices que el algoritmo Steiner pide. Otra idea es añadir un árbol con aristas de peso 0 en cada vértice que pertenezca a R, y transformar k como $k = |R|(X+1)$, siendo X la cantidad de vértices que tendrán cada uno de estos árboles, denotado como $X = |V(G)| - |R|$ De esta forma, el k-MST utilizará elos vértices de R vértices sí o sí como parte de su solución.

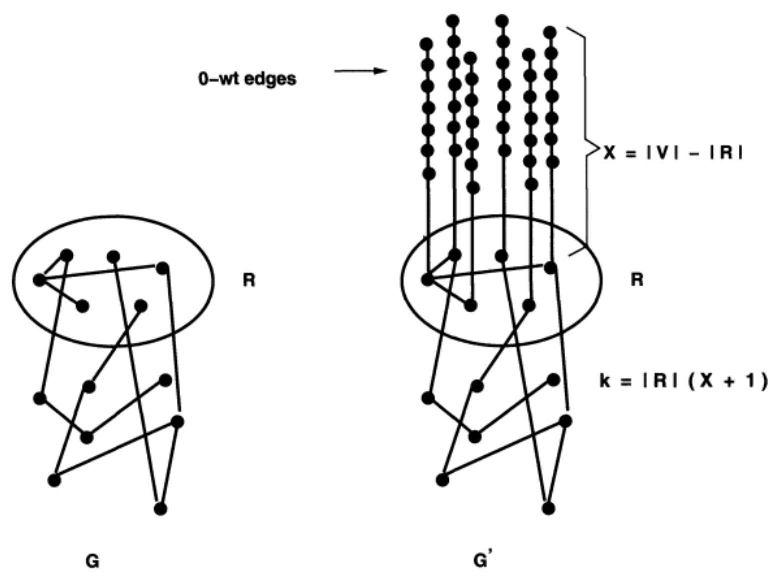

Figura 2: Transformación de la entrada de Steiner a entrda de k-MST. Fuente: www.contrib.andrew.cmu.edu

En este nuevo grafo G', las aristas que unen los nuevos vértices de X tendrán un peso de 0, las aristas correspondientes a las aristas originales de G tendrán un peso de 1, y el resto de pares del grafo tendrán un peso de ∞. De este modo, el algoritmo del k-MST encontrará el árbol de menor peso con el parámetro k en G', y verificará si es de igual o menor peso que M, satisfaciendo el requisito de las M aristas debido a que estas tendrán peso 1.

3. Algoritmo de fuerza bruta

4. Algoritmo aproximado

Bibliography

Referencias

[1] https://en.wikipedia.org/wiki/K-minimum_spanning_tree