Alberto Alvarado Sandoval
Garcia Alvarez Corinne Angelica
Delgado Alarcon Alan Ignacio
Carrillo Rodríguez Antonio
Casiano Granados Brandon Antonio
Hernández Méndez Oliver Manuel
Arango León Moisés

Kruskal Algorithm

By: The best ever team Azulito

Joseph B. Kruskal

29 de enero de 1928 – 19 de septiembre de 2010

- *Investigador de Math Center (Bell Labs)*
- *1956 crea el algoritmo para dar solución al problema de Minimum Spanning Tree (MST) el cual es un problema de optimización combinatoria.*

Objetivo del algoritmo

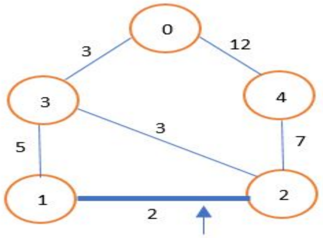
Encontrar un árbol de expansión mínimo a partir de un grafo que está conectado.

Este árbol debe estar formado por arcos sucesivos seleccionados de peso mínimo.

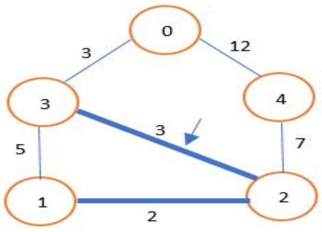
Se trata cada nodo como un árbol independiente y conecta uno con otro solo si tiene el menor costo en comparación con todas las demás opciones disponibles.

Al resultado de aplicar el algoritmo de Kruskal también se le conoce como ***Árbol Recubridor Minimal.***

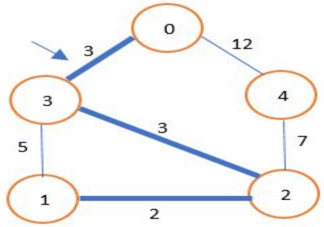
Ejemplo



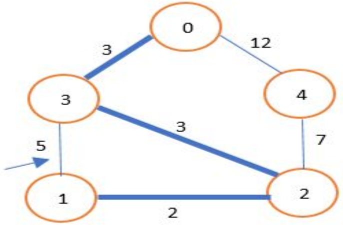
1



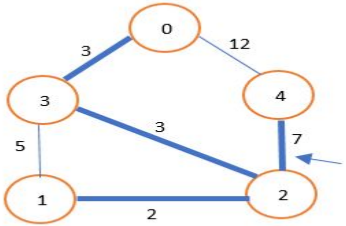
2



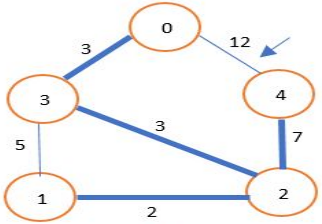
3



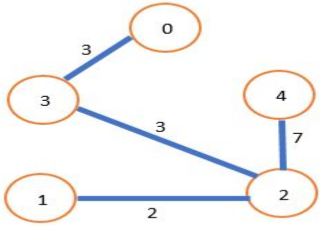
4



5



6



7

Aplicaciones

- ➔ Diseño de cableado eléctrico
- ➔ Colocación de cables telefónicos
- ➔ Conexiones de computadoras en LAN

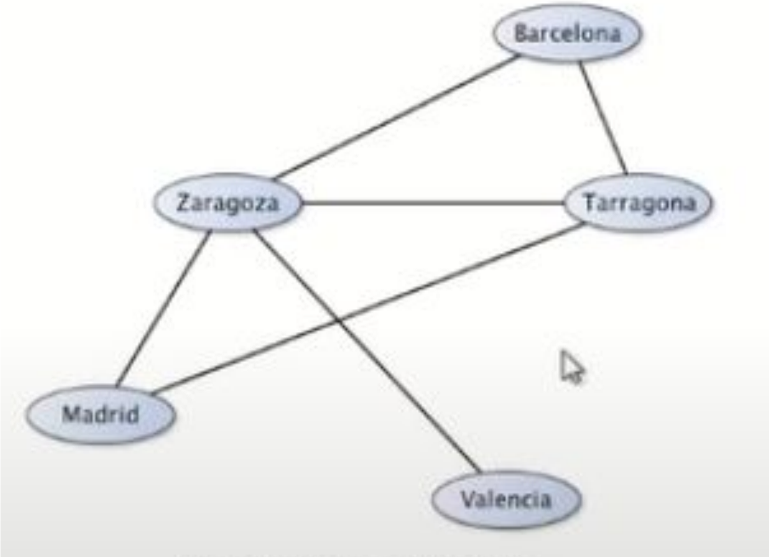
Disminución en costo de materiales, redundancia de rutas, y ruido generado en comunicaciones.

Funcionamiento

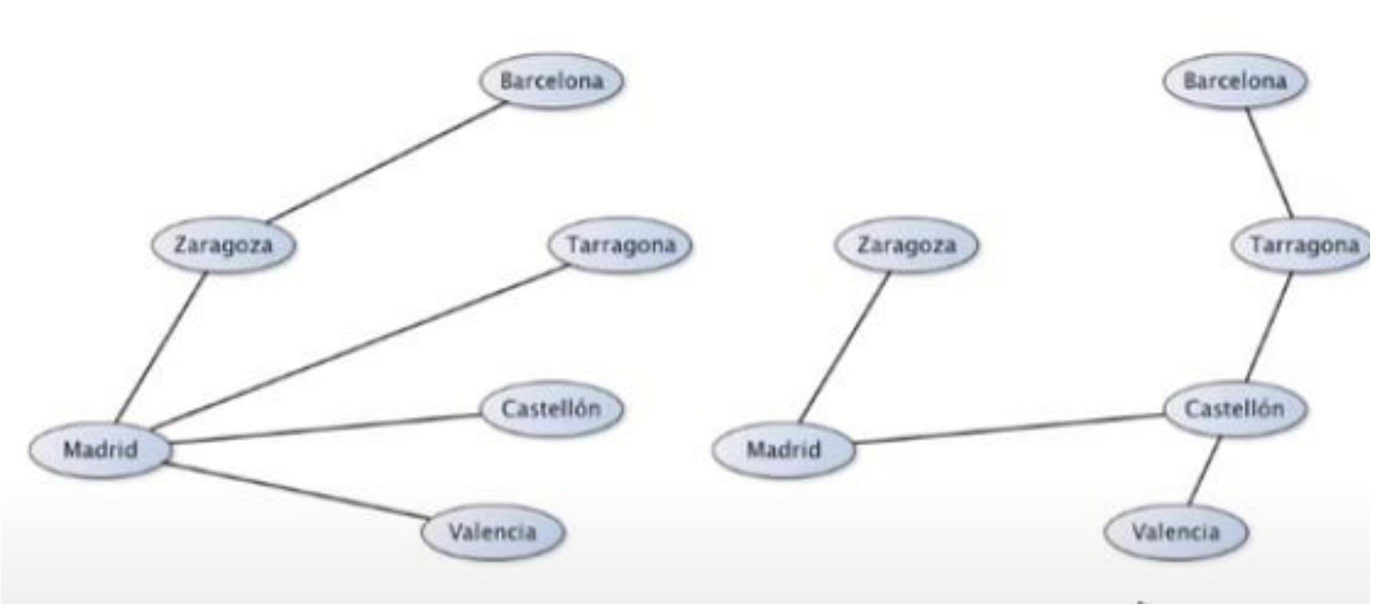
Algoritmo de la teoría de grafos para encontrar un árbol recubridor mínimo en un grafo conexo y ponderado.

Spanish def. Visitar todos los nodos *(puntitos)* y que el valor de la suma de las aristas *(rayitas)* por las que pasa sea la menor.

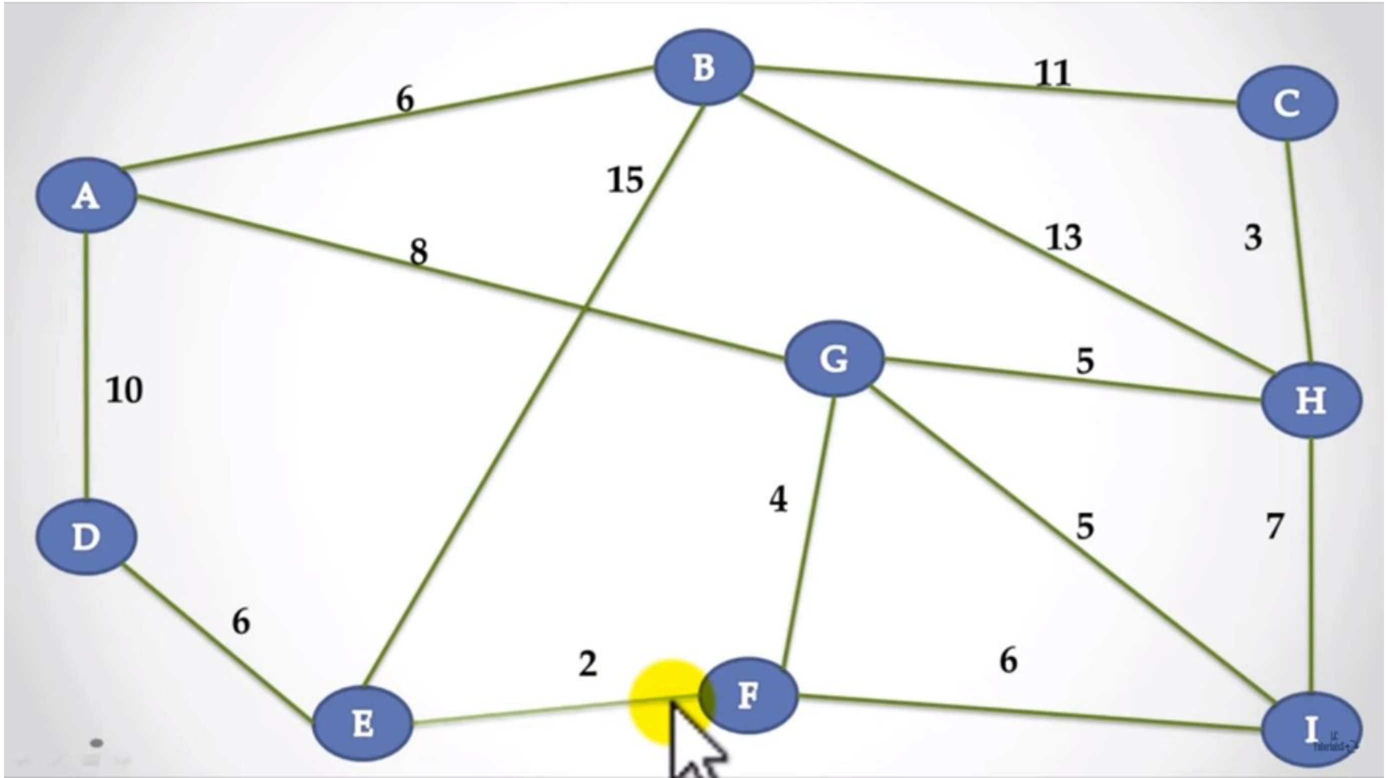
Árbol generador

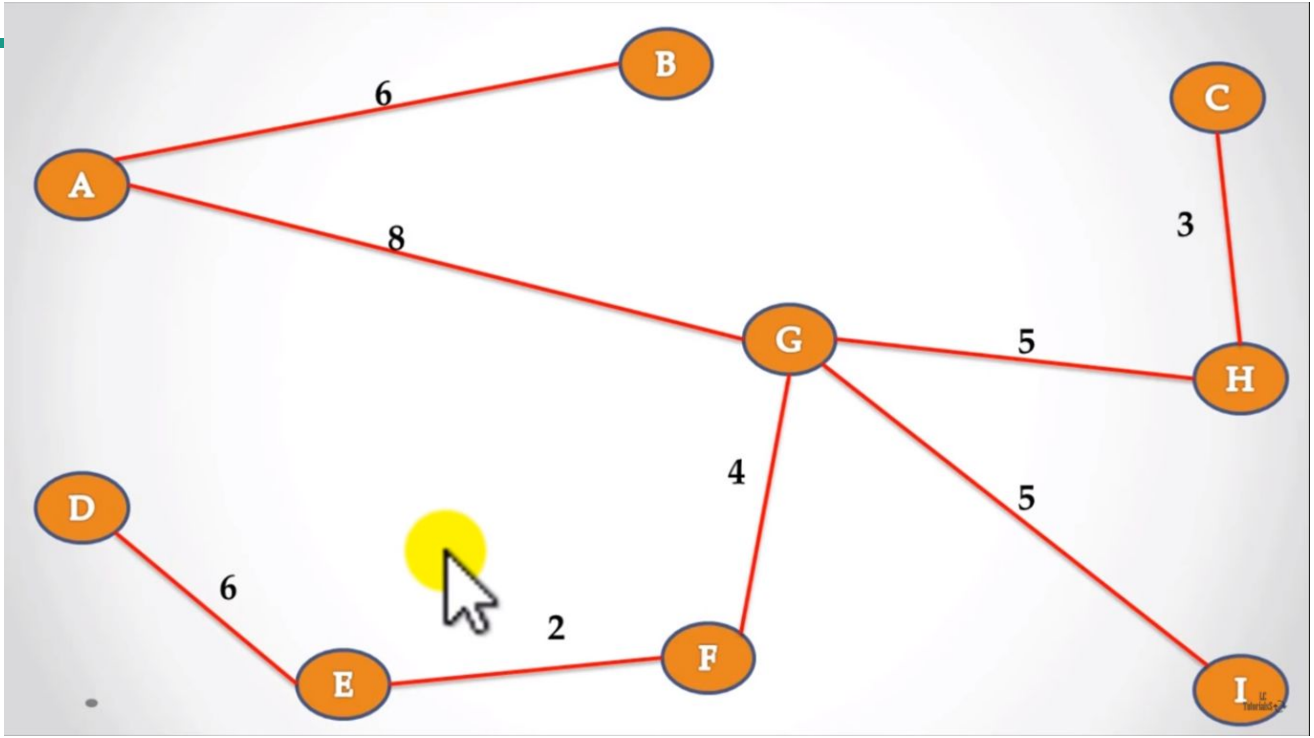


Árbol Generador



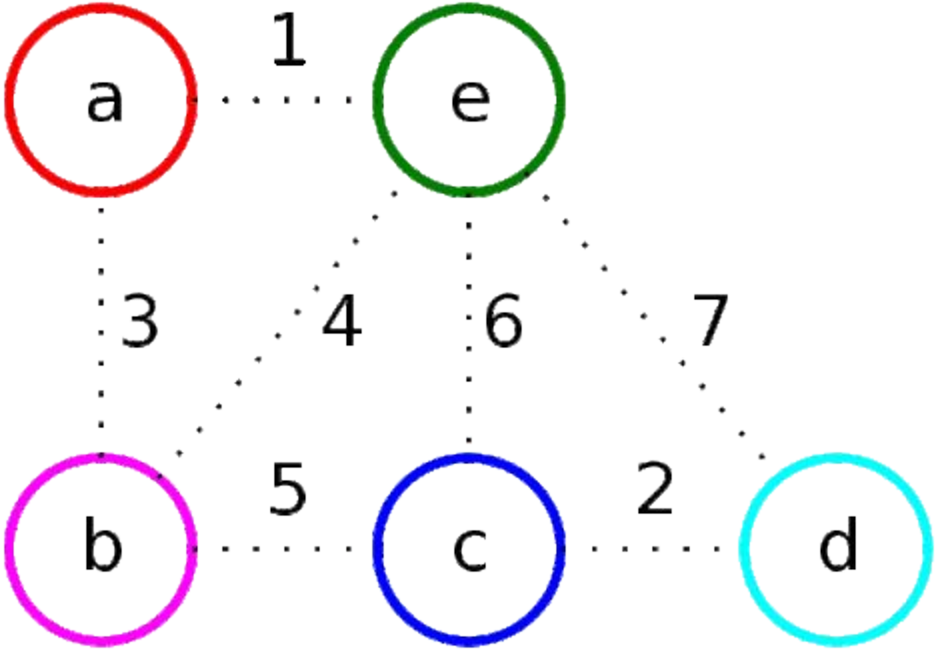
Árbol Generador de Mínimo Peso





A
B
C
D
E
F
G
H
I
6
8
3
5
4
5
6
2

Edge	ab	ae	bc	be	cd	ed	ec
Weight	3	1	5	4	2	7	6



Complejidad

m número de aristas, y n el número de vértices, el algoritmo muestra una complejidad:

- $O(m \log m)$
- ó $O(m \log n)$ si se ejecuta sobre estructuras de datos simples.

Los tiempos de ejecución son equivalentes porque:

- m es a lo mas n^2 y $\log n^2 = 2\log n$ es $O(\log n)$.
- ignorando los vértices aislados, los cuales forman su propia componente del árbol de expansión mínimo, $n \leq 2m$, así que $\log n$ es $O(\log m)$.

Obtener complejidad

Se ordenan las aristas de acuerdo a su peso usando comparación (***Comparison sort***) con una complejidad del orden de O(m log m); esto permite que el paso "eliminar una arista de peso mínimo" se ejecute en tiempo constante.

En caso de usar **merge sort** para ordenar el tamaño de las aristas la complejidad también es O(m log m)

```
A=∅
for all vértice v ∈ G.V do
  Hacer-Conjunto(v)
end for
ordena las aristas de G.E en un orden no decreciente por el peso w.
    for all arista(u, v) ∈ G.E, tomado en orden no decreciente por peso
do
         if Encontrar-Conjunto(u) = Encontrar-Conjunto(v) then
            A = A ∪ (u, v)
            Unión (u, v)
        end if
    end for

return A
```

Pseudocódigo

Referencias

- ➔ *algoritmo_kruskal – Grafos - software para la construcción, edición y análisis de grafos. (upv.es)*
- ➔ *What is the use of Kruskal's algorithm in real applications? - Quora*
- ➔ *Kruskal's Algorithm (programiz.com)*
- ➔ *Dasgupta, S., Algorithms, C. P. H., & Vazirani, U. (2006). Algorithms (1.a ed.). McGraw-Hill Education.*