Topologikus sorrend

Csima Judit
BME SZIT
csima@cs.bme.hu

2019. november 20.

Topologikus sorrend

Azt már láttuk, hogy mindkét tanult bejárás, a szélességi és a mélységi is alkalmas arra, hogy

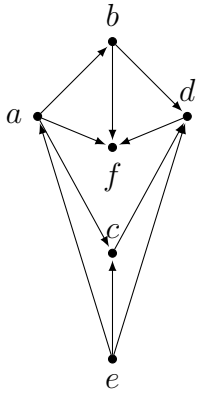
- irányítatlan, összefüggő gráfban feszítőfát találjunk
- eldöntsük, hogy egy irányítatlan gráf összefüggő-e
- megkeressük az összes olyan csúcsot, ami egy kezdő s csúcsból elérhető (akár irányított, akár irányítatlan gráfban)

Azt is láttuk, hogy a szélességi bejárás ezen felül még arra is jó, hogy a legkevesebb élből álló utakat megkeressük egy kezdő s csúcsból, a mélységi bejárás viszont erre nem használható. Jogosan kérdezhetjük, hogy akkor miért használunk mélységi bejárást egyáltalán, ha (látszólag) kevesebbet tud, mint a szélességi. Azt fogjuk a következőkben látni, hogy ez nincsen így, van egy olyan jellemzője a mélységi bejárásnak, ami nagy hasznunkra lesz majd, a befejezési számokat kiszámoló változat segítségével tudunk úgynevezett toplogikus sorrendet találni irányított gráfokban, amennyiben létezik ilyen a gráfban.

Most először megismerkedünk a topologikus sorrend definíciójával, majd azt nézzük meg, hogy mely gráfokban van ilyen és hogyan lehet megtalálni, ha létezik. Ebben a részben minden gráf irányított gráf, a definíciónak csak ekkor van értelme.

Definíció Egy irányított G gráf topologikus sorrendje a csúcsok olyan sorrendje, amire igaz az, hogy bármely xy irányított él esetén (azaz ha x-ből van él y-ba) teljesül, hogy x a sorrendben y előtt szerepel.

Az alábbi gráfban például toplogikus sorrend az e, a, c, b, d, f sorrend, mert ezen sorrend szerint a gráf minden éle "előre megy", azaz balról jobbra halad: a-ból b-be, c-be és f-be vezet él, mindhárom csúcs a után van; b-ből d-be és f-be van él, mindkét csúcs b után van, stb.



Ugyanebben a gráfban topologikus sorrend például e, a, b, c, d, f is, de nem topologikus sorrend c, b, e, d, a, f, mert ezen sorrend szerint az ec él nem előre megy, azaz c korábban van, mint az e csúcs a sorrendben.

Miért érdekes a topologikus sorrend?

Vannak olyan gyakorlati alkalmazások, amikben a feladat igazából egy topologikus sorrend keresését jelenti egy irányított gráfban. Ilyen például a következő helyzet:

Ha a gráf csúcsai elvégzendő munkák, a köztük levő xy irányított élek pedig azt jelentik, hogy az y munkát csak x elvégzése után lehet elvégezni, akkor a topologikus sorrend igazából a munkák egy olyan ütemezését jelenti, amiben minden munkára akkor kerül sor, amikor minden előfeltétele el lett végezve.

Vannak olyan alkalmazások is, amikor nem maga a topologikus sorrend érdekel minket, hanem a topologikus sorrend azért kell, mert a segítségével valami feladatot gyorsabban tudunk megoldani, mint általában. Erre lesz majd példa a jövő órán előkerülő legrövidebb út keresési feladat.

Milyen gráfokban van topologikus sorrend?

Ha a gráf maga egy irányított kör, akkor biztosan nincsen a gráfban topologikus sorrend, hiszen a kör (a gráf) egyik csúcsa sem lehet első ebben a sorrendben, hiszen a topologikus sorrend első csúcsába nem vezethet él, de ebben a gráfban minden csúcsba vezet él.

Hasonlóan látható, a következő tétel bizonyítása is:

1. tétel Ha van az irányított G gráfban irányított kör, akkor G-nek nincsen topologikus sorrendje.

Bizonyítás Ha lenne toplogikus sorrend, akkor ebben a kör csúcsai közül valamelyiknek az összes többi előtt kell szerepelnie, de akkor az ebbe a csúcsba a körben bevezető él visszafele menne a sorrend szerint.

Igaz a fenti tétel megfordítása is, azaz az, hogy ha G-ben nincsen irányított kör, akkor van G-ben topologikus sorrend, sőt az is ismert, hogy hogyan lehet egy körmentes, irányított gráfban megtalálni egy topologikus sorrendet:

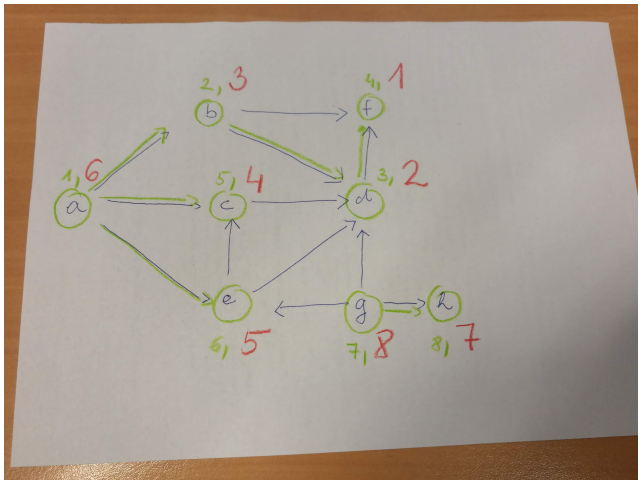
2. tétel Ha az irányított G gráfban nincsen irányított kör, akkor az alábbi módszerrel topologikus sorrendet kapunk:

1. Mélységi bejárást futtatunk (akárhonnan lehet kezdeni) és meghatározzuk a befejezési számokat. Ha az első indítás nem jár be minden csúcsot, akkor ahhoz, hogy minden csúcsot bejárjunk és minden csúcsnak legyen befejezési száma lehet, hogy az első kezdőcsúcsból indított mélységi bejárás befejezése után újra kell kezdenünk (esetleg többször is) az eljárást. Addig kezdjük újra és újra, amíg minden csúcsnak lesz befejezési száma.

2. A csúcsokat az előbb kapott befejezési számozás csökkenő sorrendjébe állítjuk, ez topologikus sorrend lesz.

Ezt a tételt nem bizonyítjuk (van ugyan bizonyítás erre, de nem tanuljuk), csak egy példán szemléltetjük a használatát.

Az alábbi gráfban lefuttatva a mélységi bejárást az a csúcsból (majd újrakezdve a g-ből) az alábbi történik (a zöld élek a felfedező élek, a zöld számok mutatják, hogy milyen sorrendben érjük el a csúcsokat, a piros számok pedig a befejezési számok):



Ez alapján a befejezési számok csökkenő sorrendjébe írva a csúcsokat a g, h, a, e, c, b, d, f sorrendet kapjuk. Ez valóban topologikus sorrend, mert minden él előre megy eszerint (ez az élek végignézésével látható).

Vegyük észre, hogy ez az eljárás, amivel topologikus sorrendet találunk, $O(n^2)$-es, hiszen a mélységi bejárás lépésszáma $O(n^2)$, utána pedig a topologikus sorrendet megadó *top* tömb (ahol a tömbben a topologikus sorrend szerint következnek az elemek) az alábbi kóddal $O(n)$ lépésben megkapható a bejárás végén kapott *bsz* tömbből:

```
top[v] = * minden v-re

ciklus v = 1-tól n-ig:
    top[n-bsz[v]] := v
ciklus vége
```

Ez a kód azért jó, mert ha a v csúcs befejezési száma $bsz[v]$, akkor ez a v csúcs a topologikus sorrendben az $(n-bsz[v])$-edik cellába kell, hogy kerüljön: ha $bsz[v] = 1$, akkor ezt a v-t fejeztük be először, azaz v lesz a tömb utolsó, azaz $n-1$-edik eleme, ha $bsz[v] = 2$, akkor ezt a v-t fejeztük be másodszor, azaz v lesz a tömb utolsó előtti, azaz $n-2$-edik eleme, ..., ha $bsz[v] = n$, akkor ezt a v-t fejeztük be utoljára, azaz v lesz a tömb első, azaz a $n - n = 0$ indexű cellába kerülő eleme.

A fenti kód lépésszáma $O(n)$, mert a *top* tömb beállítása n lépés az elején, utána pedig n-szer fut le a konstans lépésből álló ciklusmag.

A fentiek alapján tehát meg tudunk találni egy topologikus sorrendet a gráfban, ha a gráfban nincsen irányított kör.

Irányított körök felismerése

Az olyan irányított gráfot, amiben nincs irányított kör **DAG**-nak nevezzük (az angol elnevezés rövidítéséből: Directed Acyclic Graph). Azt láttuk tehát az előbb, hogy pontosan akkor van egy irányított gráfban toplogikus sorrend, ha a gráf DAG és tanultunk is egy olyan eljárást, ami DAG-ban talál egy topologikus sorrendet.

Ebben a témakörben egy fontos feladat az, hogy egy irányított gráfról eldöntsük, hogy DAG-e vagyis, hogy van-e benne irányított kör. Egy ilyen helyzet, amikor erre szükségünk lehet a korábban említett tranzakciók várakoznak egymásra, amíg meg tudják kapni a zárakat helyzet. Ebben a szituációban a tranzakciók a csúcsok, akkor van él egy csúcsból egy másikba, ha az első tranzakció vár a másodikra és egy irányított kör azt jelenti, hogy a tranzakciók körbevárnak egymásra, senki sem tud dolgozni, patt-helyzet alakult ki. Ezt fel kell tudnunk ismerni, hogy például az egyik tranzakció leállításával megtörjük a holtpontot.

Ennek a feladatnak a megoldására, tehát annak eldöntésére, hogy van-e irányított kör a gráfban egy, az előző részben látott algoritmushoz hasonló eljárást használhatunk:

1. Mélységi bejárást futtatunk (akárhonnan lehet kezdeni) és meghatározzuk a befejezési számokat. Ha az első indítás nem jár be minden csúcsot, akkor ahhoz, hogy minden csúcsot bejárjunk és minden csúcsnak legyen befejezési száma lehet, hogy az első kezdőcsúcsból indított mélységi bejárás befejezése után újra kell kezdenünk (esetleg többször is) az eljárást. Addig kezdjük újra és újra, amíg minden csúcsnak lesz befejezési száma.

2. Ellenőrizzük, hogy a befejezési számok csökkenő sorrendje szerinti sorrend topologikus sorrend-e. Ha igen, akkor a gráf DAG, ha nem, akkor nem DAG.

Ez az eljárás a következők miatt helyes:

- Ha az eljárás végén topologikus sorrendet kapunk G-ben, akkor G biztosan DAG, hiszen csak DAG-ban van topologikus sorrend.

- Ha az eljárás végén talált sorrendről az derül ki, hogy nem topologikus sorrend, akkor G biztosan nem DAG, mert ha G DAG lenne, akkor a korábbi (nem bizonyított, de igaz) tétel miatt a mélységi bejárásból kapott befejezési számok szerinti csökkenő sorrendnek topologikus sorrendnek kellene lennie. Ha tehát ez mégsem topologikus sorrend, akkor az csak azért lehet, mert G nem DAG.

Az most már csak a kérdés, hogy a mélységi bejárás után, a bsz tömb ismeretében hogyan döntjük el, hogy a befejezési számok szerinti csökkenő sorrend topologikus sorrend-e. Azt kell észrevenni, hogy ahhoz, hogy befejezési számok szerinti csökkenő sorrend topologikus sorrend legyen annak kell teljesülnie, hogy ha egy i csúcsból megy él egy j csúcsba, akkor j az i csúcs után álljon a befejezési számok szerinti csökkenő sorrendben, vagyis hogy $bsz[i] > bsz[j]$ fennálljon. Ezt az alábbi kóddal tudjuk leellenőrizni:

```
ciklus i = 1-tól n-ig:
    ciklus j = 1-tól n-ig:
        ha A[i,j] == 1:      // van-e él i-ből j-be
            ha bsz[i] < bsz[j]:          // rosszul állnak a befejezési számok
                return False
    ciklus vége
ciklus vége

return True
```

A kód logikája az, hogy minden élet megnézek (ott van él, ahol 1 van a szomszédossági mátrixban) és ha valahol találok visszafele menő élet, akkor False, egyébként (ha sose lesz visszafele menő él) True értéket adok vissza.

A DAG-ságot eldöntő eljárás lépésszáma azért $O(n^2)$, mert a mélységi bejárás és a bsz tömb elkészítése $O(n^2)$, a fenti eljárás pedig, amivel eldöntöttük, hogy topologikus sorrend-e a befejezési számok csökkenő sorrendje szintén $O(n^2)$, mert a belső ciklus magja $O(1)$ és n-szer fut le, vagyis a belső ciklus, ami egyben a külső ciklus magja is, $O(n)$-es és a mivel a kód maga a külső ciklus, aminek $O(n)$-es magja n-szer fut le, így az egész kód lépésszáma $O(n^2)$.