Optimización de Redes Eléctricas mediante uso de Grafos

Lopera Torres, Samuel
sloperat@eafit.edu.co

Lopez Correa, Santiago
slopezc12@eafit.edu.co

Castrillon Correa, Juan José
jjcastric@eafit.edu.co

Optimización I

Departamento de Ciencias e Ingeniería
Universidad EAFIT
Colombia
Mayo 2023

1 Introducción

En las últimas décadas, el crecimiento exponencial de la demanda de energía eléctrica ha planteado desafíos significativos para las compañías de servicios públicos y los operadores de redes eléctricas. La necesidad de suministrar energía de manera eficiente y confiable ha llevado al desarrollo de diversas técnicas y enfoques para optimizar el funcionamiento de estas redes complejas.

En este contexto, los grafos se presentan como una herramienta poderosa para abordar los desafíos de optimización en las redes eléctricas. Un grafo es una estructura matemática que permite representar y modelar las interconexiones entre los elementos de una red, donde los nodos representan componentes eléctricos (generadores, transformadores, cargas, etc.) y las aristas representan las líneas de transmisión.

El uso de grafos en la optimización de redes eléctricas permite analizar y resolver problemas complejos relacionados con el flujo de energía, la distribución de cargas y la minimización de pérdidas, entre otros. Al aplicar algoritmos y técnicas de teoría de grafos, es posible encontrar soluciones óptimas o aproximadas para problemas de dimensionamiento, enrutamiento y control de redes eléctricas.

2 Planteamiento del Problema

El problema se plantea desde la minimización de costos para la construcción de un tendido eléctrico determinado; se puede proponer mediante el siguiente modelo:

$$\min C : \sum_{j=1}^{n} \sum_{i=1}^{n} c_{ij} x_{ij}$$

$$S.A.$$

$$\sum_{i=1}^{n} x_{ij} - \sum_{i=1}^{n} x_{ji} = 0$$

$$\sum_{i=1}^{n} x_{ij} = n - 1$$

$$x_{ij} \in \{0, 1\} \quad \forall i, j$$

Pero, tomando en cuenta los datos utilizados por el equipo, que plantean el

costo en función de la longitud del cable utilizado, se puede replantear como la distancia mínima entre todos los nodos:

$$\min L : \sum_{j=1}^{n} \sum_{i=1}^{n} l_{ij} x_{ij}$$

$$S.A.$$

$$\sum_{i=1}^{n} x_{ij} - \sum_{i=1}^{n} x_{ji} = 0$$

$$\sum_{i=1}^{n} x_{ij} = n - 1$$

$$x_{ij} \in \{0, 1\} \quad \forall i, j$$

donde x_{ij} representa si la conexión de un nodo i a un nodo j esta presente y l_{ij} representa la distancia del nodo i al nodo j. Además con n representando el número de vértices.

3 Metodología

Para la solución de este problema, se tomo la opción de implementar algunos algoritmos para la obtención de un árbol de expansión mínima (o MST, por sus siglas en ingles), el cual proporciona la distancia mínima entre todos los nodos. En particular, se implementaron el algoritmo de Prim y el algoritmo de Kruskal, para obtener el MST del grafo que se le ingresa.

Estos algoritmos se implementaron en python 3.11

3.1 Algoritmo de Prim

El algoritmo de Prim es un algoritmo Greedy que permite hallar el MST empezando desde un nodo arbitrario s y expandiéndolo hacia afuera [1]

Mediante un proceso iterativo, el algoritmo se puede expresar en 3 pasos:

1. Se elige arbitrariamente un vértice v_1 y se determinan dos variables $V = \{v_1\}$ y $E = \{\}$, donde V Representa el conjunto de todos los vértices visitados, y E el conjunto de todas las aristas en el MST

2. Se elige el vértice mas cercano al actual (el ultimo en el conjunto de visitados) para añadirlo a V, y se añade la arista que conecta el vértice anterior con el vértice mas cercano (v_i, v_j) al conjunto E, siempre y cuando (v_i, v_j) no forme ciclos con ninguno de los elementos de E

3. Se repite el paso 2 hasta que $|E| = n - 1$ (donde n representa la cantidad de vértices del grafo), siendo esta la condición de parada del algoritmo

Al finalizar el algoritmo E Contiene las aristas que conectan los vértices pertenecientes al MST del grafo [2]

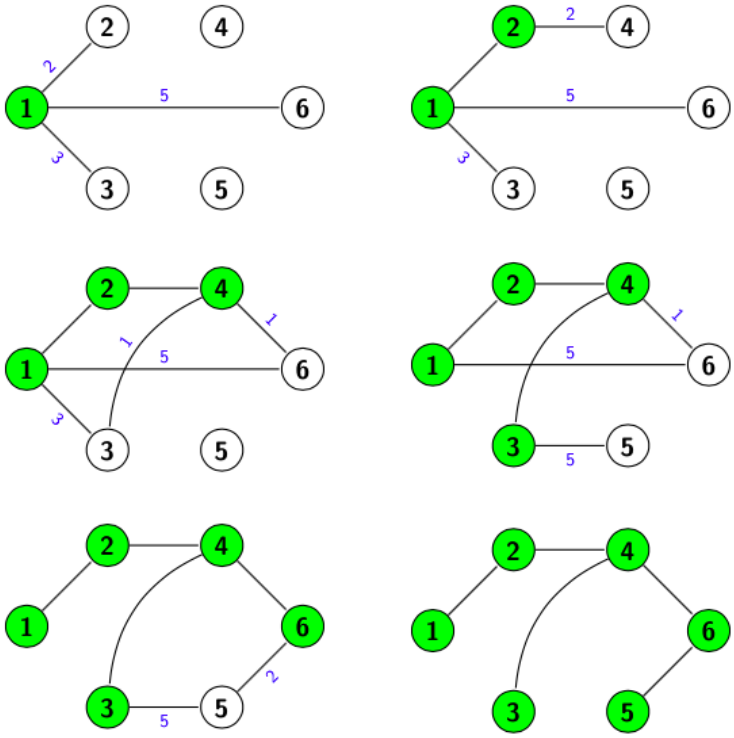

Figure 1: Ejemplo del algoritmo de Prim [3]

3.2 Algoritmo de Kruskal

Al igual que el algoritmo de Prim, el algoritmo de kruskal es un algoritmo Greedy que permite hallar el MST de un grafo ponderado y no dirigido [4].

El procedimiento de este es el siguiente:

1. Se ordenan de forma ascendente los pesos de las aristas del grafo y se determina la variable $E = \{\}$
2. Se marca la arista con el menor peso como la arista inicial del MST y se añade a E
3. De las aristas restantes, se selecciona la de menor peso
4. Se repite el páso 3, añadiendo aristas siempre y cuando no formen ciclos en E, hasta que $|E| = n - 1$ dandose por terminado el algoritmo

[4]

4 Resultados

Utilizamos varios algoritmos para encontrar árboles de expansión mínima. Aplicamos estos algoritmos en las redes propuestas por Jossely Richard Aquino Dominguez y Vladimur Giovanni Rodriguez Sabino en su artículo para UNASAM [5]. Específicamente, utilizamos los algoritmos de Prim y Kruskal para encontrar los árboles de expansión mínima de estas redes. Ambos algoritmos nos dieron el mismo resultado.

En el artículo, los autores también aplicaron el algoritmo de Prim para encontrar un árbol de expansión mínima y optimizar la red. Sin embargo, nuestros resultados difirieron un poco de los obtenidos por los autores.

4.1 Primera Red

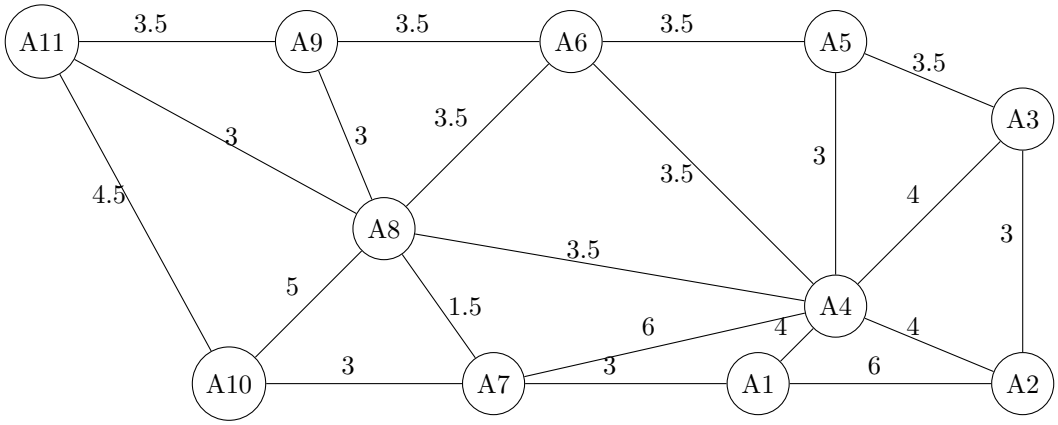

Figure 2: Primera Red [5]

En el articulo, aplicar el algoritmo de prim resulta en el siguiente MST

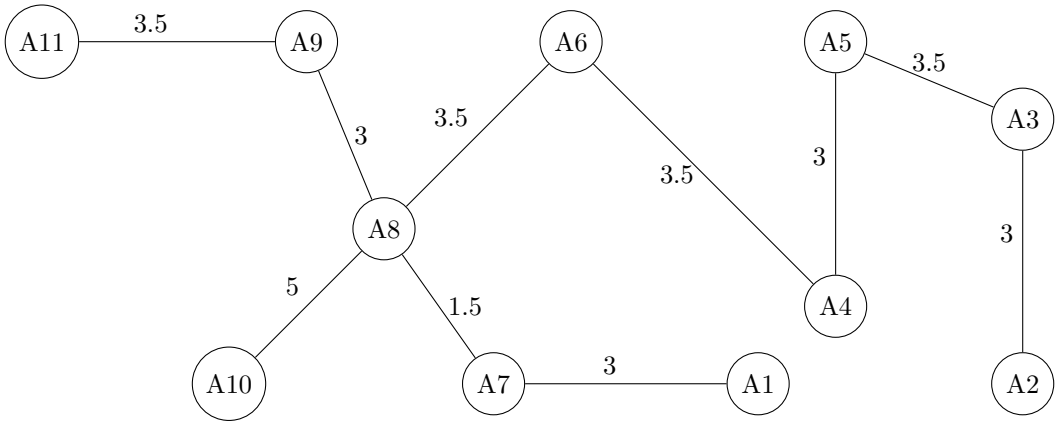

Figure 3: MST UNASAM mediante Algoritmo de Prim [5]

Para la primera red, los grafos no fueron identicos, pero la longitud de nuestro MST fue igual a la del artículo. La longitud del árbol de expansión mínima es 30.5 metros.

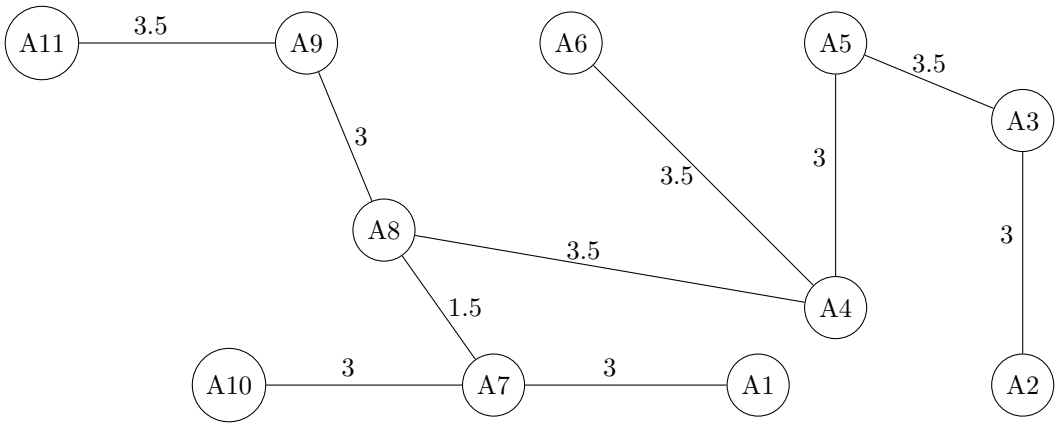

Figure 4: Resultado MST Primera Red (mediante Algoritmo de Prim y Kruskal)

La longitud original de la primera red era 35.6 metros. La red optimizada tiene una reducción de 5.1 metros de longitud.

4.2 Segunda Red

El segundo grafo propuesto es el siguiente:

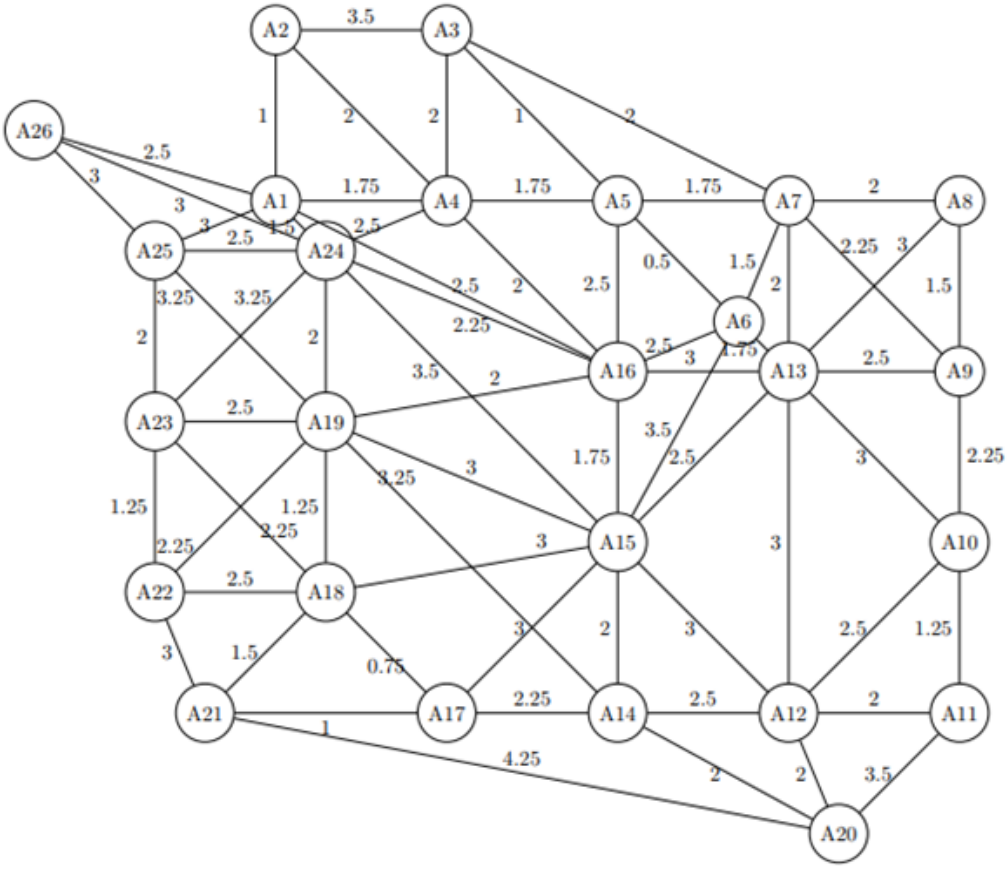

Figure 5: Segunda Red [5]

Al igual que con la primera red, los investigadores de UNASAM utilizaron el algoritmo de prim para obtener el MST de esta red.

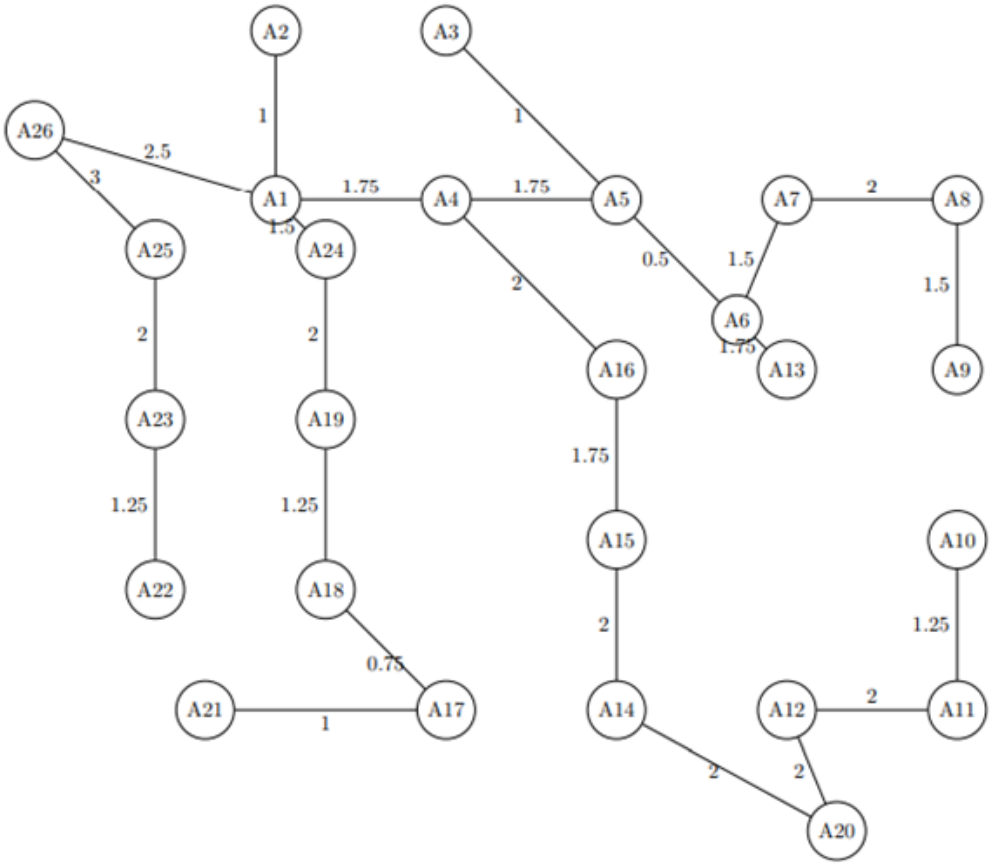

Figure 6: MST UNASAM mediante Algoritmo de Prim [5]

En la segunda red los resultados fueron diferentes. En el artículo indicaban que la longitud del árbol de expansión mínima era de 39.5 metros. En nuestro caso, se obtuvo una longitud de la red de 39.25 metros.

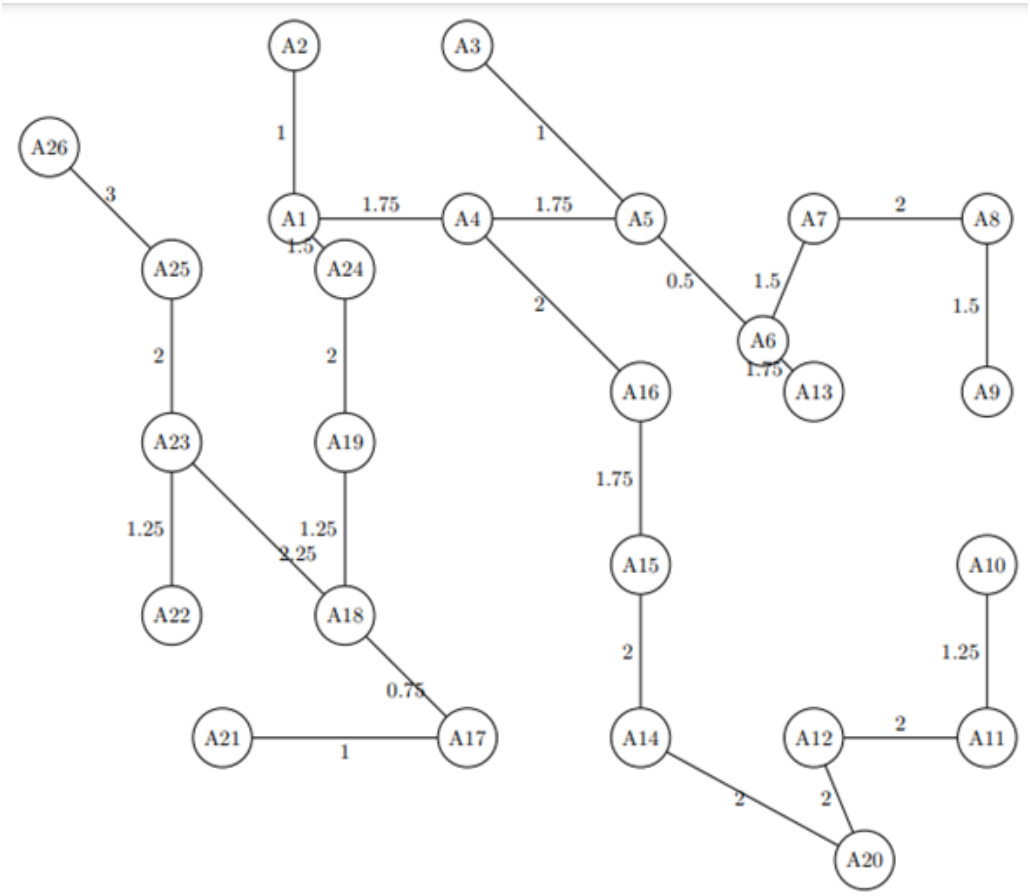

Figure 7: Resultado MST Segunda Red (mediante Algoritmo de Prim y Kruskal)

La longitud original de la segunda red era 50.75 metros. Nuestra red optimizada tiene una reducción de 11.5 metros de longitud.

5 Conclusiones

Los árboles de expansión mínima son una metodología de optimización de redes muy útil, aplicable a sistemas representados como grafos o redes. Dos de los principales algoritmos para obtener un árbol de expansión mínima son el algoritmo de Prim y el de Kruskal. Al aplicar estos algoritmos, observamos que ambos llegaron al mismo resultado. Por lo tanto no se pudo determinar que algoritmo puede optimizar de manera mas eficiente las redes.

Para probar los algoritmos, utilizamos las redes propuestas por Jossely Richard Aquino Dominguez y Vladimur Giovanni Rodriguez Sabino en su artículo para UNASAM [5]. En la primera red, nuestros resultados coincidieron con los del artículo, mientras que en la segunda red se pudo llegar a una red con un menor valor que los mencionados en el artículo.

Es importante destacar que la optimización de redes debería llevarse a cabo antes de su construcción, ya que modificar redes ya existentes trae muchas dificultades.

De esta manera, pudimos evidenciar la utilidad de estos algoritmos en la optimización de redes, lo cual nos permite reducir costos en la construcción e implementación de las mismas. Aunque en nuestro caso nos enfocamos en las redes eléctricas, esta metodología también puede aplicarse a otros tipos de redes, como las de telecomunicaciones u otras.

6 Referencias

1 Algoritmos greedy sobre grafos, Universidad de Granada, Departamento de Ciencias de la Computacion e I.A. https://elvex.ugr.es/decsai/algorithms/slides/problems/Greedy%20Graph%20Algorithms.pdf

2 Algoritmo de Prim, Universidad de Guanajuato https://nodo.ugto.mx/wp-content/uploads/2018/08/Prim.pdf

3 Minimum Spanning Tree, Aprende Programación Competitiva,20/09/2019, https://aprende.olimpiada-informatica.org/algoritmia-minimum-spanning-tree-prim-kruska

4 Algoritmo de Kruskal, Universidad de Guanajuato, https://nodo.ugto.mx/wp-content/uploads/2018/08/Kruskal.pdf

5. Reducción de Costos en las Instalaciones de Redes Eléctricas Usando Arboles de Expansión Mínima en Una Edificación de la Región Áncash, https://repositorio.unasam.edu.pe/bitstream/handle/UNASAM/3364/T033_41698484_T.pdf?sequence=1&isAllowed=y