ÁRBOL RECUBRIDOR MÍNIMO

0. TENGAMOS EN CUENTA….

Los grafos no dirigidos se emplean para modelar relaciones simétricas entre entes, los vértices del grafo. Cualquier arista (v , w) de un grafo no dirigido está formada por un par no ordenado de vértices. Como consecuencia directa, la representación de un grafo no dirigido da lugar a matrices simétricas.

Una propiedad, que normalmente interesa conocer, de un grafo no dirigido es si para todo par de vértices existe un camino que los une; en definitiva, si el grafo es conexo. A los grafos conexos también se les denomina Red Conectada.

Buscar un árbol de expansión de un grafo, en una red, es una forma de averiguar si está conectado. Todos los vértices del grafo tienen que estar en el árbol de expansión para que sea un grafo conectado. La Figura 1 muestra un grafo no dirigido y su árbol de expansión; este grafo es una red conectada.

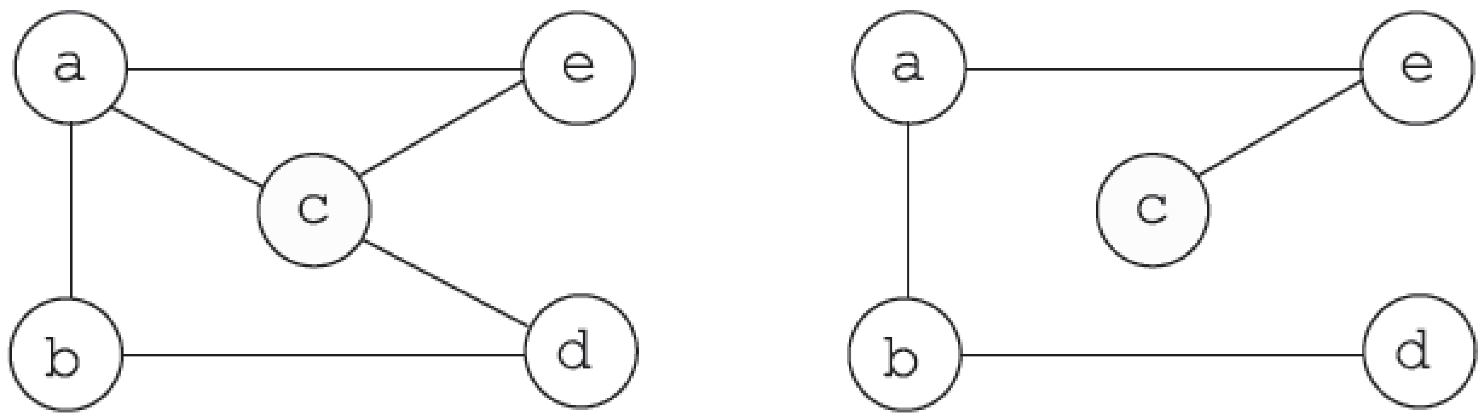


Figura 1. Grafo no dirigido y su árbol de expansión

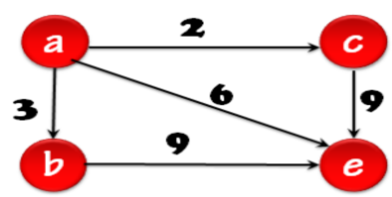
<table>
<tr>
<td>Es necesario que tengamos en cuenta el concepto de Grafo Ponderado, Etiquetado, Pesado o con Costos, que se refiere a aquel grafo donde cada arista tiene asociado un valor o etiqueta, para representar costo, peso, longitud, etc.</td>
<td>

Figura 2. Ejemplo de grafo ponderado o etiquetado</td>
</tr>
</table>

El problema del árbol recubridor mínimo o también llamado árbol de expansión de coste mínimo consiste en buscar un árbol que abarque todos los vértices del grafo, con suma de pesos de aristas mínimo. Los árboles de expansión se aplican en el diseño de redes de comunicación.

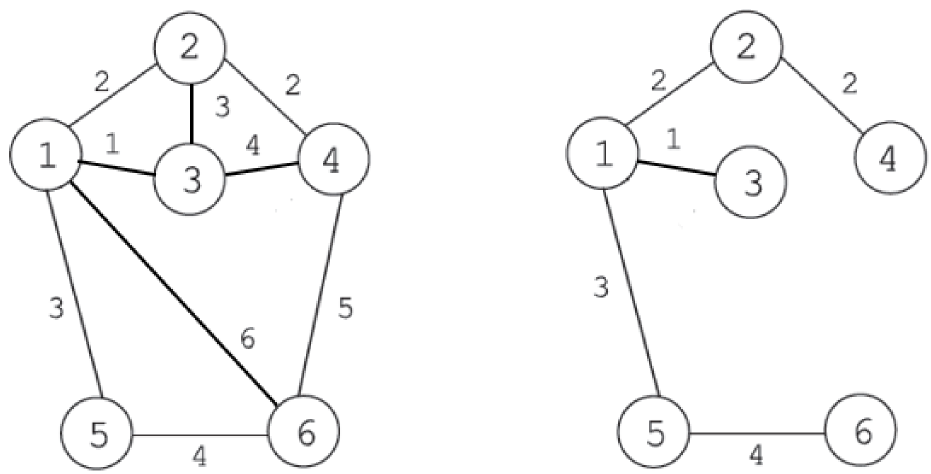


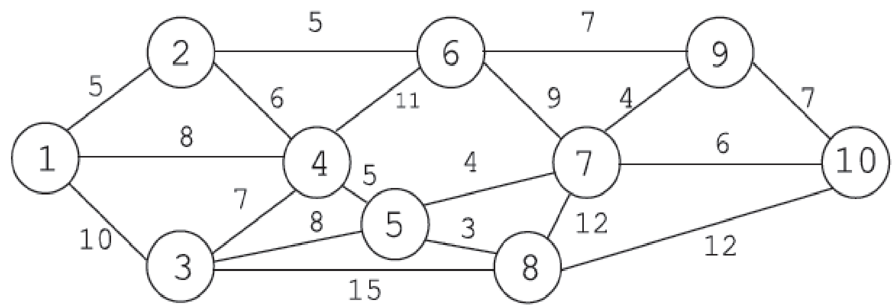
Figura 3. Ejemplo de grafo ponderado y su árbol recubridor mínimo o de expansión mínimo

El planteamiento del problema es el siguiente: dado un grafo no dirigido ponderado y conexo, encontrar el subconjunto del grafo compuesto por todos los vértices, con conexión entre cada par de vértices, sin ciclos y que cumpla que la suma de los pesos de las aristas sea mínimo. Para ello, se analizan 2 algoritmos, el de Prim y el de Kruskal.

1. ALTERNATIVA 1: EL ALGORITMO DE PRIM

El algoritmo de Prim encuentra el árbol de expansión mínimo de un grafo no dirigido. Realiza sucesivos pasos, siguiendo la metodología clásica de los algoritmos voraces: hacer en cada paso lo mejor que se pueda hacer. En este problema, lo mejor consiste en incorporar al árbol una nueva arista del grafo de menor longitud (o peso).

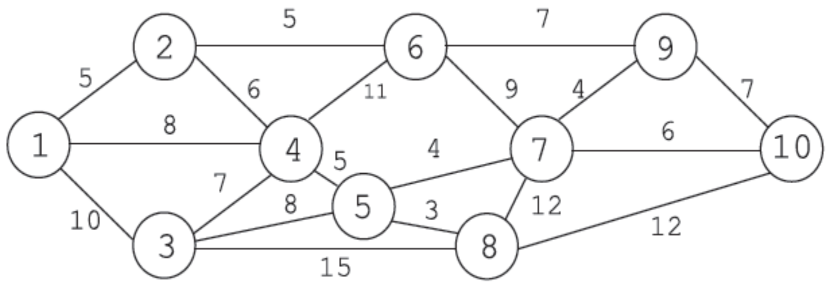
Dado el siguiente grafo ponderado, encuentre su árbol recubridor mínimo:



El algoritmo arranca asignando un vértice inicial al conjunto W; por ejemplo el vértice 1: W = {1}.

A partir del vértice inicial, el árbol de expansión crece, añadiendo a W, en cada pasada otro vértice z todavía no incluido en W, de tal forma que si u es un vértice cualquiera de W, la arista (u,z) es la más corta, la de menor coste. **El proceso termina cuando todos los vértices del grafo están en W**, y por consiguiente, el árbol de expansión con todos los vértices está formado; además es mínimo porque en cada pasada se ha añadido la menor arista.

Ejemplo 1. Obtenga el árbol recubridor mínimo para el siguiente grafo:



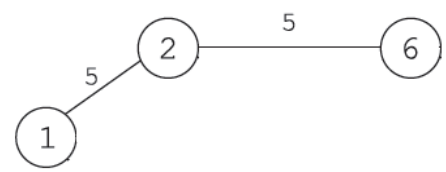
W = {1}. F={2, 3, 4, 5, 6, 7, 8, 9, 10}
Aristas del nodo 1 (Su costo entre paréntesis):
1-2 (5), 1-3 (10) y 1-4 (8)
La arista con costo menor es 1-2 (costo 5).

Se agrega el nodo 2 al conjunto W = {1 , 2}

W = {1 , 2}. F={3, 4, 5, 6, 7, 8, 9, 10}

Aristas del nodo 1: 1-3 (10) y 1-4 (8)
Aristas del nodo 2: 2-4 (6) y 2-6 (5)

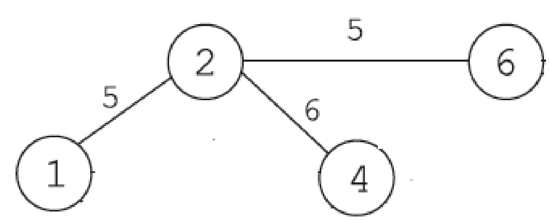
La arista con costo menor es 2-6 (costo 5).
Se agrega el nodo 6 al conjunto W = {1 , 2 , 6}



W = {1, 2, 6}. F={3, 4, 5, 7, 8, 9, 10}

Aristas del nodo 1: 1-3 (10) y 1-4 (8)
Aristas del nodo 2: ~~2-4 (6)~~, ~~2-6 (5)~~
Aristas del nodo 6: 4-6 (11), 6-7 (9), 6-9 (7)

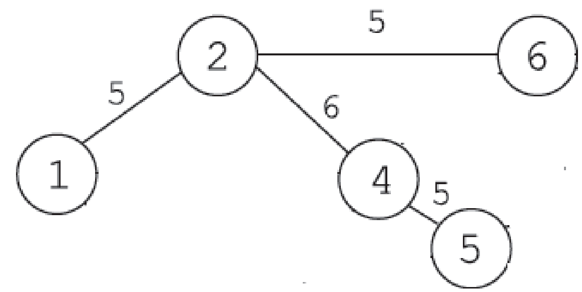
La arista con costo menor es 2-4 (costo 6).
Se agrega el nodo 4 al conjunto W = {1 , 2 , 6 , 4}



W = {1, 2, 6, 4}. F={3, 5, 7, 8, 9, 10}

Aristas del nodo 1: 1-3 (10) y 1-4 (8)
Aristas del nodo 2: ~~2-4 (6)~~, ~~2-6 (5)~~
Aristas del nodo 6: 4-6 (11), 6-7 (9), 6-9 (7)
Aristas del nodo 4: 3-4 (7) , 4-5 (5)

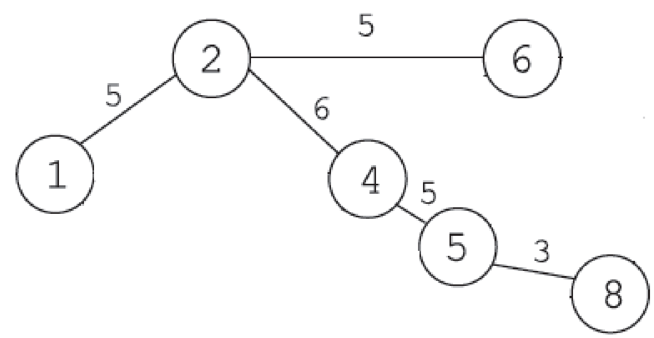
La arista con costo menor es 4-5 (costo 5).
Se agrega el nodo 5 al conjunto W = {1 , 2 , 6 , 4 , 5}



W = {1, 2, 6, 4, 5}. F={3, 7, 8, 9, 10}

Aristas del nodo 1: 1-3 (10) y 1-4 (8)
Aristas del nodo 2: ~~2-4 (6)~~, ~~2-6 (5)~~
Aristas del nodo 6: 4-6 (11), 6-7 (9), 6-9 (7)
Aristas del nodo 4: 3-4 (7) , ~~4-5 (5)~~
Aristas del nodo 5: 3-5 (8) , 5-7 (4) , 5-8 (3)

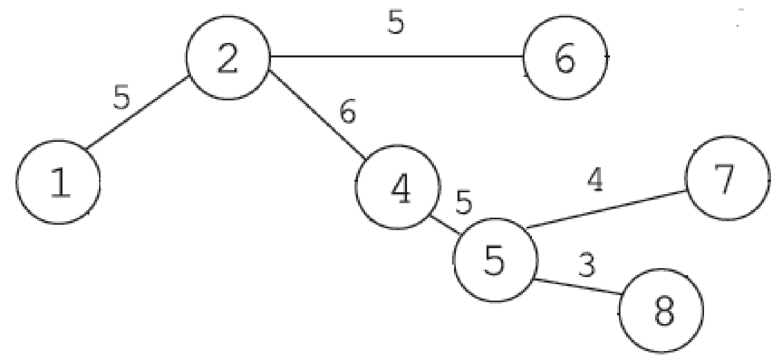
La arista con costo menor es 5-8 (costo 3).
Se agrega el nodo 8 a W = {1, 2, 6, 4, 5, 8}



W = {1, 2, 6, 4, 5, 8}. F={3, 7, 9, 10}

Aristas del nodo 1: 1-3 (10) y 1-4 (8)
Aristas del nodo 2: ~~2-4 (6)~~, ~~2-6 (5)~~
Aristas del nodo 6: 4-6 (11), 6-7 (9), 6-9 (7)
Aristas del nodo 4: 3-4 (7) , ~~4-5 (5)~~
Aristas del nodo 5: 3-5 (8) , 5-7 (4) , ~~5-8 (3)~~
Aristas del nodo 8: 3-8 (15), 7-8 (12), 8-10 (12)

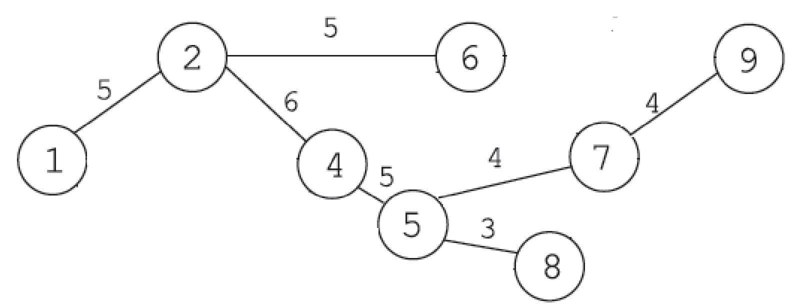
La arista con costo menor es 5-7 (costo 4).
Se agrega el nodo 7 a W = {1, 2, 6, 4, 5, 8, 7}



W = {1, 2, 6, 4, 5, 8, 7}. F={3, 9, 10}

Aristas del nodo 1: 1-3 (10) y 1-4 (8)
Aristas del nodo 2: ~~2-4 (6)~~, ~~2-6 (5)~~
Aristas del nodo 6: 4-6 (11), 6-7 (9), 6-9 (7)
Aristas del nodo 4: 3-4 (7) , ~~4-5 (5)~~
Aristas del nodo 5: 3-5 (8) , ~~5-7 (4)~~ , ~~5-8 (3)~~
Aristas del nodo 8: 3-8 (15), 7-8 (12), 8-10 (12)
Aristas del nodo 7: 7-9 (4), 7-10 (6)

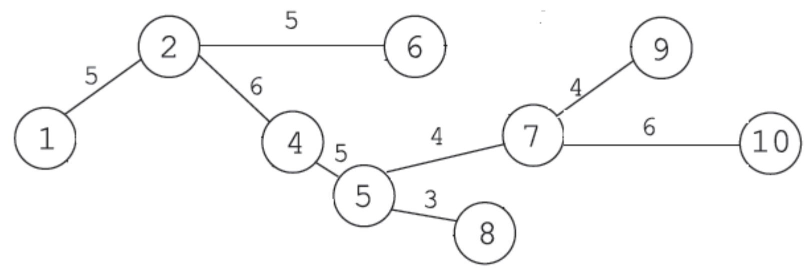
La arista con costo menor es 7-9 (costo 4).
Se agrega el nodo 9 a W = {1, 2, 6, 4, 5, 8, 7, 9}



W = {1, 2, 6, 4, 5, 8, 7, 9}. F={3, 10}

Aristas del nodo 1: 1-3 (10) y 1-4 (8)
Aristas del nodo 2: ~~2-4 (6)~~, ~~2-6 (5)~~
Aristas del nodo 6: 4-6 (11), 6-7 (9), 6-9 (7)
Aristas del nodo 4: 3-4 (7) , ~~4-5 (5)~~
Aristas del nodo 5: 3-5 (8) , ~~5-7 (4)~~ , ~~5-8 (3)~~
Aristas del nodo 8: 3-8 (15), 7-8 (12), 8-10 (12)
Aristas del nodo 7: ~~7-9 (4)~~, 7-10 (6)
Aristas del nodo 9: 9-10 (7)

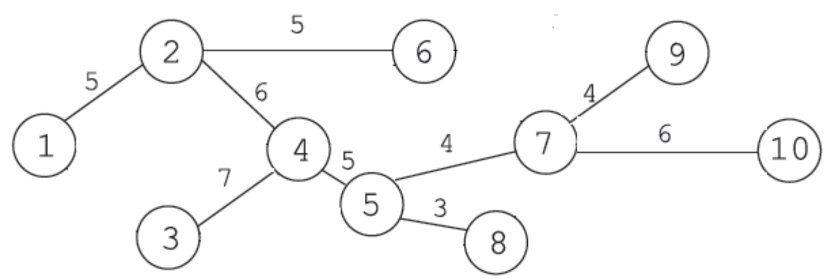
La arista con costo menor es 7-10 (costo 6).
Se agrega el nodo 10 a W = {1, 2, 6, 4, 5, 8, 7, 9, 10}



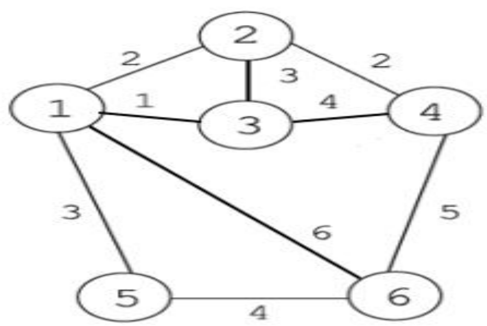
W = {1, 2, 6, 4, 5, 8, 7, 9, 10}. F={3}

Aristas del nodo 1: 1-3 (10) y 1-4 (8)
Aristas del nodo 2: ~~2-4 (6)~~, ~~2-6 (5)~~
Aristas del nodo 6: 4-6 (11), 6-7 (9), 6-9 (7)
Aristas del nodo 4: 3-4 (7) , ~~4-5 (5)~~
Aristas del nodo 5: 3-5 (8) , ~~5-7 (4)~~ , ~~5-8 (3)~~
Aristas del nodo 8: 3-8 (15), 7-8 (12), 8-10 (12)
Aristas del nodo 7: ~~7-9 (4)~~, ~~7-10 (6)~~
Aristas del nodo 9: 9-10 (7)
Las demás aristas formarían ciclos, se descartan.
Falta el nodo 3: 3-4 (7), 3-5 (8), 3-8 (15)

La arista con costo menor es 3-4 (costo 7).
Se agrega el nodo 3 a W = {1, 2, 6, 4, 5, 8, 7, 9, 10, 3}
Todos los nodos están en W, se termina el proceso.



Ejemplo 2. Obtenga el árbol recubridor mínimo para el siguiente grafo:



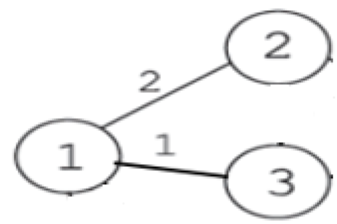
W = {1}. F={2, 3, 4, 5, 6}
Aristas del nodo 1 (Su costo entre paréntesis):
1-2 (2), 1-3 (1), 1-5 (3), 1-6 (6)
La arista con costo menor es 1-3 (costo 1).

Se agrega el nodo 3 al conjunto W = {1 , 3}

W = {1 , 3}. F={2, 4, 5, 6}

Aristas del nodo 1: 1-2 (2), ~~1-3 (1)~~, 1-5 (3), 1-6 (6)
Aristas del nodo 3: 2-3 (3), 3-4 (4)

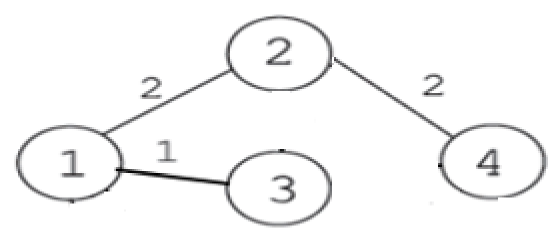
La arista con costo menor es 1-2 (costo 2).
Se agrega el nodo 2 al conjunto W = {1, 3, 2}



W = {1, 3, 2}. F={4, 5, 6}

Aristas del nodo 1: ~~1-2 (2), 1-3 (1)~~, 1-5 (3), 1-6 (6)
Aristas del nodo 3: 2-3 (3), 3-4 (4)
Aristas del nodo 2: 2-4 (2)

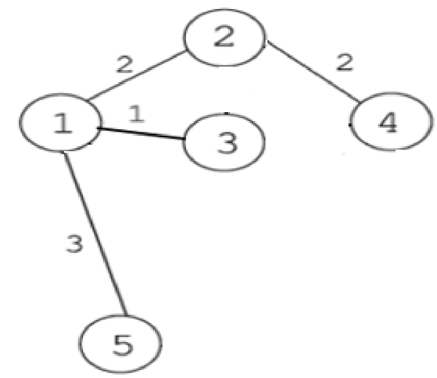
La arista con costo menor es 2-4 (costo 2).
Se agrega el nodo 4 al conjunto W = {1, 3, 2, 4}



W = {1, 3, 2, 4}. F={5, 6}

Aristas del nodo 1: ~~1-2 (2), 1-3 (1)~~, 1-5 (3), 1-6 (6)
Aristas del nodo 3: ~~2-3 (3)~~, 3-4 (4)
Aristas del nodo 2: ~~2-4 (2)~~
Aristas del nodo 4: 4-6 (5)

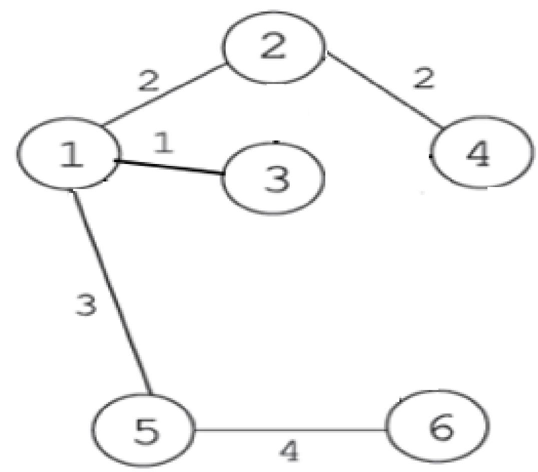
La arista con costo menor es 1-5 (costo 3).
Se excluye la arista 2-3, aunque su costo también es 3, debido a que con esa arista se forma un ciclo, lo que NO está permitido.
Se agrega el nodo 5 al conjunto W = {1, 3, 2, 4, 5}



W = {1, 3, 2, 4, 5}. F={6}

Aristas del nodo 1: ~~1-2 (2), 1-3 (1)~~, ~~1-5 (3)~~, 1-6 (6)
Aristas del nodo 3: ~~2-3 (3)~~, ~~3-4 (4)~~
Aristas del nodo 2: ~~2-4 (2)~~
Aristas del nodo 4: 4-6 (5)
Aristas del nodo 5: 5-6 (4)

La arista con costo menor es 5-6 (costo 4).
Se excluye la arista 3-4, aunque su costo también es 4, debido a que con esa arista se forma un ciclo, lo que NO está permitido.
Se agrega el nodo 6 al conjunto W = {1, 3, 2, 4, 5, 6}
Todos los nodos están en W, se termina el proceso.



2. ALTERNATIVA 2: EL ALGORITMO DE KRUSKAL

El algoritmo de Kruskal propone otra estrategia para encontrar el árbol de expansión de coste mínimo.

El árbol se empieza a construir con todos los vértices del grafo G pero sin aristas; se puede afirmar que cada vértice es una componente conexa en sí misma.

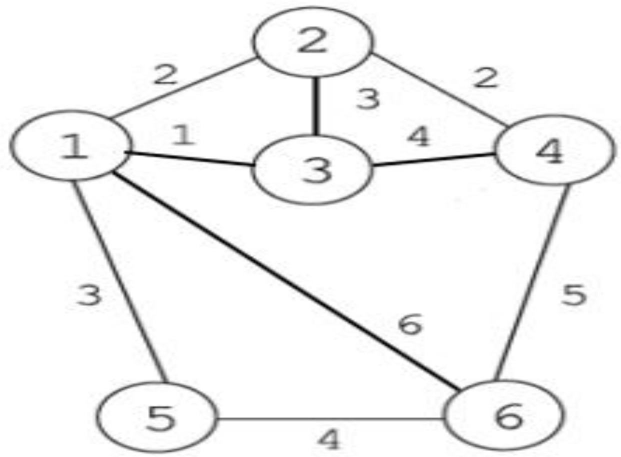
El algoritmo construye componentes conexas cada vez mayores examinando las aristas del grafo **en orden creciente del peso**.

Si la arista conecta dos vértices que se encuentran en dos componentes conexas distintas, entonces se añade la arista al árbol de expansión T.

En el proceso, se descartan las aristas que conectan dos vértices pertenecientes a la misma componente, ya que darían lugar a un ciclo.

Cuando todos los vértices están en un solo componente, T, éste es el árbol de expansión de coste mínimo del grafo G.

Ejemplo 3. Usando el algoritmo de Kruskal, obtenga el árbol recubridor mínimo para el siguiente grafo:

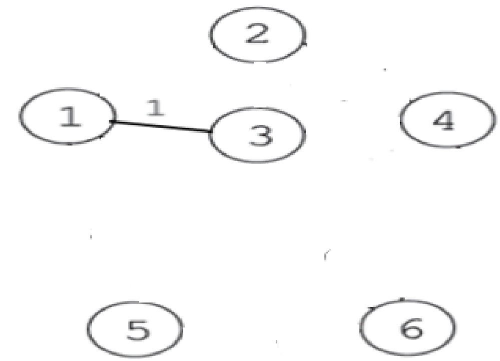


Se elabora el listado de aristas en orden creciente de peso:

ARISTA	PESO
1-3	1
1-2 , 2-4	2
1-5 , 2-3	3
3-4 , 5-6	4
4-6	5
1-6	6

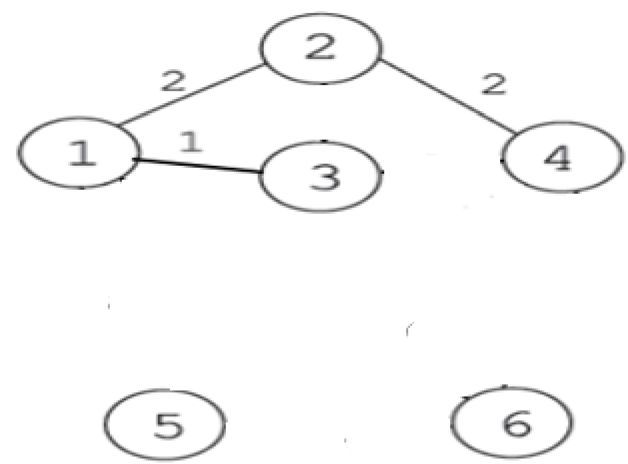
Al grafo SIN aristas agregamos las aristas de peso 1.
En este caso, la 1-3.

G = {1, 3}. F={2, 4, 5, 6}



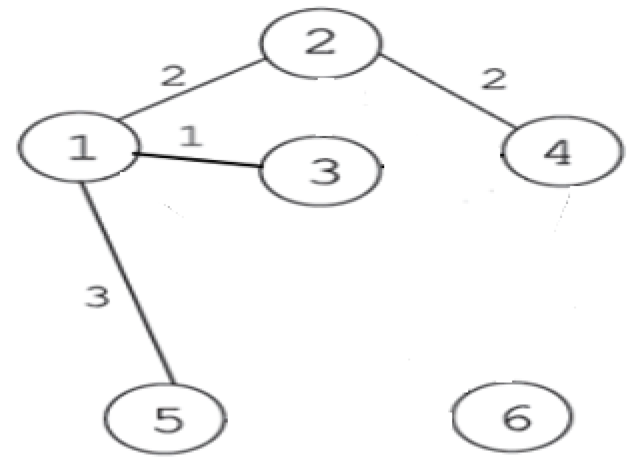
Agregamos las aristas de peso 2.
En este caso, la 1-2, y la 2-4.

G = {1, 3, 2, 4}. F={5, 6}



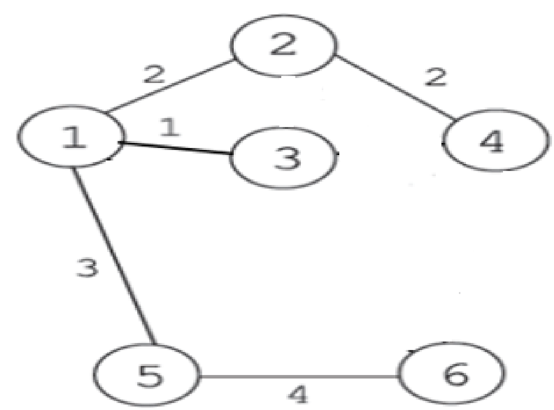
Agregamos las aristas de peso 3.
En este caso, solo la 1-5,
pues con la 2-3 se forma un ciclo (NO permitido)

G = {1, 3, 2, 4, 5}. F={ 6 }

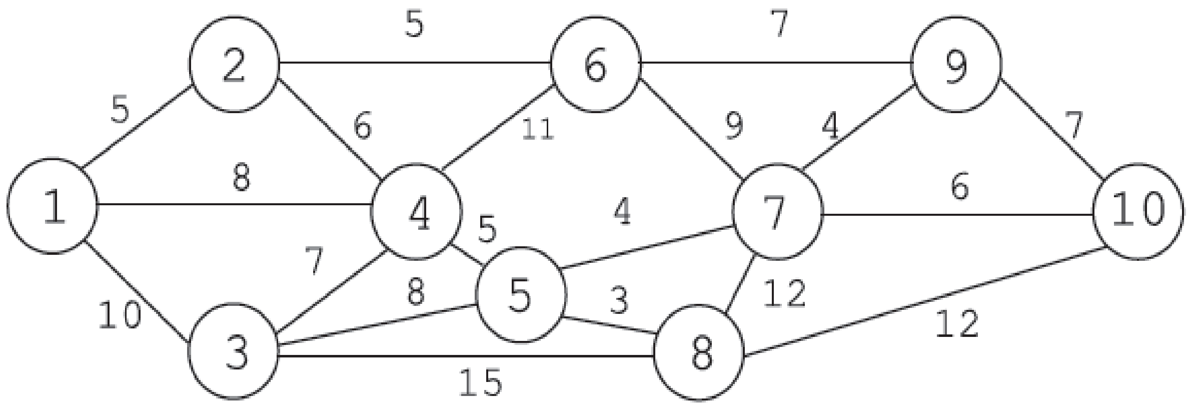


Agregamos las aristas de peso 4.
En este caso, solo la 5-6,
pues con la 3-4 se forma un ciclo (NO permitido).

G = {1, 3, 2, 4, 5, 6}. F={ }
Ya están todos los nodos se termina el proceso.



Ejemplo 4. Usando el algoritmo de Kruskal, obtenga el árbol recubridor mínimo para el siguiente grafo:



Se elabora el listado de aristas en orden creciente de peso:

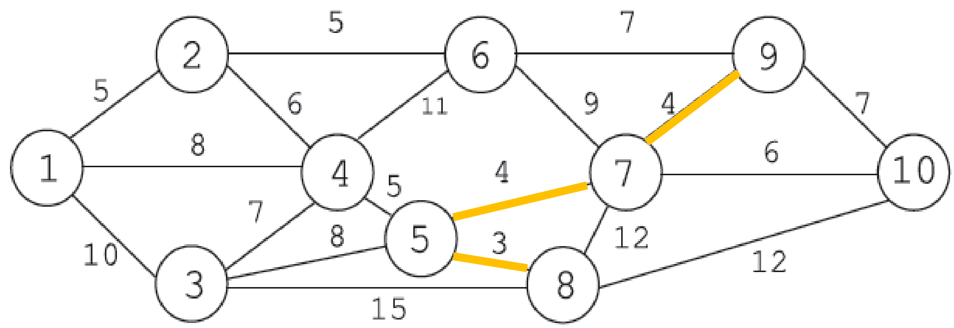
ARISTA	PESO
1-2 , 2-6 , 4-5	5
1-3	10
1-4 , 3-5	8
2-4 , 7-10	6
3-4 , 6-9 , 9-10	7
3-8	15
4-6	11
5-7 , 7-9	4
5-8	3
6-7	9
7-8 , 8-10	12

ARISTA	PESO
5-8	**3**
5-7 , 7-9	**4**
1-2 , 2-6 , 4-5	**5**
2-4 , 7-10	**6**
3-4 , 6-9 , 9-10	**7**
1-4 , 3-5	**8**
6-7	**9**
1-3	**10**
4-6	**11**
7-8 , 8-10	**12**
3-8	**15**

Y señalamos en el grafo las aristas de menor peso según lo indicado en la Tabla de la página anterior.

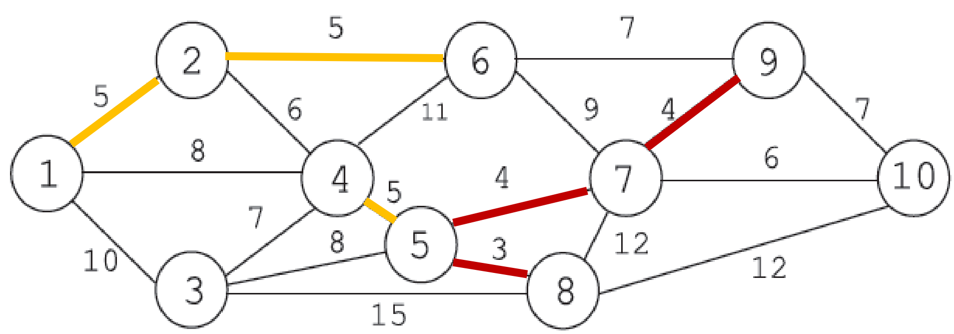
ARISTA	PESO
5-8	**3**
5-7 , 7-9	**4**
1-2 , 2-6 , 4-5	**5**
2-4 , 7-10	**6**
3-4 , 6-9 , 9-10	**7**
1-4 , 3-5	**8**
6-7	**9**
1-3	**10**
4-6	**11**
7-8 , 8-10	**12**
3-8	**15**

F = {1, 2, 3, 4, 6, 10}   V = {5, 7, 8, 9}



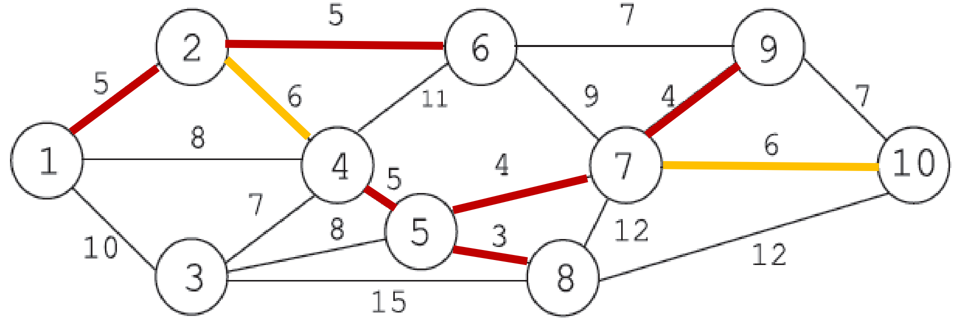
ARISTA	PESO
5-8	**3**
5-7 , 7-9	**4**
1-2 , 2-6 , 4-5	**5**
2-4 , 7-10	**6**
3-4 , 6-9 , 9-10	**7**
1-4 , 3-5	**8**
6-7	**9**
1-3	**10**
4-6	**11**
7-8 , 8-10	**12**
3-8	**15**

F = {3, 4, 10}   V = {1, 2, 5, 6, 7, 8, 9}



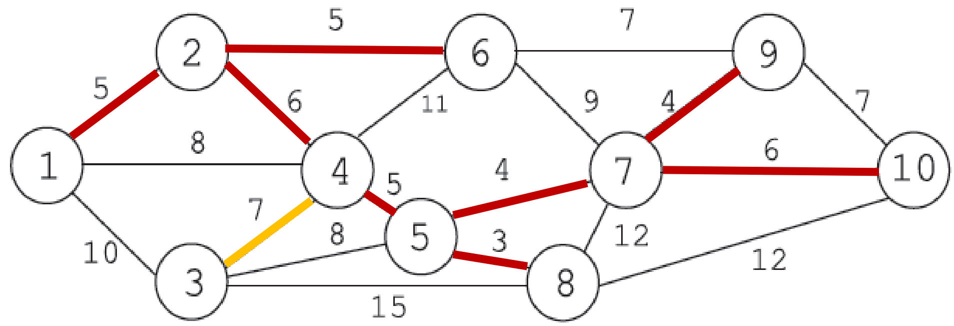
F = { 3 }   V = {1, 2, 4, 5, 6, 7, 8, 9, 10}

ARISTA	PESO
5-8	**3**
5-7 , 7-9	**4**
1-2 , 2-6 , 4-5	**5**
2-4 , 7-10	**6**
3-4 , 6-9 , 9-10	**7**
1-4 , 3-5	**8**
6-7	**9**
1-3	**10**
4-6	**11**
7-8 , 8-10	**12**
3-8	**15**



F = { }   V = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

ARISTA	PESO
5-8	**3**
5-7 , 7-9	**4**
1-2 , 2-6 , 4-5	**5**
2-4 , 7-10	**6**
3-4 , ~~6-9 , 9-10~~	**7**
1-4 , 3-5	**8**
6-7	**9**
1-3	**10**
4-6	**11**
7-8 , 8-10	**12**
3-8	**15**



-- FIN DEL DOCUMENTO