

Algoritmuselmélet

DAG, topologikus rendezés

Katona Gyula Y.

Számítástudományi és Információelméleti Tanszék
Budapesti Műszaki és Gazdaságtudományi Egyetem

Írányított körmentes gráfok

Definíció

Egy G irányított gráf **DAG**, ha nem tartalmaz irányított kört.

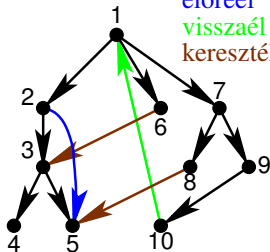
Directed Acyclic Graph

Alkalmazásai például:

- Teendők ütemezése \implies PERT
- Várakozási gráfok \implies adatbázisok

Fontos, hogy egy irányított gráfról el tudjuk dönteni, tartalmaz-e irányított kört.

faél
előreél
visszaél
keresztél



Ha a gráf egy mélységi bejárása során találunk visszaélet akkor a gráf nyilván tartalmaz irányított kört, azaz nem DAG.

Tétel

Legyen $G = (V, E)$ egy irányított gráf. Ha G egy DAG, akkor egyetlen mélységi bejárása során sincs visszaél. **Fordítva erősebb igaz:** ha G -nek van olyan mélységi bejárása, amelyre nézve nincs visszaél, akkor G egy DAG.

Bizonyítás.

⇒ ✓

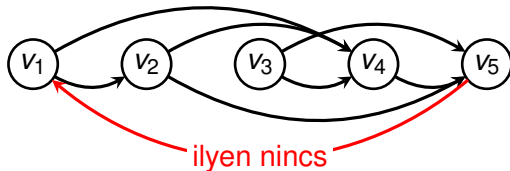
⇐ Indirekt: Tegyük fel, hogy nincs visszaél, de G nem DAG \implies van benne irányított kör: vegyük ennek a legkisebb mélységi számú v csúcsát, a kör előző pontja legyen u . $\implies \text{mszám}[v] < \text{mszám}[u]$
 $\implies uv$ vissza- vagy keresztél, de u elérhető v -ből G -ban irányított úton, ahol a mélységi számok mindegyike $\geq \text{mszám}[v]$ (a kör élein keresztül; **(részfa lemma)**) \implies nem lehet keresztél
 $\implies u$ a v leszármazottja a DFS fában $\implies uv$ visszaél. ⚡

□

DAG topologikus rendezése

Definíció

Legyen $G = (V, E)$ ($|V| = n$) egy irányított gráf. G egy **topologikus rendezése** a csúcsoknak egy olyan v_1, \dots, v_n sorrendje, melyben $x \rightarrow y \in E$ esetén x előbb van, mint y (azaz ha $x = v_i, y = v_j$, akkor $i < j$).



DAG topologikus rendezése

Tétel

Egy irányított gráfnak akkor és csak akkor van topologikus rendezése, ha DAG.

Bizonyítás.

⇒: Ha G nem DAG, akkor nem lehet topologikus rendezése, mert egy irányított kör csúcsainak nincs megfelelő sorrendje, mert semelyik csúcs sem előzheti meg a körben előtte állót.

⇐: Ez az irány a következő tételből következik. □

Topologikus rendezés mélységi kereséssel

Tétel

Végezzük el a G DAG egy mélységi bejárását, és írjuk ki G csúcsait a befejezési számaik szerint növekvő w_1, \dots, w_n sorrendben. A w_n, w_{n-1}, \dots, w_1 sorrend a G DAG egy topologikus rendezése.

Bizonyítás.

Azt kell belátnunk, hogy ha $w_i \rightarrow w_j$ éle G -nek, akkor $i > j$.

Tegyük fel, hogy van olyan $w_i \rightarrow w_j$, amire
 $j = \text{bszám}[w_j] > \text{bszám}[w_i] = i$.

Az élek osztályozásánál már láttuk, hogy ez csak úgy lehetne, ha **visszaél**. De mivel a gráf DAG ez sem lehet. ⚡



Lépésszám: $O(n + m)$

Legrövidebb utak élsúlyozott gráfban

Élsúlyozott gráf: $G = (V, E, c)$, ahol $c : E \rightarrow \mathbb{R}$ egy az élhalmazon értelmezett függvény. $c(e)$ az e él súlya (vagy **költsége** vagy **hossza**), lehet negatív is.

Élsorozat súlya (költsége, hossza): Az élsorozat éleinek súlyának összege.

Legrövidebb élsorozatok egy adott pontból:

Input: $G = (V, E, c)$, $s \in V$

(irányított vagy irányítatlan gráf is lehet)

Output: A legrövidebb (**legkisebb súlyú**) élsorozat és annak hossza s -ből a gráf minden más pontjába.

Megjegyzés: Ha az irányított gráfban van negatív súlyú irányított kör, akkor ezen sokszor végigmelve végtelenségig csökken az összsúly \implies nem feltétlen lesz legrövidebb élsorozat. Hasonló a helyzet irányítatlan gráfban lévő negatív összsúlyú kör esetén.

Legrövidebb utak élsúlyozott gráfban

Feltesszük, hogy a gráfban nincs negatív súlyú kör.

Így viszont ha egy élsorozat egy pontot többször is érint, akkor tartalmaz egy kört, aminek súlya ≥ 0 . Ha ezt kihagyjuk az élsorozatból, akkor az összsúly nem nő \implies kaptunk egy rövidebb vagy ugyanakkora súlyú élsorozatot.

Legtöbbször nem érdekes, hogy esetleg beilleszthető az útba egy nulla összsúlyú kör. \implies

Feltesszük, hogy a gráfban minden kör hossza pozitív.

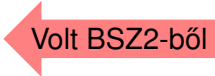
Ekkor minden legrövidebb élsorozat egy út.

Legrövidebb utak DAG-ban

Speciális eset: $c(e) = 1$ minden élre
 \implies Szélességi keresés – BFS

Mostani speciális eset: a gráf DAG

Az általános feladatról majd később lesz szó BSZ2-n.



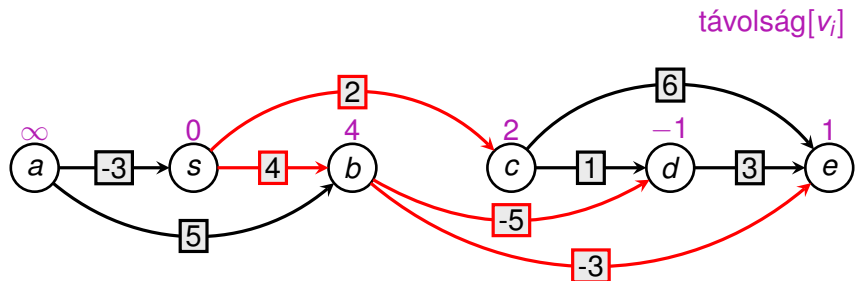
Volt BSZ2-ből

Legrövidebb utak DAG-ban

Az algoritmus éllistával adott gráf esetén

```
1: procedure SHORTEST-PATH( $G = (V, E, c), s$ )
2:   megkeresek  $G$ -nek egy topologikus rendezését:
    $v_1, \dots, v_n; S = v_k$   $\triangleright$  ehhez először DFS( $G, v$ )-t futtatjuk
3:   előállítom  $G$  fordított éllistáját  $\triangleright$  volt gyakorlaton
4:   for  $i = 1$  to  $k - 1$  do
5:     távolság[ $v_i$ ] =  $\infty$ 
6:   end for
7:   távolság[ $v_k$ ] = 0
8:   for  $i = k + 1$  to  $n$  do
9:     távolság[ $v_i$ ] :=  $\min_{(v_j, v_i) \in E} \{ \text{távolság}[v_j] + c(v_j, v_i) \}$ 
10:    honnan[ $v_i$ ] := az a  $v_j$ , ahol az előbb a minimumot találtuk
11:  end for
12: end procedure
```

Legrövidebb utak DAG-ban



Animáció

Legrövidebb utak DAG-ban

Tétel

Ha G egy éllistával adott súlyozott élű DAG, akkor az egy forrásból induló legrövidebb utak meghatározásának feladata $O(n + m)$ lépésben megoldható.

Bizonyítás.

Helyesség: Mivel minden él és út balról jobbra megy, ezért az s -től balra lévő csúcsokba nyilván nem lehet eljutni.

A jobbra levő pontokra indukcióval bizonyítunk: Tegyük fel, hogy minden $j < i$ -re távolság[v_j] már a legrövidebb út hossza. Indirekt tegyük fel, hogy v_i -be van rövidebb út, mint

$$\min_{(v_j, v_i) \in E} \{ \text{távolság}[v_j] + c(v_j, v_i) \}.$$

Ha ennek utolsó éle (v_x, v_i) , akkor v_x -be volna egy út, ami rövidebb, mint távolság[v_x]. ⚡



Legrövidebb utak DAG-ban

Tétel

Ha G egy éllistával adott súlyozott élű DAG, akkor az egy forrásból induló legrövidebb utak meghatározásának feladata $O(n + m)$ lépésben megoldható.

Bizonyítás.

Lépésszám:

- Topologikus sorrend keresése (2: sor a kódban): $O(n + m)$
- Fordított éllista előállítása (3: sor): $O(n + m)$
- Kezdőértékek beállítása (4-7: sor): $O(n)$
- (8-11: sor): Az összegek kiszámolása és egy minimum keresés $d_{be}(v_i)$ érték között (de mindenképpen legalább 1 lépés) a 9: sorban. Összegezve: $\sum_{k+1}^n (1 + O(d_{be}(v_i))) = O(n + m)$

Összesen: $O(n + m)$ lépés. □

Leghosszabb utak DAG-ban

Leghosszabb élsorozat nem feltételen létezik, ha van pozitív összsúlyú irányított kör. Mivel DAG-ban nincs semmilyen irányított kör, így mindig van leghosszabb élsorozat, ami egyben a leghosszabb út is.

Tétel

Ha G egy éllistával adott súlyozott élű DAG, akkor az egy forrásból induló leghosszabb utak meghatározásának feladata $O(n + m)$ lépésben megoldható.

Bizonyítás.

Ugyanaz az algoritmus működik, csak

$$\text{leghosszabb}[v_i] := \max_{(v_j, v_i) \in E} \{\text{leghosszabb}[v_j] + c(v_j, v_i)\},$$

illetve az s -ből el nem érhető csúcsok távolsága legyen $-\infty$. □

Nem DAG-ban is értelmes kérdés, hogy mi a leghosszabb út (ami nem megy át egy ponton többször).

Ez jóval nehezebb feladat, nem ismert rá gyors algoritmus.