Algoritmusok és gráfok - Vizsga
2020. január 8.
Megoldások

A válaszokat indokolni kell, de a feladatokban szereplő tanult algoritmusokat nem kell részletesen leírni, elég csak azokat a részeket kifejteni, amelyek az indokláshoz szükségesek.

1. Lássa be megfelelő c konstans és n_0 küszöbérték megadásával, hogy ha egy algoritmus lépésszáma
$$2020n^2\log n + \frac{(n-1)(n-2)(n-3)(n-4)}{n} - 2019$$
akkor az algoritmus futásideje $O(n^3)$.

   Megoldások
   Olyan c pozitív konstanst és egy olyan n_0 küszöbindexet kell találnunk, melyre igaz, hogy
$$2020n^2\log n + \frac{(n-1)(n-2)(n-3)(n-4)}{n} - 2019 \le c\cdot n^3, \text{ha } n \ge n_0$$

   Ilyen c és n_0 értéket például az alábbi egyenlőtlenség-sorozattal tudunk találni:
   $2020n^2\log n + \frac{(n-1)(n-2)(n-3)(n-4)}{n} - 2019 \le 2020n^2\log n + \frac{(n-1)(n-2)(n-3)(n-4)}{n}$
   mert a negatív tagot el lehet hagyni, tetszőleges $n \ge 1$ esetén, továbbá
   $2020n^2\log n + \frac{(n-1)(n-2)(n-3)(n-4)}{n} \le 2020n^3 + \frac{(n-1)(n-2)(n-3)(n-4)}{n}$
   mert $\log n \le n$ tetszőleges $n \ge 1$ esetén, továbbá
   $2020n^3 + \frac{(n-1)(n-2)(n-3)(n-4)}{n} \le 2020n^3 + (n-2)(n-3)(n-4)$
   mert $\frac{(n-1)}{n} \le 1$ tetszőleges $n \ge 1$ esetén és
   $2020n^3 + (n-2)(n-3)(n-4) \le 2020n^3 + n^3$
   mert $(n-2) \le n, (n-3) \le n, (n-4) \le n$.

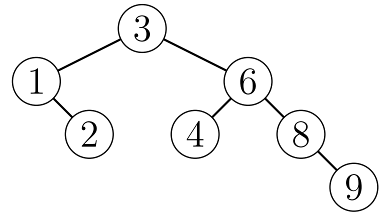
   Azt kaptuk tehát, hogy
   $2020n^2\log n + \frac{(n-1)(n-2)(n-3)(n-4)}{n} - 2019 \le 2021n^3$ tetszőleges $n \ge 1$ esetén, vagyis $c = 2021$ és $n_0 = 1$ jó választás.

2. (a) Írja le, hogy irányítatlan gráfban mit jelent az, hogy egy v csúcs $d(v)$-vel jelölt fokszáma 8.
   (b) Mondja ki az irányítatlan gráfokban a fokszámok összegéről tanult tételt és bizonyítsa be.

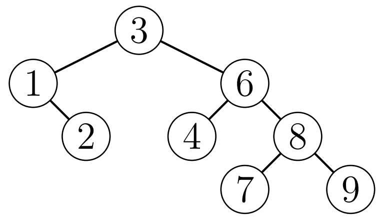
   Megoldás
   (a) Egy csúcs fokszáma irányítatlan gráfban a csúcsra illeszkedő élek száma, azaz $d(v) = 8$ azt jelenti, hogy a v csúcsra 8 él illeszkedik.
   (b) Irányítatlan gráfban a fokszámok összege $2e$, azaz
$$\sum_{v\in V} d(v) = 2e$$

mert amikor a fokszámokat összeadjuk, akkor minden élet kétszer számolunk, az él két végpontjánál.

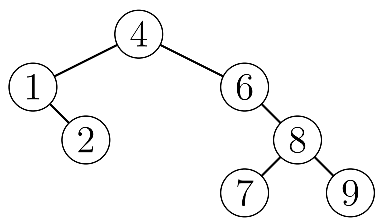
3. (a) Az órán tanult eljárásokkal szúrja be az alábbi bináris keresőfába a 7-et, majd törölje ki az így kapott fából a 3-at. Mindkét műveletnél magyarázza is el 1-2 mondatban, hogy mi miért történt.
(b) AVL-fa-e a kiindulási fa és AVL-fa-e a két művelet után kapott fa?



Megoldás
(a)



Mert a 7 nagyobb a 3-nál és 6-nál (jobbra megyünk), de kisebb a 8-nál (balra megyünk).



Mert a jobb részfa legkisebb elemét, azaz a 4-et kell a 3 helyére rakni.

(b) A kezdeti fa AVL-fa, mert minden csúcsra igaz, hogy a bal és jobb részfájának magassága maximum 1-gyel tér el, de a végén kapott fa nem AVL-fa, mert a 6 bal részfája üres, azaz 0 magasságú, a jobb fája meg kettő magas.

4. Egy hat csúcsú gráf csúcsai meg vannak címkézve a $2, 3, 4, 5, 6, 7$ számokkal és a gráfban akkor van él két csúcs között, ha a csúcsok címkéi relatív prímek (azaz a legnagyobb közös osztójuk 1). A gráf élei súlyozottak, az i és j címkéjű csúcsok közt futó él súlya $i \cdot j$ (azaz például a 6 és 7 címkéjű csúcsok között van él és ennek súlya 42).
Minimális súlyú feszítőfát keresünk ebben a gráfban Kruskal algoritmusával. Melyik éleket választja be az algoritmus és milyen sorrendben?

Megoldás
Kruskal algoritmusa élsúly szerint növekvő sorban nézi az éleket és pontosan akkor

választ be egy élet, ha az nem alkot kört a már beválasztott élekkel.
A gráf élei élsúly szerint növekvően: c(2,3) = 6, c(2,5) = 10, c(3,4) = 12, c(2,7) = 14, c(3,5) = 15, c(4,5) = 20, c(3,7) = 21, c(4,7) = 28, c(5,6) = 30, c(5,7) = 35, c(6,7) = 42.
Végigmegyünk ezen a listán és először bekerül a 6-os, a 10-es , 12-es és 14-es él, mert ezek egyike sem alkot kört a korábbiakkal. A 15-ös, 20-as, 21-es és 28-as él kört csinálna, ezeket nem vesszük be, de 30-t igen és ezzel kész is a fa.

5. Dijkstra algoritmusát használjuk az A, B, C, D, E csúcsokból álló irányított, élsúlyozott G gráfban az A kezdőcsúcsból, eközben a d tömb így változik (az egyes sorok a d tömb változását mutatják egy-egy csúcs KÉSZ halmazba kerülése után).

A	B	C	D	E
*	5	∞	1	∞
*	3	5	*	∞
*	*	4	*	∞
*	*	*	*	8
*	*	*	*	*

Milyen csúcsokkal van biztosan összekötve a B csúcs (bejövő és kimenő élekre is gondoljon) és mi ezeknek az élsúlya?

Megoldás
Az első sorból látszik, hogy A-ból van egy él a B-be 5 súllyal, mert az első sor a kezdőcsúcsból kimenő élek súlyait tartalmazza.
Az első és a második sor közti változásból látható, hogy ekkor D került be a KÉSZ-be 1 távolsággal és emiatt lett B értéke 3, azaz biztosan van él D-ből B-be, 2 súllyal.
A második és harmadik sor közti változásból látható, hogy ekkor B került be a KÉSZ-be 3 távolsággal és emiatt lett C értéke 4, azaz a 3 távolságú B-n át van egy rövidebb út C-be, 1 súllyal.

6. DFS-t (mélységi bejárást) futtatunk az A, B, C, D, E, F csúcsokból álló irányítatlan G gráfon a D kezdőcsúcsból, a DFS fába a DA, AF, FE, DB, BC élek kerülnek be ebben sorrendben. Mely csúcsokkal lehet összekötve az F csúcs a gráfban és melyekkel nem?

Megoldás
A-val biztosan össze van kötve F, hiszen van a DFS fában AF él.
B-vel nem lehet összekötve F, mert ha lenne BF él, akkor az FE él bejárása után nem lépnénk vissza F-ből A-ba, hanem az FB éllel felfedeznénk B-t.
A C csúcs ugyanezért nem lehet összekötve F-fel: ha lenne CF él, akkor az FE él bejárása után nem lépnénk vissza F-ből A-ba, hanem az FC éllel felfedeznénk C-t.
D és F között lehet él, akkor is futhat ugyanígy a mélységi bejárás (azaz ekkor is a DA, AF, FE, DB, BC élek választódnak be ebben a sorrendben), mert F éleinek vizsgálatakor D már be van járva és így mindegy, hogy vezet-e oda él vagy sem.
E-vel biztosan össze van kötve F, hiszen van a DFS fában FE él.

7. Szomszédossági mátrixával adott egy n csúcsú, irányítatlan G gráf és adott egy, a csúcsokkal indexelt R tömbben a csúcsok egy részhalmaza oly módon, hogy ha a v csúcs nincsen benne a részhalmazban, akkor $R[v] = 0$, ha pedig v benne van a részhalmazban, akkor $R[v] = 1$.
   Azt szeretnénk eldönteni, hogy igaz-e, hogy a gráf minden élének legalább egyik végpontja az R halmazban van.

   Adjon erre a feladatra $O(n^2)$ lépésszámú algoritmust.

   Megoldás
   Végigmegyünk a szomszédossági mátrix elemein (két egymásba ágyazott ciklussal, a külső ciklus a sorokon fut, a belső pedig a sorokon belül az oszlopok indexein) és ha az i-edik sor és j-edik oszlop találkozásánál 1-t látunk (azaz van él az i-edik és j-edik csúcs között), akkor megnézzük, hogy $R[i]$ és $R[j]$ közül legalább az egyik 1-e.
   Ha végig tudunk menni a mátrixon úgy, hogy ez mindig teljesül, akkor a gráf minden élének legalább egyik végpontja az R halmazban van, egyébként (ha találunk olyan 1-es bejegyzést a máztixban valami i és j csúcs között, ahol $R[i]$ és $R[j]$ is 0), akkor nem igaz, hogy a gráf minden élének legalább egyik végpontja az R halmazban van.

   Pszeudokóddal:

```
Igaz_e := True
ciklus i =1-től n-ig:
    ciklus j = 1-től n-ig:
        ha A[i,j] == 1:
            ha (R[i] == 0 és R[j] ==0):
                Igaz_e : = False
    ciklus vége
ciklus vége
return Igaz_e
```

   Ez azért jó, mert $A[i, j] == 1$ ellenőrzése pont azt jelenti, hogy éleket nézem meg és pont azt ellenőrzöm, ami a kérdés, hogy vagy i vagy j benne van-e R-ben.
   A lépésszám pedig $O(n^2)$, mert a külső ciklus n-szer fut le, ennek a magja egy n-szer lefutó belső ciklus, aminek a magja $O(1)$ lépés, vagyis a belső ciklus $O(n)$, az egész eljárás pedig $O(n^2)$.

8. Szomszédossági mátrixával adott egy n csúcsú, irányított, élsúlyozott G gráf csupa pozitív élsúllyal, ahol minden csúcsnak van egy címkéje, egy pozitív egész szám, ezek a címkék nem feltétlenül különbözőek. A csúcsok címkéi egy, a csúcsokkal indexelt T tömbben adottak. Ki van jelölve még két csúcs is, A és B, a gráfban. Adjon $O(n^2)$ lépésszámú algoritmust, ami megtalálja a legrövidebb összhosszúságú olyan

utat A-ból B-be ahol az úton csak 100-nál kisebb címkéjű csúcsokon haladunk át (A és B címkéjét is beleértve) vagy jelzi, ha nincsen ilyen út.

Megoldás

Algoritmus:

1. Módosítsuk a gráfot úgy, hogy a 100 vagy annál nagyobb címkéjű csúcsokba belemenő és onnan kijövő éleket kitöröljük.
2. Ebben az új gráfban futtassuk a Dijkstra algoritmust az A csúcsból. Ha B távolsága a végén végtelen, akkor nincs jó út, egyébként pedig B távolsága a keresett érték.

A gráf módosítása pontosabban: végigmegyek a T tömbön és ha az aktuálisan vizsgált $T[v]$ érték legalább 100, akkor a gráf szomszédossági mátrixában a v-edik sor és oszlop minden értékét végtelenre állítom.

```
ciklus v =1-től n-ig:
    ha T[v] >= 100:
        ciklus j = 1-től n-ig:
            A[v,j] : = végtelen
            A[j,v]: = végtelen
        ciklus vége
ciklus vége
```

Lépésszám:

1. A T tömb minden értékére egyszer megyek végig egy során és egy oszlopán a mátrixnak, az $2n$ módosítás, amit n-szer kell legfeljebb megtennem, ez $n \cdot O(n) = O(n^2)$. (Vagy a pszeudokódra hivatkozva: a külső ciklus n-szer fut le, ennek a magja 1 lépés és egy belső ciklus, ami O(n)-es, mert n-szer fut le és magja $O(1)$, vagyis $n(1 + O(n)) = n \cdot O(n) = O(n^2)$).
2. A kapott mátrix is egy n csúcsú gráfhoz tartozik, ebben Dijkstra $O(n^2)$ lépésben fut, így a módosítás és a futtatás együtt $O(n^2) + O(n^2) = O(n^2)$.

Jóság: A módosítással éppen azok a csúcsok válnak elérhetetlenné, amiket nem szabad használnunk, azaz az új gráf útjai megegyeznek az eredeti gráf használható útjaival. Dijkstra algoritmusa hasznáható, mert minden élsúly pozitív.