0.1 Általános gráfbejárás & BFS

A gráfbejárási algoritmus az inputgráf csúcsait és éleit fedezi fel. Minden csúcs az eléretlen → elért → befejezett állapotokat veszi fel. A bejárás akkor ér véget, amint minden csúcs befejezetté vált.

1. Van elért csúcs. Választunk egyet, mondjuk u-t.

(1a) Ha van olyan uv él, amire v eléretlen, akkor v elértté válik.

(1b) Ha nincs ilyen uv él, akkor u befejezetté válik.

2. Nincs elért csúcs.

(2a) Ha van eléretlen u csúcs, akkor u-t elértté tesszük.

(2b) Ha nincs eléretlen csúcs (azaz $\forall$ csúcs fejezett), akkor END.

Szélességi bejárás (BFS) szabálya:

Az 1. esetben mindeg a legkorábban elért u-t választjuk.

Input: $G = (V, E)$ (ir/ir.tatlan) gráf, ($v \in V$ gyökérpont[1]).

Output: (1) A csúcsok elérési és befejezési sorrendje. (2) Az élek osztályozása:

faél: Olyan él, ami mentén egy csúcs elértté vált.

uv **előreél:** nem faél, de u-ból v-be faélekből irányított út vezet.

uv **visszaél:** v-ből u-ba faélekből irányított út vezet.

keresztél: minden más él (u és v közt nincs leszármazott viszony).

(3) A **bejárás fája:** a faélek alkotta részgráf. (A bejárás fája valójában egy gyökereiből kifelé irányított erdő.)

Megf: Irányítatlan esetben az előreél és a visszaél ugyanazt jelenti.

Terminológia: Ha a bejárás fájában u-ból v-be irányított út vezet, akkor u a v őse és v az u leszármazottja. A faél és az előreél tehát ősből leszármazottba, a visszaél leszármazottból ősbe vezet.

0.2 A BFS tulajdonságai

Nézzük meg egy **irányított** gráf BFS bejárását is.

Állítás: Tfh $G = (V, E)$ BFS bejárása után a csúcsok elérési sorrendje $v_1, v_2, \ldots, v_n$. Ekkor az alábbiak teljesülnek.

(1) Ha $i < j$, akkor v_i-t hamarabb fejezük be, mint v_j-t, továbbá v_i gyerekei megleőzik v_j gyerekeit az elérési sorrendben.

Biz: A v_i-t befejezésének pillanatában v_i minden gyereke elért, de v_j-nek még egy gyereke sem az. Ezért v_j gyerekeit a v_i csúcs befejezése után érjük el, majd ezt követően fejezzük be v_j-t. □

(2) **Az elérési és befejezési sorrend (BFS esetén) megegyezik.**

Biz: Ha v_i-t korábban érjük el, mint v_j-t, akkor (1) miatt v_i-t korábban is fejezzük be v_j-nél. Ezért bármely két csúcs sorrendje ugyanaz az elérési sorrendben mint befejezési sorrendben. Tehát az elérési sorrendnek meg kell egyeznie a befejezési sorrenddel. □

(3) **Gréfél nem ugorhat át falét:** ha $k < i < j \leq l$ és $v_i v_j$ faél, akkor $v_k v_l$ nem lehet gráfél.

Biz: Ha $v_k v_l \in E(G)$, akkor v_l szülője v_k vagy egy v_k-t megelőző csúcs. (1) miatt v_j szülője sem következhet v_k után, vagyis v_i nem lehet v_j szülője.

(4) **Nincs előreél.** (Irányítatlan eset: csak faél és keresztél van.)

Biz: Indirekt: ha $v_i v_j$ előreél lenne, akkor v_i-ből v_j-be irányított út vezetne a BFS-fában, és $v_i v_j$ ennek a faélekből álló útnak az utolsó élét átugraná. □

(5) Ha a BFS-fában k-élű irányított út vezet u-ból v-be, akkor G-ben nincs k-nál kevesebb élű uv-út.

[1] A gyökérben kezdetben elért állapotú, ezért kivétel az általános szabály alól.

Biz: Ha lenne a BFS fa-beli útnál kevesebb elű út G-ben, akkor lenne olyan gráfél, ami faélt ugrik át. $\square$

(6) **A BFS-fa egy legrövidebb utak fája:** a BFS-fa v_1 gyökeréből bármely v_i csúcsba vezető faút a G egy legkevesebb élű $v_1 v_i$-útja.

0.3 Legrövidebb utak

Def: Adott G (ir) gráf és $l : E(G) \to \mathbb{R}$ hosszfüggvény esetén egy P út hossza a P éleinek összhossza: $l(P) = \sum_{e \in E(P)} l(e)$.

Az u és v csúcsok távolsága a legrövidebb uv-út hossza: $dist_l(u, v) := \min\{l(P) : P\ uv\text{-út}\}$ ($\nexists uv$-út$\Rightarrow$ $dist_l(u, v) = \infty$.) Az l hosszfüggvénye nemnegatív, ha $l(e) \geq 0$ teljesül minden e élre. Az l hosszvüggvény konzervatív, ha G-ben $\nexists$ negatív összhosszú ir. kör.

Cél: Legrövidebb út keresése irányított/irányítatlan gráfban.

Megf: Ha $l(e) = 1$ a G minden e élére, akkor $l(P)$ a P élszáma. Ezért a BFS-fa minden gyökérből elérhető csúcsba tartalmaz egy legrövidebb utat a gyökérből elérhető csúcsba tartalmaz egy legrövidebb utat a gyökérből, azaz a szélességi bejárás tekinthető egy legrövidebb utat kereső algoritmusnak is.

Def: Adott G (ir) gráf, $l : E(G) \to \mathbb{R}$ hosszfüggvény és $r \in V(G)$. (r, l)-felső becslés olyan $f : V(G) \to \mathbb{R}$ függvény, ami felülről becsli minden csúcs r-től mért távolságát: $dist_l(r, v) \geq f(v) \forall v \in V(G)$.

Triviális (r, l)-felső becslés: $f(v) = \begin{cases} 0 & v = r \\ \infty & v \neq r \end{cases}$

Pontos (r, l)-felső becslés: $f(v) = dist_l(r, l)\ \forall v \in V(G)$.

0.4 Az elméleti javítás

Def: Tfh f egy (r, l)-felső becslés és $uv \in E(G)$. Az f uv-elméleti javítása az az f', amire

$$f'(z) = \begin{cases} f(z) & z \neq v \\ min\{f(v), f(u) + l(uv)\} & z = v \end{cases}$$

Megf: Tfh az $l : E(G) \to \mathbb{R}$ hosszfüggvény konzervatív és $f(r) = 0$.

Ekkor (1) Az $f(r, l)$-felső becslés élmenti javítása mindig (r, l)-felső becslést ad.

Biz: Azt kell megmutatni, hogy van olyan rv-út, aminek a hossza legfeljebb $f(u)+l(uv)$. Ha egy legrövidebb ru-utat kiegészítünk az uv éllel, akkor olyan rv-élsorozatot kapunk, aminek az összhossza $dist_l(r, u) + l(uv) \leq f(u) + l(uv)$. „Könnyen" látható, hogy az élhosszfüggvény konzervativitása miatt ha van x összhosszúságú rv-élsorozat, akkor van legfeljebb x összhosszúságú rv-út is. Ezek szerint van legfeljebb $f(u) + l(u, v)$ hosszúságú uv-út is, azaz az érdemi élmenti javítás után szintén (r, l)-felső becslést kapunk. $\square$

(2) $f(r, l)$-felső becslés (pontosan) $\iff$ (f-en $\nexists$ érdemi élmenti javítás).

Biz: $\Rightarrow$: Ha f pontos, akkor biztosan nincs rajta érdemi élmenti javítás: ha volna, akkor egy felső becslés a pontos érték alá csökkenne, így az élmenti javítás nem (r, l)-felső becslést eredményezne. $\Leftarrow$: Legyen $v \in\ V(G)$ tetsz, és legyen P egy legrövidebbb rv-út. A P egyik éle mentén sincs érdemi élmenti javítás, ezért P minden u csúcsára pontos a felső becslés: $f(u) = dist_l(r, u)$. Ez igaz az út utolsó csúcsára, a tetszőlegesen választott v-re is. $\square$

Köv: Adott G, konzervatív l és $r \in V(G)$ esetén ha kiindulunk a triviális (r, l)-felső becslésből, és addig végzünk émj-kat, amíg lehet, akkor a végén megkapjuk minden csúcs r-től való távolságát.

Itt a jegyzet 17. oldaláról az utolsó kettő pont hiányzik, mivel nem tudom, hogy mennyire lényegesek.

Def: Tfh f egy (r,l)-felső becslés és $uv \in E(G)$. Az f uv-élmenti javítása az az f', amire $f'(z) = \begin{cases} f(z) & z \neq v \\ min\{f(v), f(u) + l(uv)\} & z = v \end{cases}$

Megf: Tfh az $l : E(G) \to \mathbb{R}$ hosszfüggvény konzervatív és $f(r) = 0$. Ekkor (1) Az $f(f,l)$-felső becslés élmenti javítása mindig (r,l)-felső becslést ad. (2) $f(r,l)$-felső becslés (pontosan) $\Leftrightarrow$ (f-en $\nexists$ érdemi élmenti javítás).

Dijkstra-algoritmus: Input: $G = (V,E), l : E \to \mathbb{R}_+, r \in V$. Output: $dist_l(r,v) \forall v \in V$ Működés: $U_0 := \emptyset$, f_0 a triviális. (r,l)-felső becslés.

Az i-dik fázis:

1. Legyen $U_i := U_{i-1} \cup \{u_i\}$, ahol u_i olyan csúcs a $V \setminus U_{i-1}$ halmazból, amelyre $f_{i-1}(v)$ minimális.

2. f_i : f_{i-1} élmenti javítása minden U_i-ből kivezető $u_i x$ élen. Output: $f_{|V|}$. Megjelöljük a végső $f_{|V|}(V)$ értékeket beállító éleket.

0.5 Dijkstra, egy példán

Dijkstra-algoritmus: Input: $G = (V,E), l : E \to \mathbb{R}_+, r \in V$. Output: $dist_l(r,v) \forall v \in V$ Működés: $U_0 := \emptyset$, f_0 a triviális. (r,l)-felső becslés.

Az i-dik fázis:

1. Legyen $U_i := U_{i-1} \cup \{u_i\}$, ahol u_i olyan csúcs a $V \setminus U_{i-1}$ halmazból, amelyre $f_{i-1}(v)$ minimális.

2. f_i : f_{i-1} élmenti javítása minden U_i-ből kivezető $u_i x$ élen. Output: $f_{|V|}$. Megjelöljük a végső $f_{|V|}(V)$ értékeket beállító éleket.

Megf: Ha a v-be vezet megjelölt él, akkor vezet r-ből v-be megjelölt éleken út, és ennek hozza megegyezik $f_{|V|}(v)$-vel.

Biz: $f_{|V|}(r) = 0$, és a megjelölt élek mentén haladva az $f_{|V|}$ érték az élhosszal növekszik. $\square$

Köv: Ha a Dijsktra-algoritmus helyes, akkor az algoritmus végén a megjelölt élek egy legrövidebb utak fáját alkotják r gyökérrel.

0.6 Dijkstra helyessége

Megf: Tfh $u_1, u_2, \ldots, u_n$ a G csúcsainak sorrendje a Dijkstra-algoritmus végrehajtása után.

(1) Ekkor $f_{|V|}(u_i) \leq f_{|V|}(u_{i+1})$ teljesül $\forall 1 \leq i \leq n$.

Biz: Az i-dik fázisban $f_i(u_i) \leq f_i(u_{i+1})$ teljesült az u_i választása miatt. Ezek után $f_i(u_i)$ már nem változott: $f_{|V|}(u_i) = f_i(u_i)$. Ugyan $f_i(u_{i+1})$ még csökkenhetett, de csak az $u_i u_{i+1}$ él mentén történt javítás miatt, hiszen az $(i+1)$-dik fázisban u_{i+1} bekerült az U_i halmazba, és a hozzá tartozó (r,l)-fb már nem csökken tovább. Ekkor $f_{i+1}(u_{i+1}) = min\{f_i(u_{i+1}), f_i(u_i) + l(u_i u_{i+1})\} \geq f_i(u_i)$, mivel $l(u_i u_{i+1}) > 0$. Ezért $f_{|V|}(u_i) = f_i(u_i) \leq f_{i+1}(u_{i+1}) = f_{|V|}(u_{i+1})$ $\square$

(2) $f_{|V|}(u_1) \leq f_{|V|}(u_2) \leq \cdots \leq f_{|V|}(u_n)$

(3) A Dijsktra-algoritmus outputjaként kaptt $f_{|V|}$-n élmenti javítás nem tud változtatni.

Biz: Tegyük fel, hogy $u_i u_j \in E(G)$ a G egy tetszőleges éle. Ha $i > j$, akkor (2) miatt $f_{|V|}(u_i) \geq f_{|V|}(u_j)$, ezért az $u_i u_j$ mentén történő javítás nem tudja $f_{|V|}(u_j)$-t csökkenteni, hisz $l(u_i u_j)$ pozitív. Ha pedig $i < j$, akkor az i-dik fázisban megrörtént az $u_i u_j$ mentén történő javítás, és ezt követően $f(u_i)$ nem változott, azaz $f_{|V|}(u_i) = f_i(u_i)$. A másik (r,l)-felső becslés pedig csak tovább csökkenhetett a késpbbi émj-ok során $f_{|V|}(u_j) \leq f_i(u_j)$. Ezért az $u_i u_j$ él mentén sem az i-dik fázisban, sem később nincs érdemi javítás. $\square$

Tétel: A Dijsktra-algoritmus helyesen működik, azaz G minden csúcsára igaz, hogy $dist(r,v) = f_{|V|}(v)$.

Biz: A Dijsktra-algoritmus az f_0 triviális (r,l)-felső becslésből indul ki, és élmenti javításokat alkalmaz. Így minden f_i (speciálisan $f_{|V|}$ is) (r,l)-felső becslés lesz. A fenti (3)-as megfigyelés miatt $f_{|V|}$-n nem végezhető érdemi élmenti javítás. Ezért egy korábbi (2)-es megfigyelés miatt $f_{|V|}$ pontos (r, l)-felső becslés, azaz $f_{|V|}(v) = dist_l(r, v) \forall v \in V(G)$. $\square$

„Lépésszámanalízis": Ha a G gráfnak n csúcsa és m éle van, akkor a Dijkstra-algoritmus n-szer keresi meg legfeljebb n szám minimumát, ami összességében legfeljebb $konst \cdot n^2$ lépést igényel. Ezen kívül legfeljebb m élmenti javítást véges, ami $konst' \cdot m$ lépés. Összességében tehát legfeljebb $konst'' \cdot (n^2 + m)$ lépésre van szükség, az algoritmus hatékony.