Algoritmuselmélet

Minimális feszítőfák

Katona Gyula Y.

Számítástudományi és Információelméleti Tanszék
Budapesti Műszaki és Gazdaságtudományi Egyetem

Minimális költségű feszítőfák

Most egyszerű, irányítatlan gráfokkal foglalkozunk.

Definíció (minimális költségű feszítőfa)

Legyen $G = (V, E)$ egy összefüggő gráf. A G gráf egy körmentes összefüggő $F = (V, E')$ részgráfja a gráf egy feszítőfája. Legyen továbbá az éleken értelmezve egy $c : E \to \mathbb{R}$ súlyfüggvény. Ekkor a G gráf egy F feszítőfája minimális költségű, ha költsége (a benne szereplő élek súlyainak összege) minimális G összes feszítőfáját tekintve.

Probléma

Adott egy $G = (V, E)$ összefüggő irányítatlan gráf, és az élein értelmezett $c : E \to \mathbb{R}$ súlyfüggvény. Határozzuk meg a G egy minimális költségű feszítőfáját.

Például: villamosvezetékek kiépítése.

Kruskal algoritmusa

A $V(G)$ csúcshalmazon egy F körmentes részgráfot építünk élek hozzávételével:

- Kezdetben $E(F) = \emptyset$.
- A még nem vizsgált élek közül kiválasztjuk a legkisebb súlyút (pontosabban az egyik ilyet).
- Ha ez nem alkot kört a már F-beli élekkel, akkor hozzávesszük F-hez. (Ha kört alkot nem vesszük hozzá, eldobjuk.)
- Ha F-nek már $n-1$ éle van, akkor véget ér az algoritmus, F egy minimális súlyú feszítőfa.

F mindvégig egy erdőt fog alkotni, azaz minden összefüggő komponense egy fa. Amikor új élet veszünk hozzá, az két komponenst köt össze.

Tétel (BSZ2-n volt)

A Kruskal-algoritmus eredményeként végül F a G gráf egy minimális költségű feszítőfájának éleit tartalmazza.

A Kruskal algoritmus implementációi

Jobban járunk, ha a gráf éllistával van megadva, csak ezt az esetet vizsgáljuk.

Minden körben kétféle feladat van:

1. Melyik a legkisebb súlyú még nem vizsgált él?
2. Ez kört alkot-e F eddigi éleivel?

A Kruskal algoritmus implementációi

Megjegyzés

Egyszerű gráf esetén $O(\log m) \subseteq O(\log n)$.

Bizonyítás.

$m \leq \frac{n(n-1)}{2} \leq n^2$. Így $\log m \leq \log n^2 = 2 \log n$. □

A legkisebb súlyú, még nem vizsgált él kiválasztása:

- 1. módszer: Először rendezzük az éleket pl. összefésüléses rendezéssel $O(m \log m)$ lépésben, így a következő legkisebb kiválasztása $O(1)$ lépés. Ezt m alkalommal kell megtenni. Lépésszám: $O(m \log m) + O(m) \subseteq O(m \log m) \subseteq O(m \log n)$.
- 2. módszer: Építsünk kupacot az élekből $O(m)$ lépésben, így a következő kiválasztása $O(\log m)$. Ezt m alkalommal kell megtenni. Lépésszám: $O(m) + O(m \log m) \subseteq O(m \log n)$.

A Kruskal algoritmus implementációi

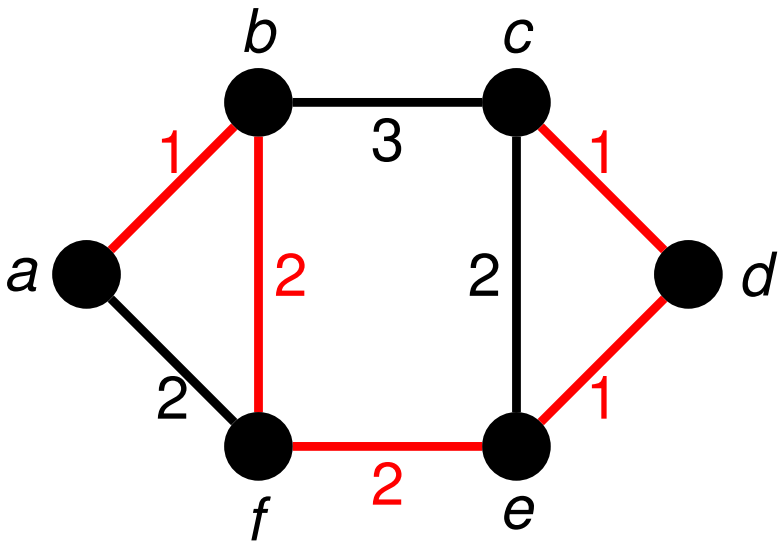
Kört alkot-e a következő él F eddigi éleivel?

Ötlet: Tartsuk nyilván F komponenseit, azaz minden v csúcsra jegyezzük fel, hogy melyik komponensben van. A komponens „neve” legyen a „legkisebb sorszámú” csúcsa. (Az éllistás megadásban a csúcsok fel vannak sorolva egy tömbben, e szerint kapnak sorszámot.) Ezt a KOMPONENS[v] tömbben tároljuk.

1. Kört alkot az (u, v) él F éleivel?
   Ha KOMPONENS[u] $\neq$ KOMPONENS[v], akkor nem, különben igen.
2. Ha nem alkot kört, akkor hozzávesszük F-hez. Ezzel viszont módosulnak a komponensek.
3. Ha KOMPONENS[u] $= x$ sorszáma kisebb, mint KOMPONENS[v] $= y$ sorszáma, akkor végigmegyek a KOMPONENS tömbön és minden y-t x-re cserélek.
4. Az algoritmus végén a KOMPONENS tömb minden eleme a legkisebb sorszámú csúcs lesz.

Példa

A pontok sorrendje legyen a, b, c, d, e, f.



a	b	c	d	e	f	új él
a	b	c	d	e	f	$\{a, b\}$
a	a	c	d	e	f	$\{c, d\}$
a	a	c	c	e	f	$\{d, e\}$
a	a	c	c	c	f	$\{b, f\}$
a	a	c	c	c	a	$\{f, e\}$
a	a	a	a	a	a	

A Kruskal lépésszáma

1. KOMPONENS[u] $\neq$ KOMPONENS[v] igaz-e? Lépésszám: $O(1)$
2. Ha nem alkot kört, akkor hozzávesszük F-hez. Lépésszám: $O(1)$.
3. KOMPONENS tömb frissítése. Lépésszám: $O(n)$.

Az ellenőrzést m-szer kell elvégezni, de mivel csak $n-1$ élet veszünk hozzá F-hez, így a frissítést csak $n-1$-szer:
Lépésszám: $O(m+n^2) \subseteq O(n^2)$.

A teljes algoritmus lépésszáma a kezdeti rendezéssel együtt:
$O(m \log n) + O(n^2)$.

Ha $m < n^2/\log n$, akkor a legtöbb idő a KOMPONENS tömb frissítése.
Ezt egy alkalmas, új adatstukatúrával felgyorsíthatjuk.

Az UNIÓ-HOLVAN adatszerkezet

Legyen adott egy véges S halmaz. (Mi a ponthalmazra alkalmazzuk.)
Ennek egy partícióját szeretnénk tárolni $\rightarrow U_1, \ldots, U_k \subseteq S$.
(A komponensek ponthalmazai.)

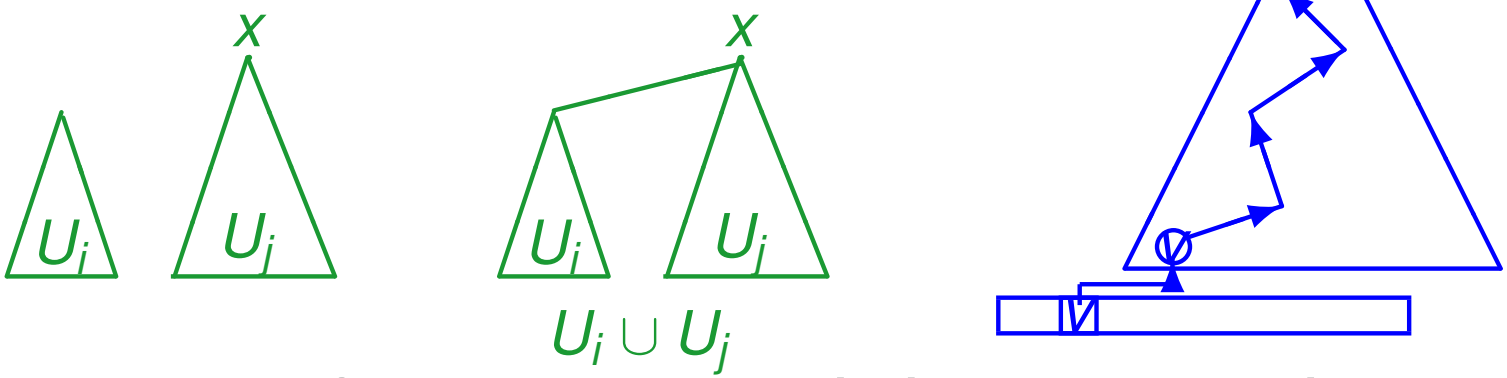
- Adott egy n elemű S halmaz, és ennek bizonyos $U_1, \ldots, U_m$ részhalmazai, melyekre $U_i \cap U_j = \emptyset$ $(i \neq j)$ és $U_1 \cup \ldots \cup U_m = S$ (vagyis az U_j részhalmazok S egy partícióját adják).
- *Műveletek:*
  UNIÓ$(U_i, U_j) = (\{U_1, \ldots, U_m\} \cup \{U_i \cup U_j\}) \setminus \{U_i, U_j\}$
  (az U_i, U_j halmazokat $U_i \cup U_j$ helyettesíti).
  HOLVAN(v) eredménye (itt $v \in S$) annak az U_i halmaznak a neve, amelynek v eleme.

Kruskalban:
Annak megvizsgálása, hogy bevegyük-e az (u, v) élet:
Ha HOLVAN$(u) \neq$ HOLVAN(v), akkor bevesszük, különben nem.
Ha bevesszük $\Longrightarrow$ UNIÓ(HOLVAN(u), HOLVAN(v))

Implementáció fákkal

$U_j \rightarrow$ gyökeres, felfelé irányított fa
U_j elemeit a fa csúcsaiban tároljuk, egy szülőmutatóval.
Egy részhalmaz *neve* legyen az őt ábrázoló fa gyökere. A gyökérben nyilvántartjuk még a fa méretét (azaz a csúcsok számát) is.

- **UNIÓ:** $U_i \cup U_j$ fáját a következőképpen készítjük el:
  Tegyük fel, hogy $|U_i| \leq |U_j|$. Ekkor az U_j fa x gyökeréhez gyermekként hozzákapcsoljuk U_i gyökerét.



- **HOLVAN:** A $v \in S$ elemet tartalmazó részhalmaz nevét, azaz a megfelelő fa gyökerét a szülőkhöz menő mutatók végigkövetésével találhatjuk meg.

Az UNIÓ hívásakor az U_i és U_j halmazok a gyökerükkel adottak
$\Longrightarrow$ költség: $O(1)$
Ha egy v csúcs új gyökér alá kerül, akkor egy szinttel lesz távolabb a gyökértől, míg az új fájának a mérete legalább az eredeti duplájára változik.
$\Longrightarrow$ Egy csúcs legfeljebb $\log_2 n$-szer kerülhet új gyökér alá (különben a fa csúcsszáma nagyobb lenne, mint n).
$\Longrightarrow$ szintszám $\leq \log_2 n$.
$\Longrightarrow$ HOLVAN költsége $O(\log n)$.

Tétel

A Kruskal-algoritmus költsége $O(m \log n)$. □

Bizonyítás.

Először az élek rendezése $O(m \log n)$ lépés. Utána $2m$ HOLVAN, és $n-1$ UNIÓ műveletet jelent. Ezek időigénye
$O(m \log n) + O(m \log n) + O(n) \subseteq O(m \log n)$. □