El problema de árbol recubridor de peso mínimo y el algoritmo de Kruskal
por f6m

Dada una gráfica simple y conexa $G(V,E)$ con pesos en las aristas y con matriz cuadrada de costos c_{ij} con $1 \leq i,j \leq |V|$, el algoritmo de Kruskal encuentra el árbol recubridor de peso mínimo. El algoritmo de la Figura 1 es una versión general del algoritmo de Kruskal en el cual podemos apreciar las entradas, salidas y proceso.

Algoritmo 4 KRUSKAL, versión general

Require: Conjunto $E(G)$ de aristas de G, órden de G n, Matriz de costos $COSTO$ de órden $n \times n$.
Ensure: Árbol recubridor para G: T, costo mínimo mc.

$V(T) = V(G), E(T) \leftarrow \emptyset$
while $|E(T)| < n-1$ y $E \neq \emptyset$ **do**
  Seleccionar $e' = (u,v) = min_{e=(i,j)\in E(G)} COSTO(i,j)$
  $BORRA(e')$ de E
  if $T \cup e'$ no forma un ciclo **then**
    $T \leftarrow T \cup e'$
  else
    Descartar e'
  end if
end while
$mc = \sum_{e \in E(T)} COSTO(e)$

Figura 1: Algoritmo de Kruskal

El algoritmo de Kruskal es un algoritmo voraz (glotón) y procede agregando aristas a un conjunto de manera óptima (buscando costos mínimos), preservando la estructura del árbol, esto es evitando ciclos. Termina (¿porque termina?) una vez que se han visitado todas las aristas posibles para un árbol esto es $n-1$ siendo n el orden de la gráfica de entrada. Sus entradas son una gráfica (simple) conexa y pesada, la matriz de costos para la gráfica, y sus salidas son el árbol recubridor de peso mínimo y el costo del árbol. El árbol recubridor puede no ser único. Para ejecutar el algoritmo de la Figura 1 manualmente tenemos que indicar cómo evolucionan las variables y/o indicar de forma gráfica cómo evoluciona la construcción del árbol recubridor mínimo.

Ejemplo

Supongamos que deseamos conocer el árbol recubridor de peso mínimo para la gráfica de la Figura 2.

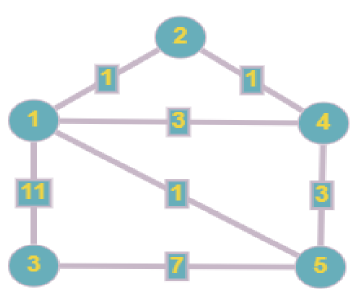


Figura 2: Grafica pesada y conexa inicial.

Una “ejecución manual” del algoritmo

Antes de iniciar el ciclo **while** el conjunto de vértices son todos los vértices de la gráfica inicial (línea 1) y el conjunto de aristas E(T) del árbol T es vacío y al igual la arista temporal e’ y E(G) tiene todas las aristas de la gráfica G. La tabla 1 muestra la evolución de los conjuntos E(G), E(T) y la cardinalidad E(T) en las iteraciones del while.

Iteración del while	*E(T)=∅, E(G)={12,24,14,15,13,45,35}*
1	*e’=12, E(G)={24,14,15,13,45,3,5}, E(T)={12}, |E(T)|=1*
2	*e’=15, E(G)={24,14,13,45,35}, E(T)={12,15}, |E(T)|=2*
3	*e’=24, E(G)={14,13,45,35}, E(T)={12,15,24}, |E(T)|=3*
4	*e’=14, E(G)={13,45,35}, E(T)={12,15,24}, |E(T)|=3, no se agrega crea un ciclo*
5	*e’=45, E(G)={13,35}, E(T)={12,15,24}, |E(T)|=3, no se agrega crea un ciclo*
6	*e’=35, E(G)={13}, E(T)={12,15,24,35}, |E(T)|=4 FIN pues n-1=5-1=4*

Tabla 1: Iteraciones del **while** en el algoritmo de Kruskal.

Finalmente, el árbol recubridor de peso mínimo es (V(T)={1,2,3,4,5}, *E(T)={12,15,24,35}*) y otra salida del algoritmo de Kruskal es el peso del árbol T, el cual es la suma de los pesos asociados a cada arista entonces mc=1+1+1+7=10 (línea 11).

Evolución grafica del algoritmo de Kruskal

La Figura 3 muestra cómo evoluciona de forma general la construcción del árbol recubridor de peso mínimo durante el algoritmo de Kruskal.

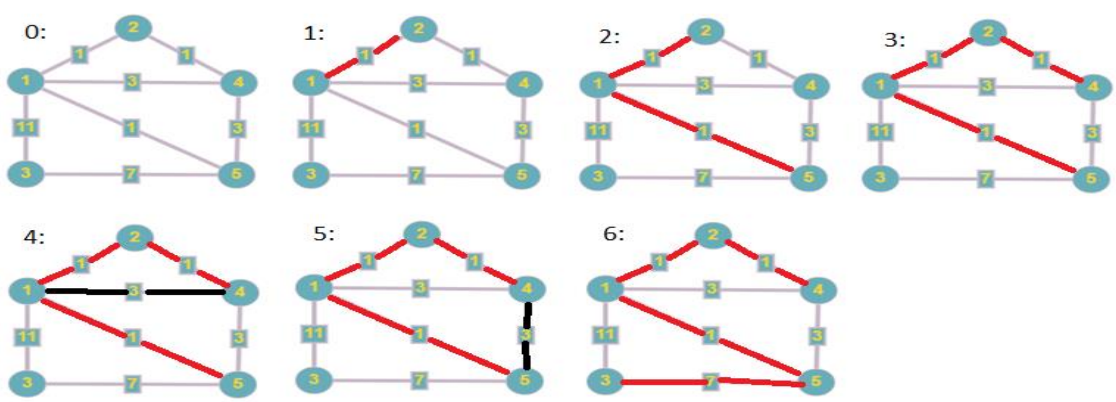
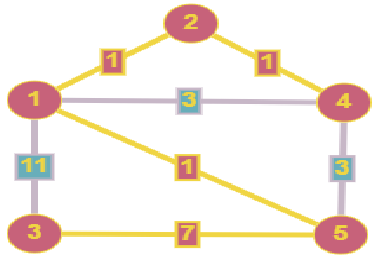


Figura 3: Evolución de las gráficas y del árbol recubridor de peso mínimo.

Solución dada por graphonline.ru



De la solución de graphonline.ru podemos observar que el árbol es el mismo camino obtenido y tiene costo mínimo igual a 10.

Referencias

Algoritmo de Kruskal. (2020, 16 de noviembre). *Wikipedia, La enciclopedia libre*. Fecha de consulta: 06:13, mayo 14, 2022 desde https://es.wikipedia.org/w/index.php?title=Algoritmo_de_Kruskal&oldid=130982430.