ALGRAF ELŐADÁS

10.

2018. 11. 15. Cs.

előző óra:

Algoritmus:

telyes_DFS(G):

bejárva[v] = 0 ∀ v∈V

T = ∅

for v = 1 to n:

    if bejárva[v] == 0:

        DFS(G, v)

DFS(G, v):

bejárva[v] = 1

for w in (v szomszédai):

    if bejárva[w] == 0:

        T.append((v, w))

        DFS(G, w)

LSz:

teljes_DFS(G): $O(n)$ mert minden csúcsra konstans műveletek

DFS(G, v): „n" ilyen van, aminek egyenkénti lépésszáma:

mátrix-szal: n / éllistával: d(v)

$O(n^2)$ / $O(e)$

$O(n^2)$ vagy $O(n+e)$

ZH:

holnap 8-10

érkezés: 8:00

kezdés 8:15

90 perc

terembeosztás vezetéknév kezdőbetűi szerint

A-Ö: E1B

P-Zs: E1C

ha algo kell akkor lépésszám és helyesség is

nem lehet számológépet sem mobiltelefont használni

összetűzött lapokra

3-4 lap elég

első lapra gyakorlatvezető neve

eredmények emailben lesznek kiküldve

40% (24/60p) aláíráshoz

pótzh:

lehet javítani is

ugyanaz lesz a tananyag is

DFS-sel bezárólag

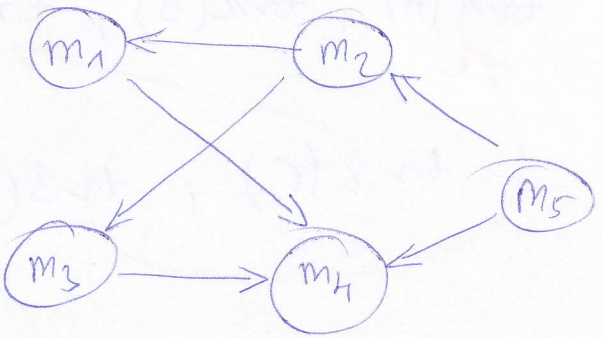
mélységi és szélességi bejárások összehasonlítása:

	BFS	DFS
legrövidebb utak meghatározása (ut = élsúly)	igen	nem
leghosszabb út (út = élsúly)	nem	nem
G összefüggő-e (irányítatlan gráf)	igen (ha nem kell újraindítani)	igen (ha nem kell újraindítani)
egy adott csúcsból elérhető-e egy másik	igen	igen
feszítőfa vagy feszítő erdő keresése	igen	igen
Lsz	$O(n^2)$ / $O(n+e)$	$O(n^2)$ / $O(n+e)$

Motivációs példák:

① Adottak munkák: $m_1, m_2, \ldots, m_n$ és az, hogy melyik munkát kell elvégezni mielőtt egy adott ~~[illegible]~~ munkát el tudnánk kezdeni.

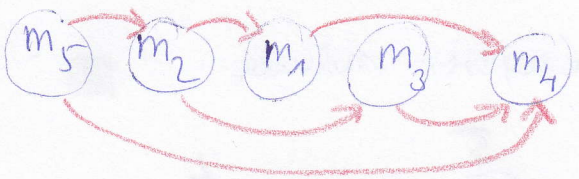
pl.:



Cél: helyes ütemezés:

m_i akkor sorra kerül, akkor minden előfeltétel már elkészült

pl.:



② Adatbázis-kezelés: tranzakciók

pl.: bankbeli átutalásnál:

„A" számláról utalás „B"-re

2 lépés: | levonás x összeget „A"-ról |
| jóváírás x összeget „B"-re |

legyenek: - 2 chat „A"-ra és „B"-re
- műveleti
- zár feloldása

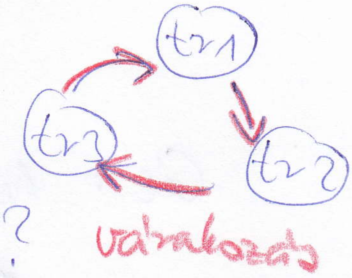
előfordulhat:

	zár
tr1 : A → B	A & B
tr2 : B → C	B & C
tr3 : C → A	C & A

zárkérés: külön-külön a rekordokra

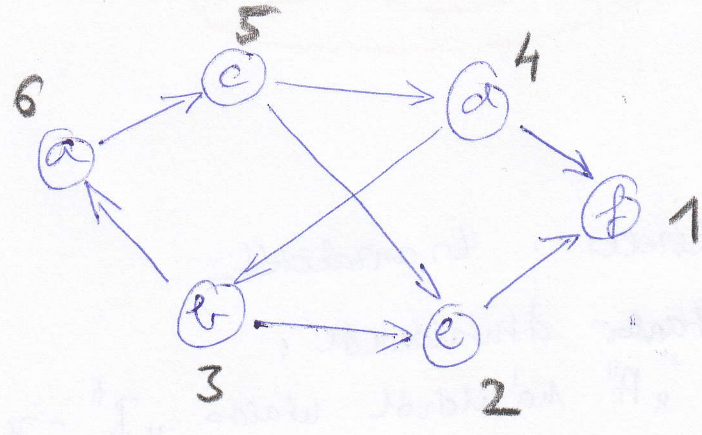
zárkérések: tr1(A), tr2(B), tr3(C), tr1(B)

tr2(C), tr3(A)



van-e irányított kör a gráfban?

DFS befejezési számok:

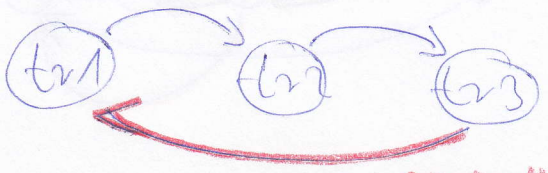


Def.: Irányított G gráfban topologikus sorrend:

olyan sorrendje a csúcsoknak, ahol ha $u \to v$ akkor u a v előtt van (minden él balról jobbra halad)

Észrevétel: ha G-ben van irányított kör, akkor a csúcsoknak nincs topologikus sorrendje

pl.:



emiatt nem jó!

Tétel: Ha van G-ben topologikus sorrend, akkor G-ben nincs irányított kör

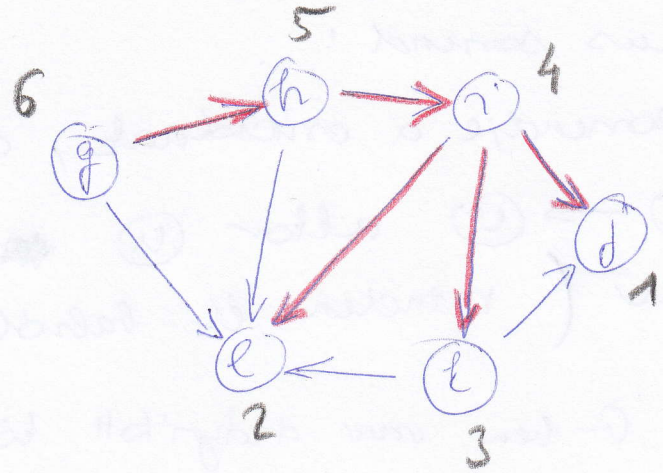
Tétel: Ha G-ben nincs irányított kör, akkor G-ben van topologikus sorrend

Bit. vázlat: G-ben nincs irányított kör és

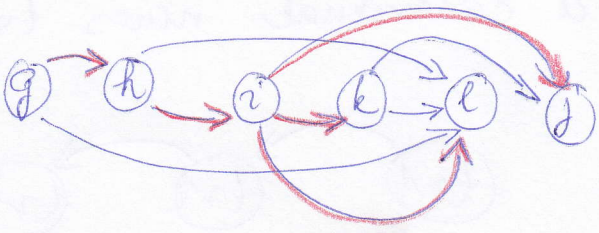
DFS-t futtatunk G-ben valamelyik s csúcsból és elkészítjük a befejezési számokat akkor a befejezési számok szerint csökkenő sorrendje a csúcsoknak topologikus sorrend

[Ezt nem bizonyítjuk]

pl.:



⇝

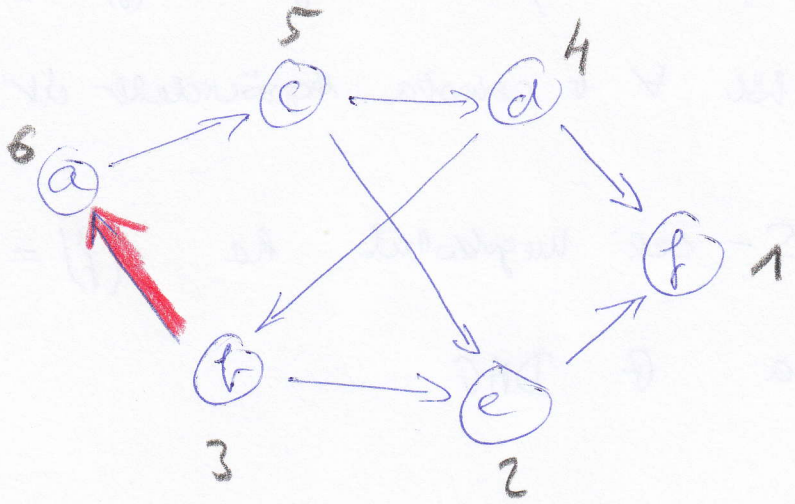


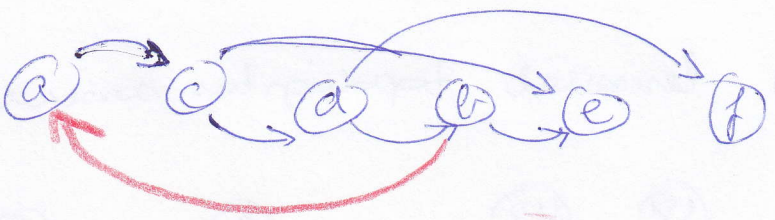
Tehát 1 db DFS-sel $O(n+e)$ / $O(n^2)$ műveletigénnyel megvan a topologikus sorrend, ha G-ben nincs irányított kör

Hogyan lehet eldönteni, hogy G-ben van-e irányított kör?

DFS-t futtatunk amíg sorra befejeződnek a csúcsok, a csúcsokat pedig mindig befejezési sorrend szerint csökkenő sorrendbe rendezem, és meg kell vizsgálni, hogy ez topologikus sorrend-e
Ha topologikus akkor nincs benne irányított kör.
Ha nem topologikus, akkor van benne irányított kör.

pl.:





nem topologikus sorrend

Def.: G (irányított) gráf DAG | Directed Acyclic Graph

ha nincs benne irányított kör.

Miért jó ha egy gráf DAG?
Mire jó a topologikus sorrend?

Példa: útvonaltervezés (Waze, Google Maps)
↳ legrövidebb út keresése
csúcsok: kereszteződések
él: utak
élsúly: távolság / idő ...
élsúlyok összege: út hossza / ideje

Input: G gráf, s csúcs, G éleinek súlyai: $c(f)$ élsúly minden f élre

Cél: s-ből $\forall v$ csúcsba legrövidebb út

Ezt a BFS-sel megoldottuk ha $c(f) = 1 \ \forall f$ élre

Most: ha G DAG

Algo: 1.) keressünk topologikus sorrendet G-ben DFS-sel

v_1 v_2 ... s ... v ... v_n

2.) $\text{távolság}(s) = 0$

$\text{távolság}(v_i) = \infty \ \forall v_i$: s előtt

("nincs visszafelé út" mert ez egy topologikus sorrend)

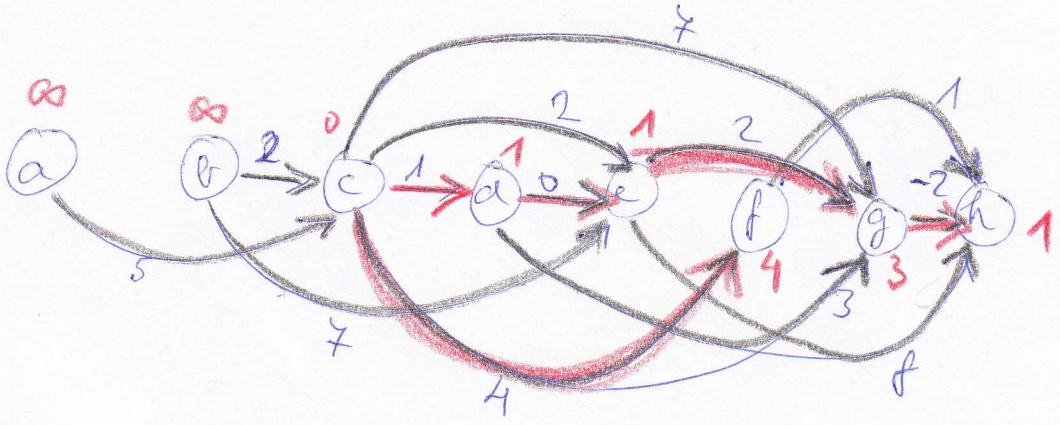
3.) topologikus sorrend szerint haladva kiszámolom a távolságokat

legrövidebb út: valamilyen út "s" és "u" között és utolsó lépésként "u" $\rightarrow$ "v" lépés ("mindig van egy utolsó lépés")

u: "s"-ből és "v" előtt van

ekkor: $\text{távolság}[v] = \min\{\text{távolság}[u] + c(u,v)\}$

$\forall u$: amire $u \rightarrow v$ él létezik



start = c

ALGRAF GYAKORLÁS 2018. 11. 15. G.

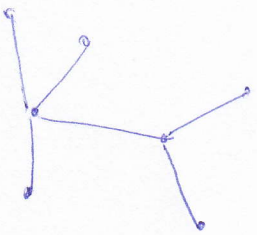
6/8

fokszámok:

a) 1, 1, 1, 1, 1, 3, 4

van-e ilyen fa?

pl.:



van ilyen!

b) 1, 1, 1, 1, 2, 3, 3, 4

fokszámok összege irányítatlan gráfban = 2·e

$16 = 2 \cdot e \Rightarrow e = 8$ éle van

fában mindig $n-1$ él van

$8-1 \neq 8$

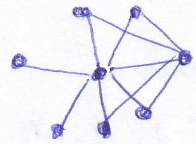
eset részletezésével is elfogadható a megoldás, amikor

6./ (10) Páratlan irányítatlan gráf

van-e olyan amiben a fokszámok:

a) 1, 1, 1, 2, 3, 4, 5, 7

konstrukció amiben



b) 1, 1, 1, 2, 3, 4, 5, 6 $\sum d(v)$ = páratlan

⇓ 2.e

nincs!

ZH kérdés P-ben
beosztás vezetéknév szerint ABC
üres lapok összetűzve

Minta ZH / 6

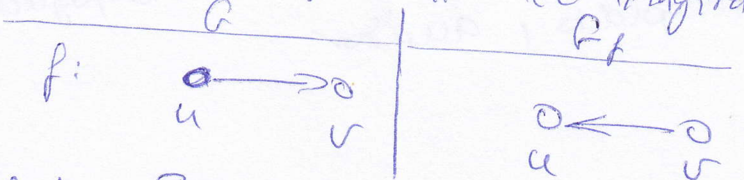
ZH-ban is az a követendő út:
meglévő algoritmust használjuk
módosított [redacted] inputtal
ne azt algoritmust módosítsuk

$O(e(n+e))$

↳ van idő arra, hogy minden élhez lefuttassunk egy BFS-t

minden f élre: megfordítjuk az él irányítását

$G \to G_f$:



ebben futtatni BFS(G_f, s) → ez megadja, hogy minden csúcs elérhető-e

* 1 db él megfordítása:

eredeti éllista: L

új éllista: L'

minden x-re: $L'[x] = L[x]$ ha $x \neq u \wedge v$

ha $x = u$: $L'[u] = L[u] \setminus \{v\}$

ha $x = v$: $L'[v] = L[v] \cup \{u\}$

LSZ:

műveleti LSZ: $n + e$

IFS: éllistás megadás esetén: $n+e$ } $e(n+e)$

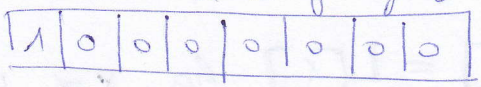
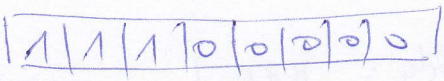
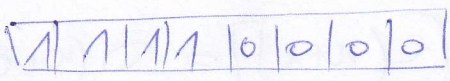
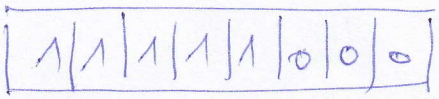
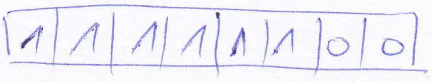
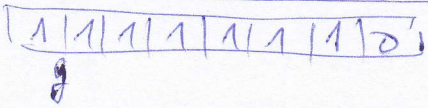
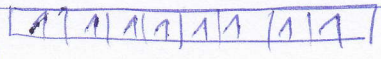
élszám: e

$$O(n+e) + O[e(n+e)] = O(n+e)$$

Helyesség: ... ugyanaz 1 mondat ...

2./4

a	b c
b	a d
c	a d
d	b c e f
e	a f g
f	a e g h
g	e f h
h	f g

bejárva	Q állapota	T : új elemei
a b c d e f g h 	{a}	∅
a	{b, c}	(a,b), (a,c)
		
b	{c, d}	(b, d)
		
c	{d}	—
		
d	{e, f}	(d, e), (d, f)
		
e	{f, g}	(e, g)
		
f	{g, h}	(f, h)
		
g	{h}	—
h 	∅	—

MINTA ZH / (5.)

Algo.: 1.) Rendezés összefésüléses rendezéssel

2.) minden A[i]-re megnézni
hogy A[i]+42 szerepel-e a tömbben
bináris kereséssel

LSZ.: 1.) $O(n \log(n))$

2.) $n \cdot O(\log n) = O(n \log(n))$

Helyesség:

2.) bináris keresés rendezett tömbben
használható

Akkor van 2 jó szám, ha $\exists i$ amire
A[i] és A[i]+42 is eleme a tömbnek