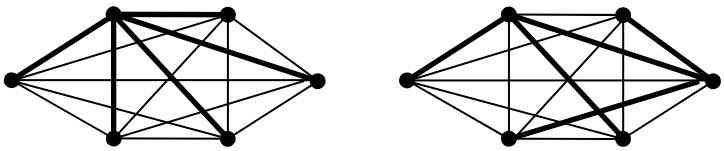
5.6 Árbol generador de un grafo

Definición 5.59. *Sea G un grafo simple. Un* **árbol generador** *de G es un subgrafo de G que es un árbol y contiene todos los vértices de G.*

Ejemplo 5.60. *Un grafo y algunos de sus árboles generadores en trazo más grueso*



Lema 5.61. *Dado un grafo simple conexo que contiene algún ciclo, si se elimina una arista del ciclo el subgrafo resultante sigue siendo conexo.*

Demostración.

Basta observar que, en un ciclo, para dos vértices adyacentes existen dos caminos que los unen, uno la arista que incide en ambos, otro el formado por las restantes aristas del ciclo. Por tanto, si se elimina dicha arista de un camino puede ser sustituida por el camino que forman las restantes aristas del ciclo.

■

Teorema 5.62. *Un grafo simple es conexo si, y sólo si, tiene un árbol generador.*

Demostración.

Sea *G* un grafo simple que admite un árbol generador *T*. *T* es subgrafo conexo de *G*, y contiene a todos los vértices de *G*. Por tanto, para cada par de vértices de *G* existe un camino en *T* que los une, este camino también lo es de *G*. Por lo cual *G* es conexo.

Recíprocamente, los tres algoritmos que se van a describir son una demostración constructiva de la existencia de árbol generador. Bastaría considerar la construcción del algoritmo de Prim[1] dando a todas las aristas del grafo un peso igual a 1.

■

Observación 5.63. Los árboles generadores de un grafo con *n* vértices tienen siempre n vértices y *n*-1 aristas.

Todos los procesos de construcción de árboles generadores de un grafo con *n* vértices terminan cuando el número de vértices es igual a *n* o el número de aristas es igual a *n*-1.

---

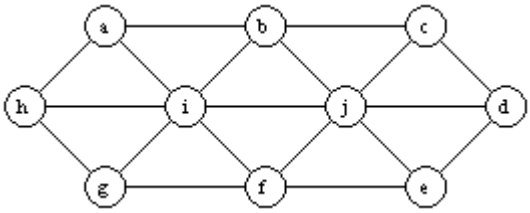
[1] Ver página 92

Algoritmo 5.64. Búsqueda en profundidad (BEP).

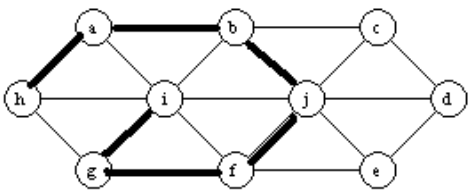
Sea G el grafo de partida. Se construye un árbol con raíz, en un proceso en el que intervienen caminos que nos llevan de la raíz a cada una de las hojas. En cada paso hay que tener en cuenta el subárbol construido hasta ese momento, que llamaremos árbol parcial, y el último vértice añadido, que se conoce como vértice activo.

- Inicio: Se elige de forma arbitraria un vértice como raíz r.
- Se supone construido un árbol parcial W, siendo u el vértice activo y $\{w, u\}$ la arista añadida.
- Veamos cómo se añade la siguiente arista. Hay dos situaciones:
  (1) El vértice u tiene vértices adyacente en G que no están en W:
  Se elige uno de ellos, z y se añade la arista $\{u, z\}$ al árbol T; el nuevo vértice activo es z.
  (2) Todos los vértices en G adyacentes a u están en W:
  Se retrocede hasta el vértice w, considerándolo como vértice activo.
- El proceso termina cuando todos los vértices de G están en T.

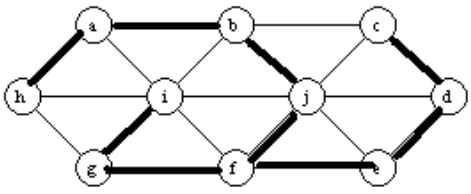
Ejemplo 5.65. *Usando algoritmo BEP construir un árbol generador del grafo de la figura:*



- *Paso* 0: Se elige un vértice raíz, h,
- *Paso* 1: a es vértice adyacente a h, se añade la arista $\{h, a\}$ al subgrafo anterior
- En los pasos del 2 al 6 se van añadiendo las aristas $\{a, b\}$, $\{b, j\}$, $\{j, f\}$, $\{f, g\}$, $\{g, i\}$ al subgrafo del paso anterior
- *Paso* 7: el vértice activo es i, sus vértices adyacentes están en el subárbol. Se retrocede hasta el vértice g, considerándolo el vértice activo.
- *Paso* 8: el vértice activo es g, le ocurre lo mismo. Se retrocede hasta el vértice f.
- *Paso* 9: e es vértice adyacente de f, se añade la arista $\{f, e\}$ al subgrafo anterior
- En los pasos del 10 y 11 se van añadiendo las aristas $\{e, d\}$ y $\{d, c\}$

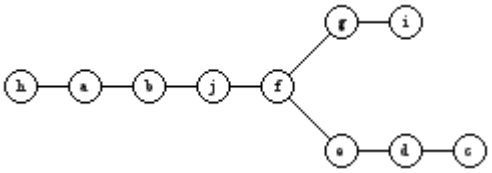


Pasos 1 al 6



Pasos 7 al 11

- Todos los vértices están en el árbol, se ha obtenido un árbol generador del grafo



Algoritmo 5.66. Búsqueda en anchura (BEA).

Sea $G = (V, E)$ el grafo de partida. Se encuentra un árbol con raíz, en un proceso en el que se van construyendo los niveles del árbol. Desde el vértice activo se visitan todos los vértices adyacentes.

Se considera el conjunto de vértices del árbol parcial no visitados, está formado por aquellos vértices del árbol parcial para los cuales no se han detectado sus hijos (los adyacentes del nivel siguiente al que está el vértice).

- Inicio: Se elige de manera arbitraria un vértice como raíz del árbol, v. Se definen los conjuntos
  $W_0 = \{v\}$, conjunto de vértices del árbol parcial en el paso 0
  $F_0 = \varnothing$, conjunto de aristas del árbol parcial en el paso 0
  $L_0 = \{v\}$, como conjunto de vértices del árbol parcial no visitados en el paso 0

- Se supone construidos en el paso (k-1)-ésimo W_{k-1}, el conjunto de vértices del árbol parcial, F_{k-1}, el conjunto de aristas del árbol parcial y L_{k-1}, el conjunto de vértices no visitados.

- Veamos cómo se construyen W_k, F_k y L_k, en el paso k-ésimo:
  Se considera el primer vértice del conjunto de vértices no visitados L_{k-1}. Sea u dicho vértice.
  Se actualiza el conjunto de vértices del árbol parcial añadiendo todos los vértices adyacentes u que no estén en W_{k-1}, esto es,

  $$W_k = W_{k-1} \cup \{\text{adyacentes a } u \text{ que no estén en } W_{k-1}\}$$

  Se actualiza el conjunto de aristas del árbol parcial añadiendo todas las aristas que unen u con los vértices añadidos, esto es,

  $$F_k = F_{k-1} \cup \{\ \{u, w\}, w \text{ vértices añadidos}\}$$

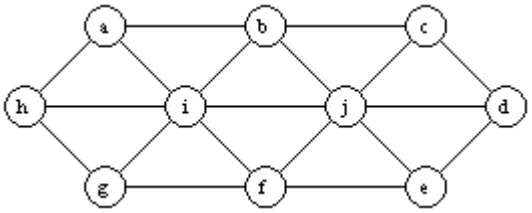
  Se actualiza el conjunto de vértices no visitados borrando u (ya se ha visitado) y añadiendo las nuevos vértices

  $$L_k = (L_{k-1} \setminus \{u\}) \cup \{w, w \text{ vértices añadidos}\}$$

- El proceso termina cuando todos los vértices han sido visitados, esto es,

  $$L_s = \varnothing$$

Ejemplo 5.67. *Usando algoritmo BEA construir un árbol generador del grafo del ejemplo 5.65*



- *Paso* 0: Se elige una raíz, *b*.

$$W_0 = \{b\},\ F_0 = \varnothing,\ L_0 = \{b\}$$

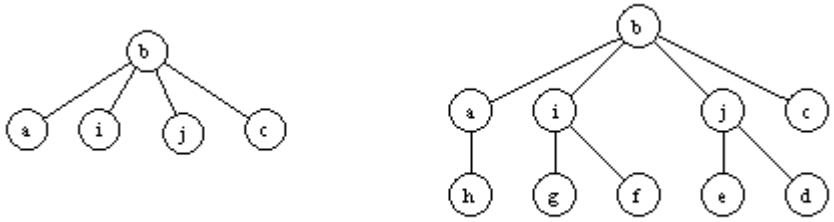
- *Paso* 1: Se consideran los vértices adyacentes a *b*:
  $\{a, i, j, c\}$, se añaden al conjunto de vértices y las aristas que los unen con *b* al conjunto de aristas
  $W_1 = \{b, a, i, j, c,\}$,
  $F_1 = \{\{b, a\}, \{b, i\}, \{b, j\}, \{b, c\}\}$
  $L_1 = \{a, i, j, c,\}$ (borrar *b* y añadir los vértices nuevos, *b* ya ha sido visitado)

- *Paso* 2: Se consideran los vértices adyacentes a *a* que no pertenecen a W_1*:*
  $\{h\}$, se añade al conjunto de vértices y las aristas que los unen con *a* al conjunto de aristas
  $W_2 = \{b, a, i, j, c, h\}$,
  $F_2 = \{\{b, a\}, \{b, i\}, \{b, j\}, \{b, c\}, \{a, h\}\}$
  $L_2 = \{ i, j, c, h\}$ (borrar *a* y añadir los vértices nuevos, *a* ya ha sido visitado)

- *Paso* 3: Se consideran los vértices adyacentes a *i* que no pertenecen a W_2*:*
  $\{g, f\}$, se añaden al conjunto de vértices y las aristas que los unen con *i* al conjunto de aristas
  $W_3 = \{b, a, i, j, c, h, g, f\}$,
  $F_3 = \{\{b, a\}, \{b, i\}, \{b, j\}, \{b, c\}, \{a, h\}, \{i, g\}, \{i, f\}\}$
  $L_3 = \{ j, c, h, g, f\}$ (borrar *i* y añadir los vértices nuevos, *i* ya ha sido visitado)

- *Paso* 4: Se consideran los vértices adyacentes a *j* que no pertenecen a W_3*:*
  $\{e, d\}$ e añaden al conjunto de vértices y las aristas que los unen con *j* al conjunto de aristas
  $W_4 = \{b, a, i, j, c, h, g, f, e, d\}$,
  $F_4 = \{\{b, a\}, \{b, i\}, \{b, j\}, \{b, c\}, \{a, h\}, \{i, g\}, \{i, f\}, \{j, e\}, \{j, d\}\}$
  $L_4 = \{ c, h, g, f , e, d \}$ (borrar *j* y añadir los vértices nuevos, *j* ya ha sido visitado)

- *Paso* 5: Se consideran los vértices adyacentes a *c* que no pertenecen a W_4*:*
  No se pueden añadir ni vértices ni aristas
  $W_5 = \{b, a, i, j, c, h, g, f, e, d\}$,
  $F_5 = \{\{b, a\}, \{b, i\}, \{b, j\}, \{b, c\}, \{a, h\}, \{i, g\}, \{i, f\}, \{j, e\}, \{j, d\}\}$
  $L_5 = \{ h, e, d \}$ (borrar *c*, no hay vértices nuevos para añadir, *c* ya ha sido visitado)
- Como $W_5 = V$, el algoritmo ha terminado.



Un árbol generador del grafo

Definición 5.68. *Un árbol* **generador mínimo** *de un grafo ponderado es un árbol generador tal que la suma de los pesos de sus aristas es la mínima posible.*

Teorema 5.69. *En todo grafo ponderado existe algún árbol generador mínimo.*

Demostración.

Dado G un grafo finito, el número de sus subgrafos es conjunto finito. En particular el número de árboles generadores es finito. Por tanto existe un valor mínimo para la suma de los pesos de las aristas.

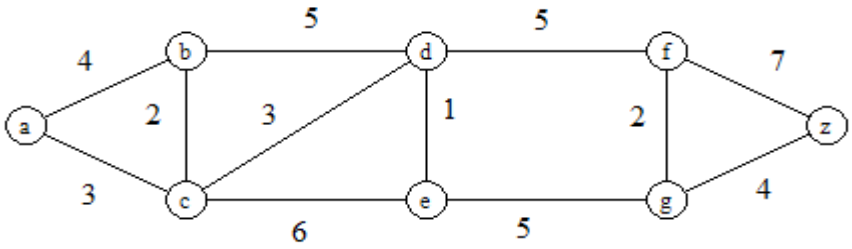
■

Algoritmo 5.70. Algoritmo de Prim

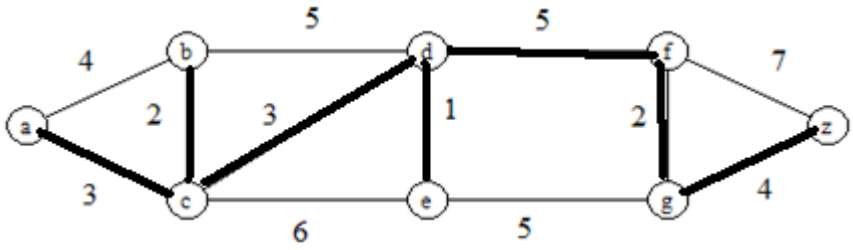
Sea G un grafo conexo con función de peso w. Se va a dar un algoritmo para construir árboles generadores con pesos mínimos, añadiendo, en cada paso, una arista al árbol parcial obtenido.

- Inicio: Se elige una arista con peso mínimo, se a dicha arista.
  Se define $T_0 = \{a\}$, es el conjunto de aristas del árbol parcial.
- Se supone construido en el paso (k-1)-ésimo el árbol parcial T_{k-1}
- Veamos cómo se construye el árbol parcial T_k en el paso k-ésimo:
  Se elige una arista de peso mínimo incidente con un vértice de parcial T_{k-1} que no forme un ciclo, sea e una tal arista. Se actualiza el árbol parcial añadiendo dicha arista
  $$T_k = T_{k-1} \cup \{e\}$$
- El proceso finaliza cuando se haya realizado n-2 veces (el proceso se haya repetido n-1 veces).

Ejemplo 5.71. *Encuentar un árbol generador mínimo para el grafo ponderado:*



- *Paso* 0: Se elige una arista con peso mínimo: sólo hay una posibilidad de elegir: $\{e, d\}$.
  $T_0 = \{\{e, d\}\}$
- *Paso* 1: Se elige una arista con peso mínimo incidente con las aristas de T_0: $\{d, c\}$.
  $T_1 = \{\{e, d\}, \{d, c\}\}$
- *Paso* 2: Se elige una arista con peso mínimo incidente con las aristas de T_1: $\{c, b\}$.
  $T_2 = \{\{e, d\}, \{d, c\}, \{c, b\}\}$
- *Paso* 3: Se elige una arista con peso mínimo incidente con las aristas de T_2: $\{c, a\}$.
  $T_3 = \{\{e, d\}, \{d, c\}, \{c, b\}, \{c, a\}\}$
- *Paso* 4: Se elige una arista con peso mínimo incidente con las aristas de T_3: hay dos posibilidades, por ejemplo $\{d, f\}$.
  $T_4 = \{\{e, d\}, \{d, c\}, \{c, b\}, \{c, a\}, \{d, f\}\}$
- *Paso* 5: Se elige una arista con peso mínimo incidente con las aristas de T_4: $\{f, g\}$.
  $T_5 = \{\{e, d\}, \{d, c\}, \{c, b\}, \{c, a\}, \{d, f\}, \{f, g\}\}$
- *Paso* 6: Se elige una arista con peso mínimo incidente con las aristas de T_5: $\{g, z\}$.
  $T_6 = \{\{e, d\}, \{d, c\}, \{c, b\}, \{c, a\}, \{d, f\}, \{f, g\}, \{g, z\}\}$
- Todos los vértices están en el árbol, se ha obtenido el árbol generador
  $w(e, d) + w(d, c) + w(c, b) + w(c, a) + w(d, f) + w(f, g) + w(g, z) = 20$



Teorema 5.72. *El árbol T obtenido mediante el algoritmo de Prim es un árbol generador mínimo, esto es, para cualquier otro árbol generador U del grafo G se verifica*

$$w(T) \leq w(U),$$

donde $w(T)$ y $w(U)$ denotan la suma de los pesos de las aristas de cada árbol.