Ejercicio 1 (PrioriKruskal)

El algoritmo de Kruskal (resp. Prim) con orden de selección es una variante del algoritmo de Kruskal (resp. Prim) donde a cada arista e se le asigna una prioridad $q(e)$ además de su peso $p(e)$. Luego, si en alguna iteración del algoritmo de Kruskal (resp. Prim) hay más de una arista posible para ser agregada, entre esas opciones se elije alguna de mínima prioridad.

a) Demostrar que para todo árbol generador mínimo T de G, si las prioridades de selección están definidas por la función:

$$q_T = \begin{cases} 0 & \text{si} \quad e \in T \\ 1 & \text{si} \quad e \notin T \end{cases}$$

entonces se obtiene T como resultado del algoritmo de Kruskal (resp. Prim) con órden de selección ejecutado sobre G (resp. cualquiera sea el vértice inicial en el caso de Prim).

b) Usando el inciso anterior, demostrar que si los pesos de G son todos distintos, entonces G tiene un único árbol generador mínimo.

♠

Solución.

a) Probemos primero que el algoritmo de Kruskal encuentra todos los árboles generadores mínimos, y luego lo mismo para el algoritmo de Prim. Llamemos $w(e) = (p(e), q_T(e))$, donde ordenamos las tuplas por órden lexicográfico. Es decir, $w(e) \leq w(e') \Leftrightarrow p(e) < p(e') \vee (p(e) = p(e') \wedge q_T(e) < q_T(e'))$.

Teorema 1

El algoritmo de Kruskal con selección devuelve T.

♡

Demostración. Por «Calamardo» en Telegram. Sea K el árbol generador que devuelve el algoritmo de Kruskal con selección. Asumimos que $K \neq T$. Luego, existe una primer arista e que el algoritmo agregó a K que no está en T. Consideremos $T' = T \cup \{e\}$. Este subgrafo de G contiene un único ciclo C, con $e \in C$. Como K es un árbol, $C \nsubseteq K$, puesto que un árbol no tiene ciclos. Luego existe alguna arista $f \in C$ tal que $f \notin K$. Como $f \notin K$ pero $e \in K$, tenemos que $f \neq e$. Como $f \neq e$, y $f \in C \subseteq T + e$, entonces $f \in T$. Consideremos entonces $T'' = T' \setminus \{f\} = (T \cup \{e\}) \setminus \{f\}$. Esto es un árbol generador, puesto que rompimos el único ciclo, C, que había en T'. Vemos que $p(T'') = p(T) + p(e) - p(f)$. Como T es un árbol generador mínimo, debemos tener $p(T'') \geq p(T)$, con lo cual $p(e) \geq p(f)$.

Como $e \notin T$, y $f \in T$, tenemos $q_T(f) = 0, q_T(e) = 1$. Luego, $w(f) < w(e)$, y el algoritmo de Kruskal con selección vio primero a f. Sea j el paso en el que el algoritmo vio (y decidió agregar) a e, e i el paso en el que el algoritmo vio a f. Tenemos que $j > i$. Dada una iteración t, sea F_t el conjunto de aristas que el algoritmo agregó al comenzar la iteración t. Tenemos que $F_t \subseteq F_{t+r}$ para todo $r \geq 1$. Como $j > i$, tenemos que $F_i \subseteq F_j$. El algoritmo de Kruskal con selección va a agregar a una arista x si y sólo si, al verla en un paso k, x no genera ciclos con F_k.

Como e es la primer arista que el algoritmo agrega que no está en T, entonces $F_j \subseteq T$ (y $F_{j+1} \nsubseteq T$). Como $f \in T$, y T es un árbol, f no genera ciclos con nadie en T, menos aún con aristas en F_j, y como $F_i \subseteq F_j$, menos aún va a generar ciclos con F_i. Luego, al ver a f en el paso i, el algoritmo pudo haber agregado a f, y no lo hizo, pero eso no puede pasar.

Luego, e no existe, y $K = T$. □

Teorema 2

El algoritmo de Prim con selección devuelve T.

♡

Demostración. Sea P el árbol generador que construye el algoritmo de Prim con selección. Recordemos que este algoritmo mantiene una partición $\left(S_i, \overline{S_i}\right)$ de $V(G)$, y un conjunto de aristas F_i, donde F_i genera S_i (la componente conexa). Inicialmente $S_0 = \{v_0\}$ para algún v_0 arbitrario, y $F_0 = \emptyset$. En cada iteración, el algoritmo busca la arista de mínimo w que cruza la partición. Si esta arista es $\{u, v\}$, con $u \in S_i$ y $v \notin S_i$, tenemos $S_{i+1} = S_i \cup \{v\}$, y $F_{i+1} = F_i \cup \{e\}$. El algoritmo se detiene cuando $S_{n-1} = V(G)$, y devuelve F_{n-1}.

Asumimos que $P \neq T$. Luego, existe una primer arista e que el algoritmo agregó a la componente conexa, que no está en T. Sean $\{u, v\} = e$ los extremos de e. Asumimos sin pérdida de generalidad que u es el extremo que está en S_i, y v el extremo que está en $\overline{S_i}$. Entonces, tenemos $S_{i+1} = S_i \cup \{v\}$, $F_{i+1} = F_i \cup \{e\}$.

Como T es un árbol generador, pero $e \notin T$, existe un (único) camino Q en T entre u y v. Como u y v están en diferentes lados de la partición $\left(S_i, \overline{S_i}\right)$, entonces en Q existe alguna arista f que cruza la partición. Consideremos entonces $T' = (T \setminus \{f\}) \cup \{e\}$. Esto es un árbol generador, por haber separado a T en dos pedazos cuando sacamos f, uno que genera S y otro que genera $\overline{S}$, y luego haberlos juntado al agregar e. Como T es un árbol generador mínimo, tenemos que $p(T) \leq p(T') = p(T) - p(f) + p(e)$, con lo cual $p(e) \geq p(f)$.

Como f está en T, tenemos que $q_T(f) = 0$. Como e no está en T, tenemos que $q_T(e) = 1$. Como $p(f) \leq p(e)$, y $q_T(f) < q_t(e)$, tenemos que $w(f) < w(e)$. Como ambas aristas cruzan la partición $\left(S_i, \overline{S_i}\right)$, al considerar la arista e, el algoritmo también consideró la arista f. Pero entonces, el algoritmo no eligió a la arista que cruza la partición, de menor w. Esto no puede suceder, por lo tanto e no existe, y luego $P = T$. □

b) Lo que nos dice el ejercicio anterior es que para todo árbol generador mínimo T, ambos el algoritmo de Kruskal y el algoritmo de Prim tienen ejecuciones que devuelven T, dependiendo sólo de cómo se desempata cuando encuentran aristas del mismo peso, en cada ejecución. Es decir, el conjunto de resultados posibles del algoritmo de Kruskal sobre G (resp. Prim), es igual al conjunto de árboles generadores mínimos de G

Luego, si para un grafo G en ningún momento hace falta desempatar, porque todos los pesos de las aristas de G son distintos, entonces en toda ejecución el resultado del algoritmo de Kruskal (resp. Prim) es único.

Como el conjunto de árboles generadores mínimos es igual al conjunto de resultados de correr estos algoritmos, y este último tiene un sólo elemento cuando los pesos de G son todos distintos, entonces G tiene un único árbol generador mínimo.