Algoritmuselmélet - Minta zárthelyi, megoldásokkal
2023 tavasza

A munkaidő 100 perc. A VÁLASZOKAT INDOKOLNI KELL.
Hivatkozni csak az előadáson tanultakra lehet.

1. Az alábbi függvények közül pontosan egyre igaz, hogy $O(n^2)$-es.

$$\log(n^2) + \tfrac{n(n-1)(n-2)}{2010} \qquad 2020n\log n + \frac{2^n \cdot n(n-1)}{3^n} \qquad 20n^2(\log n)^2 + 8$$

(a) Válassza ki ezt a függvényt és lássa be megfelelő c konstans és n_0 küszöbérték megadásával, hogy $O(n^2)$-es.
(b) A másik két függvény egyikéről (szabadon választhat, hogy melyikről) bizonyítsa be, hogy az nem $O(n^2)$-es.

Megoldás

(a) A második függvény $O(n^2)$-es, mert

$$2020n\log n + \frac{2^n \cdot n(n-1)}{3^n} \leq 2020n\log n + n(n-1) \leq 2020n^2 + n^2 = 2021n^2$$

Mindkét egyenlőtlenség $n \geq 1$ esetén igaz, az első becslésnél azt használtuk, hogy $\frac{2^n}{3^n} \leq 1$, a másodiknál pedig azt, hogy $n\log n \leq n^2$ és $n(n-1) \leq n^2$.
A fentiek miatt $c = 2021$ és $n_0 = 1$ jó választás $O(n^2)$ definíciójában.

(b) A harmadik függvényről indirekt bizonyítással látjuk be, hogy nem $O(n^2)$.
Ha $O(n^2)$-es volna, akkor létezne olyan $c > 0$ konstans és $n_0 \geq 1$ küszöb, hogy $20n^2(\log n)^2 + 8 \leq c \cdot n^2$ áll fenn, ha $n \geq n_0$.
Innen azonban

$$c \cdot n^2 \geq 20n^2(\log n)^2 + 8 \geq 20n^2(\log n)^2$$

adódik, amit n^2-tel leosztva kapjuk, hogy

$$c \geq 20 \cdot (\log n)^2$$

teljesül, ha $n \geq n_0$, ami ellentmondás, mert $\log n$ minden határon túl nő, nem maradhat semmilyen c konstans alatt örökké.

Pontozás vázlata

(a) A jó függvény kiválasztása:	**1 pont**
Jó becslések:	**2 pont**
Jó c és n_0 megadása:	**2 pont**
(b) Jó indirekt elindulás:	**1 pont**
Helyes becslések, amik az ellentmondáshoz vezetnek:	**2 pont**
Az ellentmondás elmagyarázása:	**2 pont**

A (b) részben az első függvény választása esetén is hasonlóan oszlanak el a pontok.

Az O puszta definíciójának felírásáért nem jár pont.

2. Az $A[1:n]$ tömb pozitív számokat tartalmaz (a számok nem feltételenül egészek és lehetnek 1-nél kisebb értékek is). Szeretnénk megtalálni azt az $A[i:j]$ résztömböt ($1 \leq i \leq j \leq n$), melyben a számok szorzata a lehető legnagyobb. Adjon $O(n)$ lépésszámú, dinamikus programozást használó algoritmust az elérhető legnagyobb érték megtalálására az alábbi részfeladatok megoldásával: $M[j]$ adja meg a legnagyobb elérhető szorzatot, ha a résztömb utolsó cellája j.

Megoldás és pontozás

A részfeladatokat $i = 1, 2, \ldots, n$ sorrendben oldjuk meg. **1 pont**
$M[1] = A[1]$, mert ha egy résztömb az első cellában végződik, akkor az csak az első elemből áll. **1 pont**
Az $i \geq 2$ esetben $M[i] = \max(A[i], M[i-1] \cdot A[i])$, **2 pont**
mert az i-edik cellában kétféleképpen végződhet egy résztömb:

- a résztömb csak az i-edik cellából áll, ekkor a szorzat $A[i]$
- az i-edik cella előtt még más cellákat is használunk, a legnagyobb ilyen szorzat értéke a korábbi celláig kapható legnagyobb szorzat és az aktuális elem, azaz $A[i]$ szorzata **3 pont**

Az elérhető legnagyobb érték az $M[i]$ értékek maximuma, mert így figyelembe vesszük az összes lehetséges pozíciót a résztömb utolsó cellájára. **1 pont**
Az algoritmus lépésszáma azért $O(n)$, mert az n részfeladat mindegyike konstans lépésben megoldható és a maximumkeresés is $O(n)$ az n méretű tömbben. **1 + 1 pont**

3. Egy **irányítatlan** G gráf csúcsai A, B, C, D, E, F, H. Mélységi bejárást (DFS-t) futtatva a D csúcsból kiindulva a feszítőfába a DA, AB, AC, DE, EF, FH élek kerülnek be ebben a sorrendben.
(a) Lehetséges-e, hogy a G gráfban van AE él?
(b) Lehetséges-e, hogy a G gráfban van EH él?

Megoldás és pontozás

(a) A DFS futása során visszalépünk az A csúcsból a D-be és majd csak ezután járjuk be az E csúcsot és ha lett volna AE él, akkor legkésőbb az A szomszédainak vizsgálatakor E-be mentünk volna még az előtt, hogy A-ból visszalépünk, **4 pont**
tehát biztos nincsen AE és a gráfban. **1 pont**
(b) Az EH élet a H csúcs szomszédainak vizsgálata során vizsgáljuk először, de ekkor az E csúcs már be van járva, ezért az él megléte nem változtatja meg a DFS futását , **4 pont**
ezért lehetséges, hogy van EH él a gráfban **1 pont**

Indoklás nélküli helyes tippre nem jár pont.

4. Éllistájával adott egy n csúcsú irányítatlan gráf és abban két kijelölt csúcs, A és B. A gráf minden csúcsa ki van színezve vagy pirosra vagy kékre, ez az információ egy S tömbben adott, amely a csúcsokkal van indexelve és ahol $S[v]$ a v csúcs színét adja meg. A gráf egy élét tarkának nevezzük, ha egyik végpontja piros, a másik pedig kék. Adjon $O(n+m)$ lépésszámú algoritmust, ami meghatározza, hogy van-e olyan út az A csúcsból a B csúcsba, ami csupa tarka élből áll és ha van ilyen, akkor azt is megmondja, hogy hány tarka élből áll a legrövidebb ilyen út.

Megoldás és pontozás

Az az ötlet, hogy a tarka élek gráfjában kell legrövidebb utat keresni: **1 pont**
ALGORITMUS (a fenti ötlet egy lehetséges pontos megvalósítása)

(a) Kitörlünk minden nem tarka élet a gráfból **1 pont**
Ezt úgy lehet megtenni, hogy végigmegyünk az éllistán és ha egy w csúcs benne van egy v csúcs szomszédai között, de v és w színe megegyezik, akkor w-t töröljük v szomszédai közül: **1 pont**

(b) Az új G_1 gráfban futtatunk BFS-t A-ból (újrakezdés nélkül) **2 pont**
és ha B nem járódik be eközben, akkor nincsen megfelelő út, ha pedig B bejáródik, akkor a B-hez tartozó távolság adja meg a keresett élszámot. **1 pont**

HELYESSÉG
A módosítás után pontosan a csupa tarka élekből álló uta maradnak meg az új gráfban **1 pont**
és a BFS meghatározza ezek közül a legrövidebbet. **1 pont**

LÉPÉSSZÁM
A módosítás $O(n+m)$, mert egyszer kell végigmenni az $O(n+m)$ méretű éllistán és minden bejegyzésnél konstans sok munkát végzünk. **1 pont**
Az új gráfban n csúcs és $m' \leq m$ él van, a BFS lépésszáma pedig $O(n+m')$, azaz $O(n+m)$. **1 pont**

5. A következő nyáron sok minket érdeklő fesztivál lesz, azonban sajnos ezek időpontjai között vannak átfedések. Ha egy fesztiválra elmegyünk, azon az első naptól az utolsóig ott akarunk lenni, de másnap már mehetünk egy újabbra. A szóba jövő f fesztivál mindegyikéről tudjuk,hogy melyik nap kezdődik és melyik nap végződik, célunk hogy minél több napot töltsünk fesztiválokon. Fogalmazza meg a feladatot egy gráfelméleti problémaként és adjon $O(f^2)$ lépésszámú algoritmust a fesztiválok egy ilyen kiválasztására.

Megoldás

GRÁFELMÉLETI ÁTFOGALMAZÁS ÖTLETE

A gráf csúcsai a fesztiválok és akkor van irányított él a v_i fesztiválból a v_j fesztiválba, ha v_i utolsó napja megelőzi v_j első napját (azaz el tudunk menni v_i után v_j-re). **2 pont**
Az a gondolat, hogy minden élhez az egyik fesztivál hosszát rendeljük és leghosszabb utat akarunk keresni: **1 pont**
ALGORITMUS (a fenti ötlet megvalósítása pontosan) és HELYESSÉG

A fenti gráf egy DAG, mert egy gráfbeli, legalább egy élből álló úthoz tartozó fesztiválsorozat kezdőnapjai szigorúan növő sorozatot alkotnak. **1 pont**
A pontos megvalósításhoz vegyünk fel még egy csúcsot, amiből minden fesztiválcsúcsba vezet egy irányított él. Mivel ebbe az új csúcsba nem vezet él, ezért a gráf DAG marad. **1 pont**
Mindegyik élhez rendeljük hozzá annak a fesztiválnak a hosszát, amibe az él mutat. **1 pont**
Így a gráfban az olyan utak, amik az extra csúcsból indulnak megfelelnek a lehetséges legális fesztiválsorozatoknak és egy ilyen út hossza a megfelelő fesztiválokon töltött összes idővel egyezik meg. **1 pont**
Mivel a gráf DAG, használjuk a DAG-ban leghosszabb utat kereső algoritmust az extra csúcsból futtatva (az utak nyomonkövetésével), így minden fesztiválcsúcsba megkapjuk az ezzel a fesztivállal végződő leghosszabb utat. A fesztiválokhoz így kapott értékek maximumát megkeresve, az adott fesztiválhoz tartozó leghosszabb út adja a legjobb kiválasztást. **1 pont**

LÉPÉSSZÁM

A gráf szomszédossági mátrixának elkészítése $O(f^2)$ lépés, mert minden fesztivál-párt egyszer kell összehasonlítanunk (ez $O(f^2)$ lépés, ezzel megvan az $(n+1)$-szer $(n+1)$-es mátrix n-szer n-es része), az extra csúcshoz tartozó sort pedig $O(f)$ lépésben fel tudjuk tölteni. **1 pont**
A kapott gráfnak $n = f+1$ csúcsa van, a tanult algoritmus így $O(n^2) = O((f+1)^2) = O(f^2)$ lépésben fut. **1 pont**

6. Dijkstra algoritmusát futtatjuk egy, az A, B, C, D, E, F csúcsokból álló irányítatlan gráfon. Az alábbi táblázat az *eddigi_legjobb* tömb változását mutatja a futás közben.
Melyek azok az élek, amik biztosan szerepelnek a gráfban és mi ezeknek a súlya?

A	B	C	D	E	F
$*$	∞	∞	1	3	∞
$*$	∞	5	$*$	2	4
$*$	7	4	$*$	$*$	4
$*$	6	$*$	$*$	$*$	4
$*$	5	$*$	$*$	$*$	$*$

Megoldás és pontozás

A futás A-ból indult, mert az A csúcsnál szerepel $*$ az első sorban. **1 pont**
Az első sorból tudjuk, hogy A-ból D-be és E-be van él, ezek súlya rendre 1 és 3, mert az első sor a kezdőcsúcsból kivezető élek súlyát adja meg. **1 + 1 pont**
A későbbi sorok vizsgálata során akkor lehet egy xy él meglétére következtetni, ha azt látjuk, hogy x KÉSZ-be kerülése után a következő sorra áttérve y értéke csökken és ekkor azt is tudjuk, hogy az xy él súlya *eddigi_legjobb*$[y]$ - az az érték, amivel x a KÉSZ-be került. **4 pont**
(Ezt nem muszáj így általános alakban leírni, az is megfelelő, ha más módon derül ki, hogy hogyan következtetjük ki az éleket és a súlyukat (például egyesével elmagyarázzuk). Az a lényeg, hogy szükséges az indoklás és az indoklásból ki kell derülnie, hogy melyik értéket miből találtuk ki.)
Az a csúcs kerül a KÉSZ-be, akinél a $*$ megjelenik. **1 pont**
Ezek alapján a 2. sorra áttérve látjuk, hogy a biztosan meglevő élek és súlyaik: $c(DC) = 4, c(DE) = 1, c(DF) = 3$. **1 pont**
A 3. sorból látjuk, hogy $c(EC) = 2, c(EB) = 5$, a 4. sorból kiderül, hogy $c(CB) = 2$, az 5. sorból pedig $c(FB) = 1$. **1 pont**

7. Adott egy pozitív egész számokból álló összeg: $a_1 + a_2 + \ldots + a_n$. Az összeadás jelek közül szorzásra cserélhetjük bármelyikeket, de csak úgy, hogy ne legyenek szomszédos szorzások (azaz minden szám legfeljebb egy szorzásban szerepelhet). Adjon $O(n)$ futásidejű algoritmust, ami meghatározza, hogy mekkora az ilyen cserékkel kapható számok maximuma. Például az $1+4+3+2+3+4+2$ összegből kapható maximum $29 = 1 + 4 \cdot 3 + 2 + 3 \cdot 4 + 2$.

Megoldás és pontozás

Dinamikus programozást használunk, minden $1 \leq i \leq n$ esetén a következő részfeladatot oldjuk meg:
$M[i]$ = az $a_1 + a_2 + \ldots + a_i$ összeg módosításával elérhető legnagyobb olyan érték **2 pont**
Ezeket a feladatokat $i = 1, 2, \ldots, n$ sorrendben oldjuk meg. **1 pont**
$M[1] = a_1$, mert ez az egyetlen érték, ami előállhat, illetve
$M[2] = \max(a_1 \cdot a_2, a_1 + a_2)$, mert ezek az egyetlen értékek, amik előállhatnak. **1 pont**
Ha $i \geq 3$:
Két lehetőség van:

- az utolsó műveleti jel $+$: ekkor a legjobb, amit kaphatunk $M[i-1] + a_i$
- az utolsó műveleti jel $\cdot$: ekkor a legjobb, amit kaphatunk $M[i-2] + a_{i-1} \cdot a_i$, mert ekkor az utolsó előtti művelet biztosan $+$ **3 pont**

Ezek közül kell a nagyobbat venni, ez lesz $M[i]$ értéke. **1 pont**
A keresett érték M definíciója szerint $M[n]$. **1 pont**
A lépésszám azért $O(n)$, mert n feladatot oldunk meg és mindegyik konstans lépésben megvan. **1 pont**