Apunte de árboles

Definiciones generales

Un árbol es un grafo G conexo y acíclico.

Todas las aristas de un árbol son críticas (si la eliminás del grafo, ya no es conexo).

Un nodo $v \in V(G)$ es hoja $\iff$ $\text{grado}(v) = 1$

Un **bosque** es un conjunto de árboles, donde cada componente conexa es árbol.

Sea $G = (V, E)$ un árbol, es equivalente decir:

1. G es árbol
2. G es acíclico, y si agregamos una arista $e \notin E$, entonces $G + e$ tendrá exactamente un ciclo, y ese ciclo contiene a e.
3. Existe un solo camino entre cada par de nodos $v \in V$
4. G es acíclico y $|E| = |V| - 1$
5. G es conexo y $|E| = |V| - 1$

Lema 1:
- Sea $G = (V, E)$ un grafo conexo, $e \in E$:
- $G - e$ es conexo $\iff$ e existe en un ciclo de G

Lema 2:
- Todo arbol con $|V| \geq 2$ tiene al menos 2 hojas.

Lema 3:
- Si $G = (V, E)$ es árbol, entonces $|E| = |V| - 1$

Corolario 1:
- Si $G = (V, E)$ es un bosque de C componentes conexas, entonces $|E| = |V| - C$

Corolario 2:
- Dado $G = (V, E)$ con C componentes conexas, $|E| \geq |V| - C$

Árboles enraizados

Sea $G = (V, E)$ un grafo conexo y un nodo $v \in V$ al que llamaremos raiz.

Un **árbol enraizado** es un árbol $T = (V_T, E_T)$ tal que:
- $V_T \subseteq V$, $E_T \subseteq E$
- T tiene raiz v
- Existe un nodo raiz $v \in V_T$ tal que $\forall w \in V_T$, existe un único camino en T de v a w

Tenemos una función padre : $(V_T \setminus v) \to V_T$ tal que:

- $\text{padre}(w)$ es el nodo anterior a w en el camino único $v \rightsquigarrow w$ en T

Árbol generador

Sea $G = (V, E)$ un grafo conexo.

Un **árbol generador** de G es un subgrafo $T = (V, E_T)$, es decir, tiene los mismos nodos que G y:
- $E_T \subseteq E$ tal que $|E| \geq |E_T|$
- T es árbol
- T tiene todos los nodos de G

Árbol generador mínimo/máximo

Sea $G = (V, E)$ un grafo pesado, $c : E \to \mathbb{R}$ una función de peso.

Un **árbol generador mínimo** (AGmin) es un árbol generador $T_{\min} = (V, E_{\min})$ que minimiza el peso total de sus aristas, la idea intuitiva es que sea el peso mínimo posible tal que sea conexo y mantenga los nodos originales.

En otras palabras, es el que minimo $c(T_{\min}) = \sum_{e \in E_{\min}} c(e)$ posible entre todos los árboles generadores de G.

Un **árbol generador máximo** (AGMáx) es análogo al AGMin, pero maximizando.

$T_{\max} = (V, E_{\max})$ tal que maximiza $c(T_{\max}) = \sum_{e \in E_{\max}} c(e)$.

BFS

Input:

- Un grafo $G = (V, E)$
- un nodo raiz $v \in V$

Output:

- Función de distancias $\delta : V \to \mathbb{N} \cup \{\infty\}$ donde $\delta(w)$ es la mínima cantidad de aristas entre v y w.
- Árbol enraizado T con raiz v, tal que todo $w \in V$ es alcanzable desde v, y el camino $v \rightsquigarrow w$ en T es mínimo.

Definición de v-geodésico:

- Sea $\operatorname{dist}(v, w)$ la distancia mínima de v a w en G, entonces el camino $v \rightsquigarrow w$ en el árbol generado por $\text{BFS}(G, v)$ tiene longitud $\operatorname{dist}(v, w)$.

Esto quiere decir que un árbol v-geodésico es un árbol enraizado en un nodo v, tal que el camino desde v a cualquier vértice w es el más corto posible en cantidad de aristas dentro del grafo original G.

Invariante:

- Si $\delta(w) = k$, entonces no existe camino $v \rightsquigarrow w$ con menos de k aristas.
- Genera un árbol v-geodésico

Complejidad:

- Temporal: $O(|V| + |E|)$
- Espacial: $O(|V|)$

Usos:

- Ver si G es conexo: G es conexo $\iff \text{BFS}(G, v)$ visita todos los nodos de V
- Construir árboles v-geodésicos. $\delta(u) = \delta(\text{padre}(u)) + 1$
- Calcular el diámetro de un árbol:
- ‣ $\text{BFS}(G, u) \to$ ver el nodo v más lejano.
- ‣ $\text{BFS}(G, v) \to$ ver el nodo w más lejano.
- ‣ $\operatorname{dist}(v, w)$ es el diámetro. Esto es $O(|V|)$

Dijkstra

Sea $G = (V, E)$ un grafo con **pesos positivos** en las aristas.

Dijkstra encuenta los caminos mínimos de un nodo raiz $v \in V$ a todo nodo $w \in V$

$\mathrm{dist}(v, w) = \infty \iff v$ está en una componente conexa distinta a la de w.

Input:
- $G = (V, E)$ un grafo.
- una función de peso $c : E \to \mathbb{R}_{\geq 0}$
- Un nodo raiz $v \in V$

Output:
- $\forall w \in V$, la distancia mínima $\mathrm{dist}(w)$ desde v hasta w, o sea, $\mathrm{dist}(w) = \delta(v, w)$ el peso mínimo del camino $v \rightsquigarrow w$.
- Un árbol de caminos mínimos definido por una función padre que nos permite reconstruir los caminos mínimos.

Invariante:
- Sea S el conjunto de nodos ya extraidos y la raiz $v \in V$, en cada iteración vale $\forall w \in S :$ $\mathrm{dist}(w) = \delta(v, w)$, donde $\delta(v, w)$ es la distancia mínima de v a w en G.
- Además sabemos que $\forall u \in V \setminus S : \mathrm{dist}(u) \geq \delta(v, u)$

Complejidad
- Con heap binario: $O((|V| + |E|) \log|V|)$
- Con Fibonacci heap: $O(|E| + |V| \log|V|)$
- Con cola no ordenada: $O(|V|^2)$

Usos:
- Al igual que BFS, genera un árbol v-geodésico.
- *st*-eficiencia:
- ‣ Dijkstra$(G, s) \to \delta_s$
- ‣ Invertimos toda $(u, w) \in E$ tal que quedan $(w, u) \in E$
- ‣ Dijkstra$(G, t) \to \delta_t$
- ‣ Vale $\delta(s, t) = \delta(s, u) + c(u, w) + \delta(w, t)$

Prim

Input:
- Un grafo $G = (V, E)$ no dirigido con pesos positivos.
- Un nodo raiz $v \in V$

Output:
- Un árbol generador mínimo $T = (V, E_T) \subseteq G$

Invariante:
- En cada iteración, T es un subárbol conexo y mínimo entre todos los $w \in V_T$
- La arista elegida para expandir T es la de menor peso saliente del nodo actual.

Complejidad:
- Con heap binario: $O(|E| \log|V|)$
- Con Fibonacci heap: $O(|E| + |V| \log|V|)$

Usos:
- Obtener un árbol generador mínimo
- Obtener un árbol generador máximo, tal que dado $G = (V, E), v \in V$:
- ‣ $\forall e \in E :: c(e) = -c(e)$ invertimos los pesos.

- ‣ $\text{Prim}(G, v) \to T = (V, E_T)$
- ‣ $\forall e \in E_T :: c(e) = -c(e)$ para recuperar los pesos originales.

Kruskal

Input:
- Un grafo $G = (V, E)$ no dirigido y pesado

Output:
- Un **bosque** generador mínimo de G. $T = (V, E_T)$

Invariante:
- El conjunto actual de aristas forma un bosque generador mínimo.

Complejidad:
- Heapsort + DSU: $O(|E| \log|E|)$

Usos:
- Dar un bosque generador mínimo de un grafo G.
- Dar un bosque generador máximo, misma estrategia que Prim.

MaxiMin/MiniMax

MiniMax:
Dado $G = (V, E)$ y $\mathcal{T}(G)$ un conjunto de árboles generadores de G.

El MiniMax es el árbol $T \in \mathcal{T}(G)$ donde su arista más pesada es de menor peso posible. Es decir **Mini**-miza el **Máx**-imo.

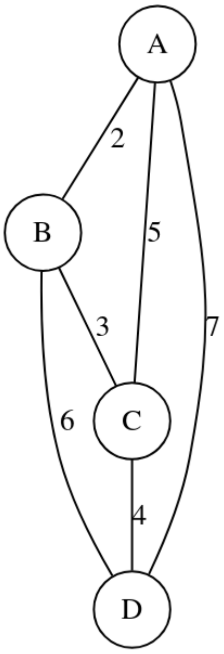
Importante: Todo árbol generador mínimo es MiniMax

MaxiMin:
Misma intuición que en Minimax, con la diferencia de que MaxiMin lo que hace es **Maxi**-mizar el **Min**-imo, o sea, es el árbol $T(G)$ tal que su arista menos pesada sea lo más pesada posible.

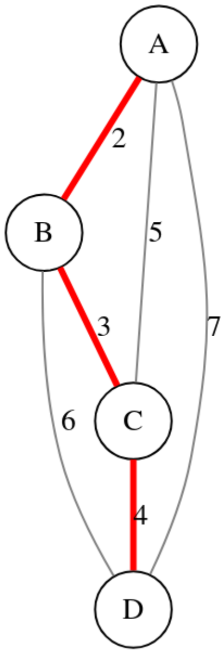
Importante: Todo árbol generador máximo es MaxiMin

Ejemplo:

Grafo original

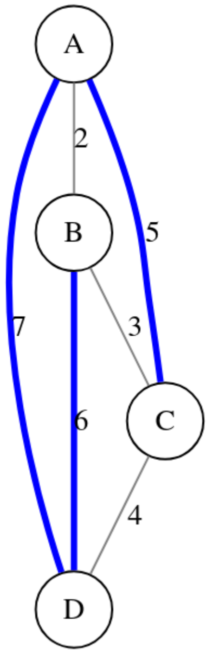


MiniMax



Máximo: 4

MaxiMin



Mínimo: 5