11. előadás

Hierarchikus lista, háló és rekord

A hierarchikus lista, a háló és a rekord

Adatszerkezetek és algoritmusok előadás
2011. április 27.

Kósa Márk és Pánovics János
Debreceni Egyetem
Informatikai Kar

A hierarchikus lista adatszerkezet

A szekvenciális lista adatszerkezet általánosításának tekinthető: a lista elemei maguk is lehetnek listák. A korábban említett összes listaművelet értelmezhető rajta. Fontos szerepet játszik a LISP programozási nyelvben (S-kifejezés).

Példa

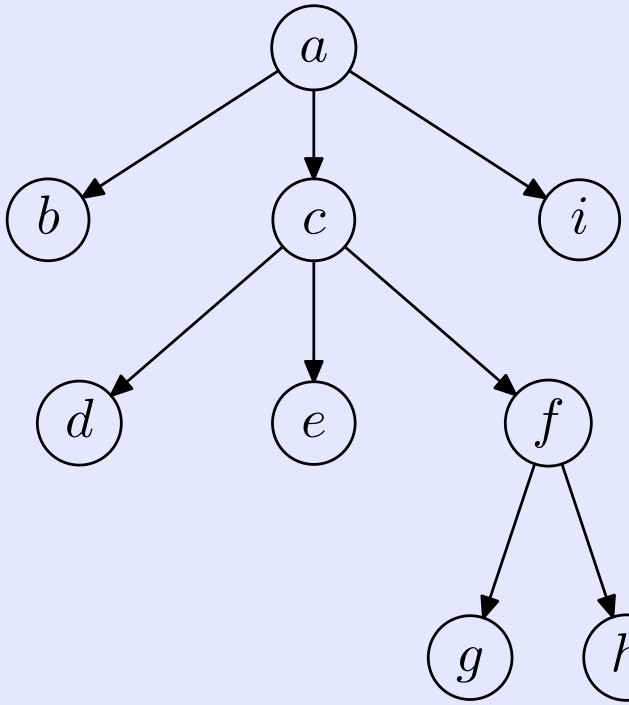
(ez egy ötelemű lista (melynek utolsó eleme egy (ötelemű lista)))

Alkalmazható rendezett (nem bináris) fa reprezentálására: a lista feje a gyökérelemet írja le, a lista farkát pedig a gyökérelemből kiinduló részfákat reprezentáló beágyazott listák alkotják. A reprezentációt a fa preorder bejárásával kaphatjuk meg. A rendezett többágú fák műveletei értelmezhetők rajta.

Kósa Márk
Pánovics János

Rendezett fa reprezentálására hierarchikus listával

Példa



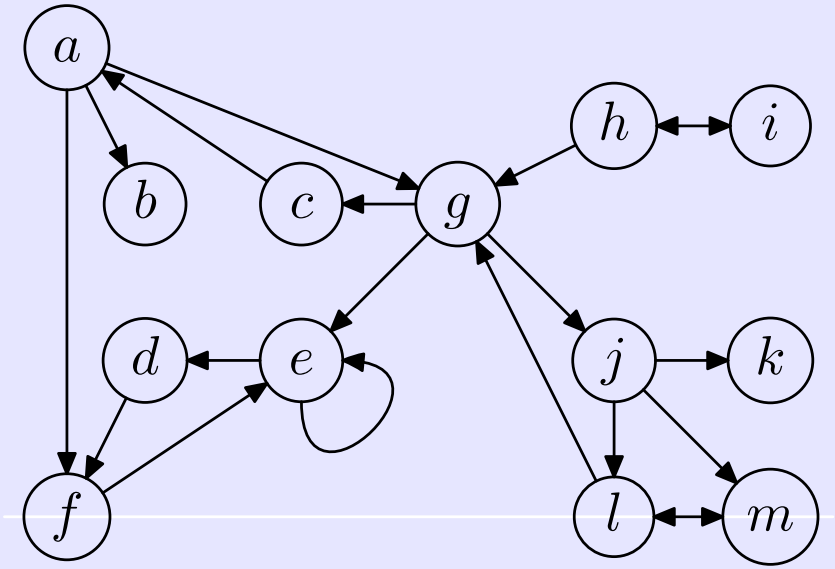
(a (b) (c (d) (e) (f (g) (h))) (i))

Kósa Márk
Pánovics János

Hálós adatszerkezetek

Minden adatelemnek tetszőleges számú megelőzője és tetszőleges számú rákövetkezője lehet. Egy elem lehet egy másik elem megelőzője, rákövetkezője, mindkettő vagy egyik sem. Egy elem lehet saját magának a megelőzője, illetve rákövetkezője. Ilyen adatszerkezet a háló, amely a matematikai irányított gráf struktúrának felel meg.

Példa



Kósa Márk
Pánovics János

Hálóval végezhető műveletek

A háló dinamikus és homogén adatszerkezet.

Hálóval végezhető műveletek

- Létrehozás: üres hálót hozunk létre.
- Bővítés: az új elem értéke mellett meg kell adni a megelőzőinek és a rákövetkezőinek a listáját is.
- Törlés: fizikai. Érinti azokat az elemeket is, amelyekkel a törölt elem szomszédsági viszonyban állt.
- Csere: megoldható.
- Rendezés: nem értelmezett.
- Keresés, elérés, feldolgozás: a bejárás alapján.
- Bejárás: szélességi vagy mélységi.

A háló bejárása

Kiválasztunk egy tetszőleges elemet (S), és ebből kiindulva térképezzük fel a háló S-ből elérhető elemeit. Felépítünk egy S gyökerű feszítőfát (szélességi vagy mélységi fa). Ha S-ből nem érhető el a háló összes eleme, akkor a maradék elemekből újra kiválasztunk egyet, és újabb feszítőfát építünk fel. Hogy minél kevesebb feszítőfa jöjjön létre, célszerű S-nek azt az elemet választani, amelyik a legtöbb rákövetkezővel rendelkezik.

A háló elemeit háromféle színnel látjuk el:

- fehér: még nem elért elemek,
- szürke: elért elemek, amelyek rákövetkezőit még nem értük el,
- fekete: elért elemek, amelyek rákövetkezőit is elértük már.

A két algoritmus abban különbözik egymástól, hogy szélességi bejárás esetén a legkorábban, mélységi bejárás esetén pedig a legkésőbb elért szürke elemből kiindulva folytatjuk a feszítőfa építését.

Kósa Márk
Pánovics János

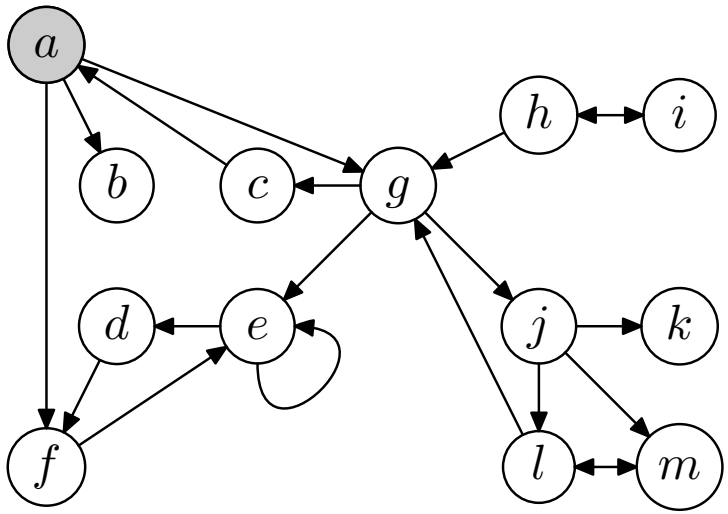
Szélességi bejárás

A bejárás során az adatelemeket fehér, szürke és fekete színűre fogjuk színezni, valamint egy sort fogunk használni a szürke színű adatelemek tárolására. A bejárás során a feldolgozott elemekből feszítőfá(ka)t építünk fel.

1. Színezzük a háló adatelemeit fehér színűre, és hozzuk létre az üres sort.
2. Ha minden adatelem fekete színű, a bejárás véget ér.
3. Ha a sor üres, válasszunk tetszőlegesen egy fehér színű elemet, színezzük szürkére, és helyezzük el a sorban. Ennek az elemnek a feldolgozásakor új feszítőfát fogunk elkezdeni építeni.
4. Ha a sorban van elem, vegyük ki a sor első elemét, fehér színű gyermekeit szürkére festve helyezzük el a sorban, majd az adatelemet fessük feketére, és helyezzük el a megfelelő feszítőfában.
5. Folytassuk az algoritmust a 2. lépéssel.

Kósa Márk
Pánovics János

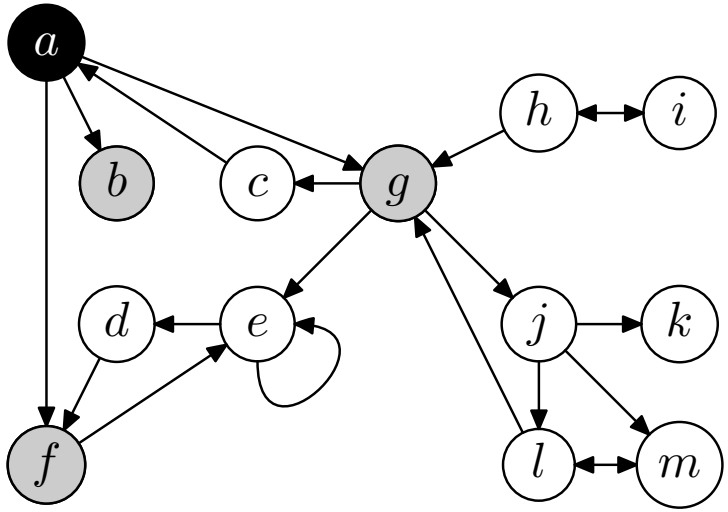
Példa szélességi bejárásra

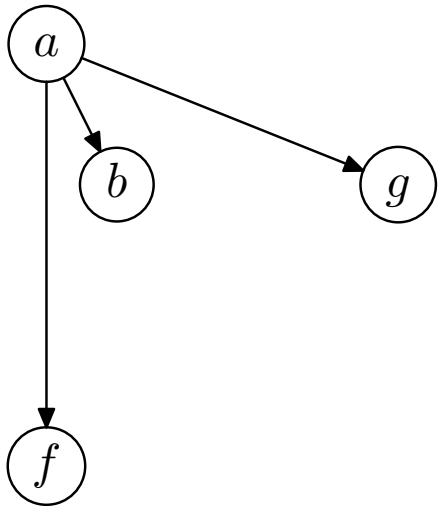


Sor: a

Kósa Márk
Pánovics János

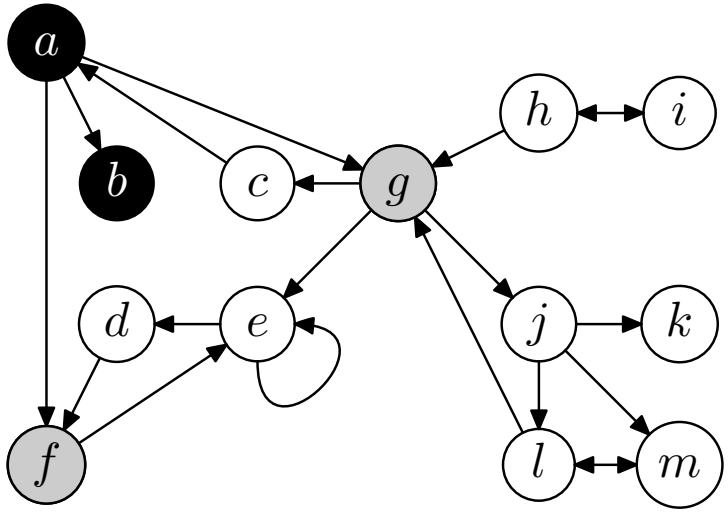
Példa szélességi bejárásra

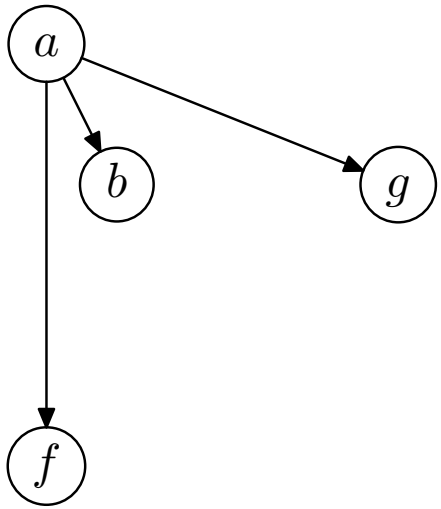




Sor: b, f, g

Példa szélességi bejárásra

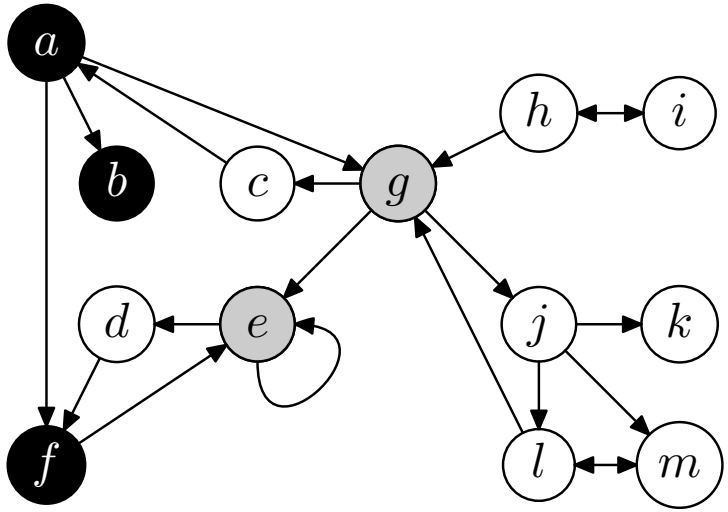


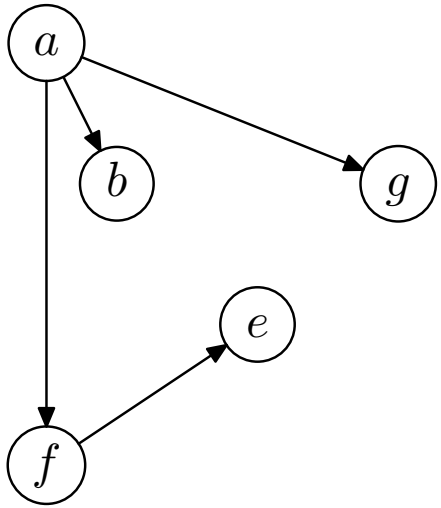


Sor: f, g

Kósa Márk
Pánovics János

Példa szélességi bejárásra

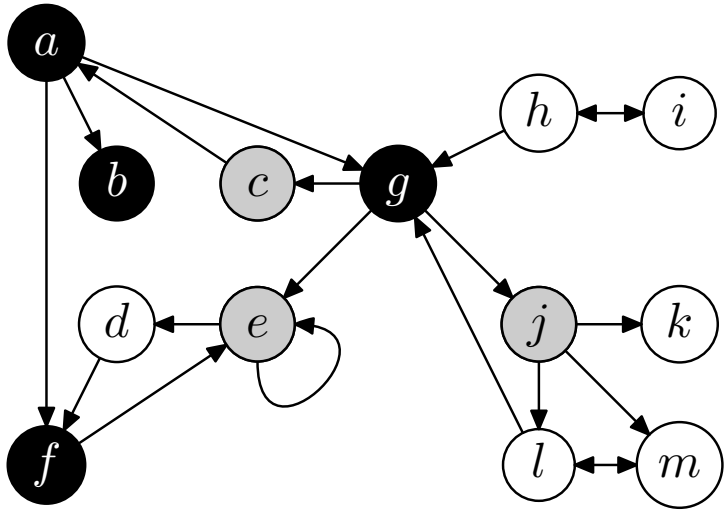


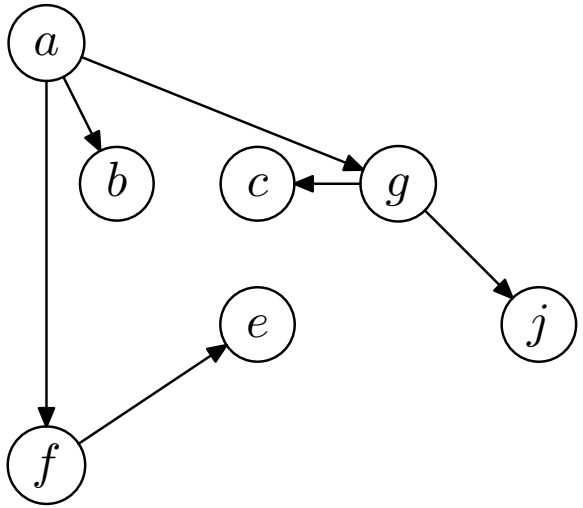


Sor: g, e

Kósa Márk
Pánovics János

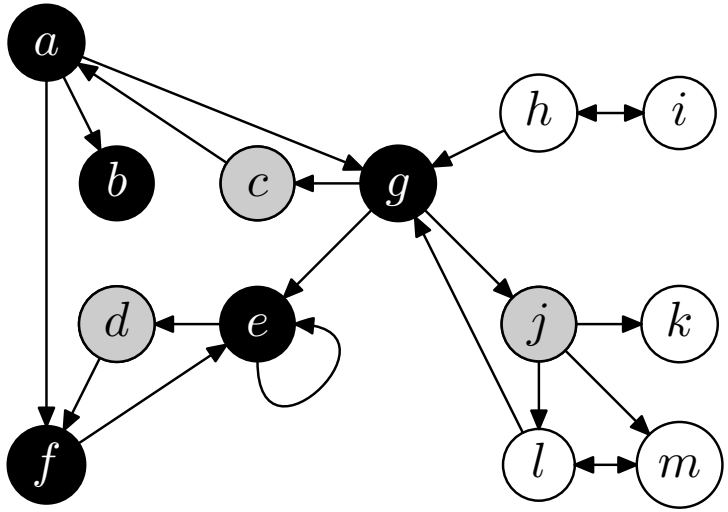
Példa szélességi bejárásra

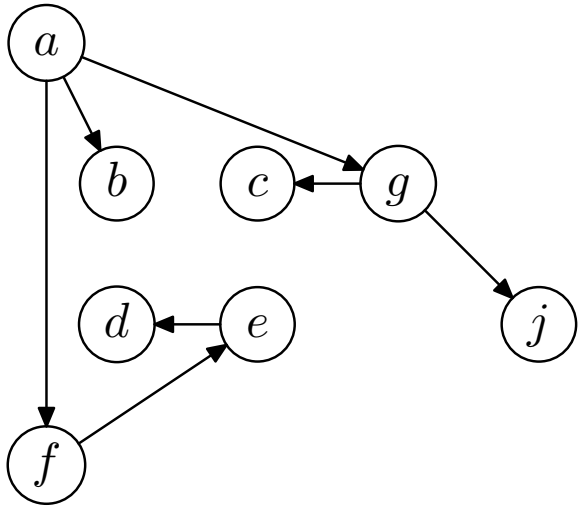




Sor: e, c, j

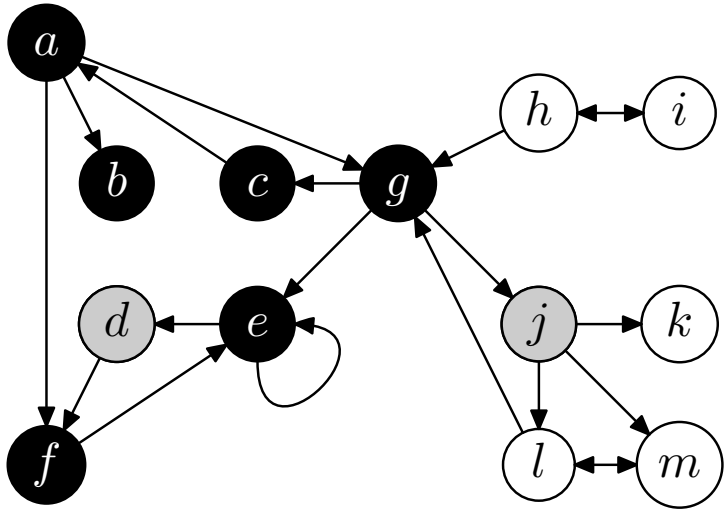
Példa szélességi bejárásra

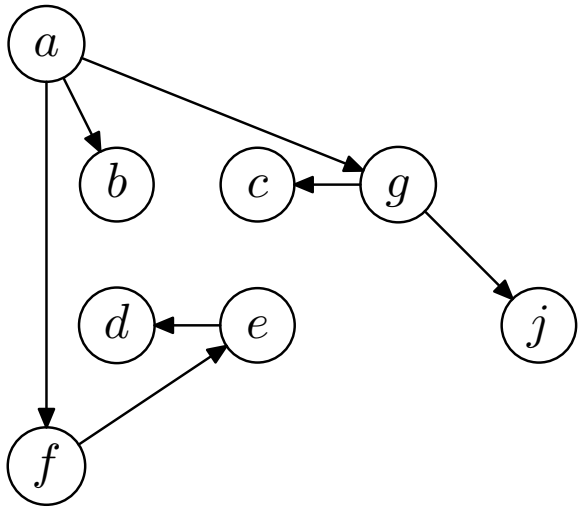




Sor: c, j, d

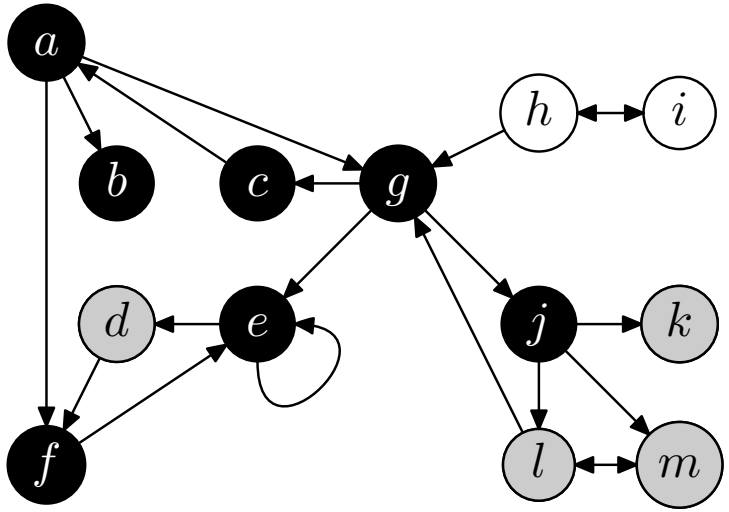
Példa szélességi bejárásra

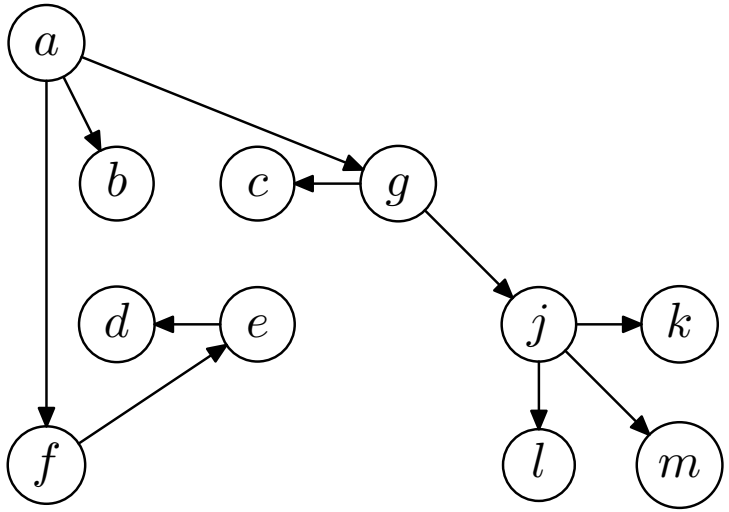




Sor: j, d

Példa szélességi bejárásra

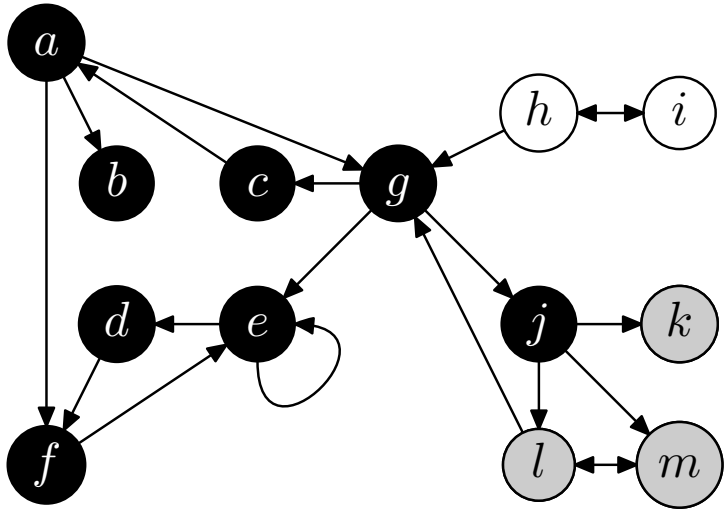


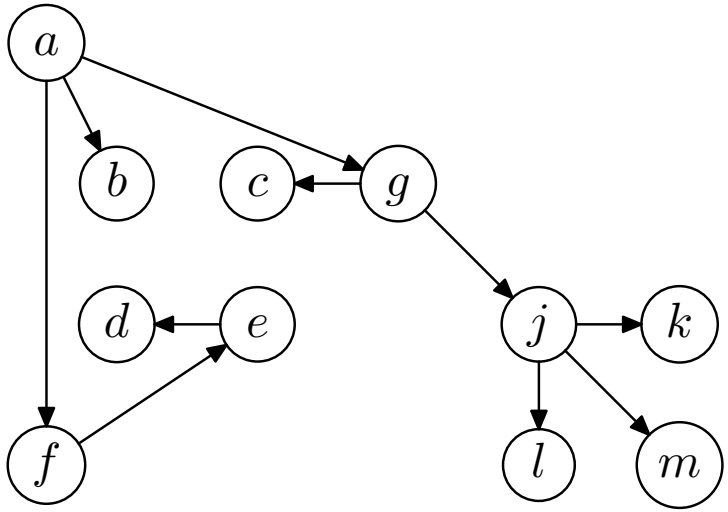


Sor: d, k, l, m

Kósa Márk
Pánovics János

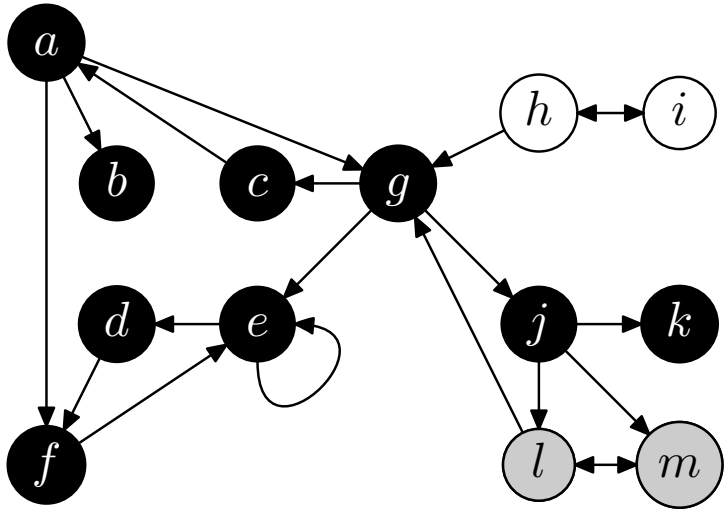
Példa szélességi bejárásra

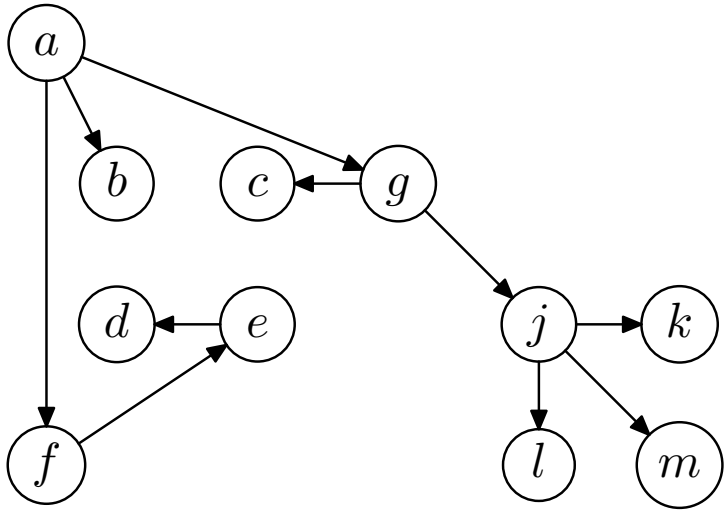




Sor: k, l, m

Példa szélességi bejárásra

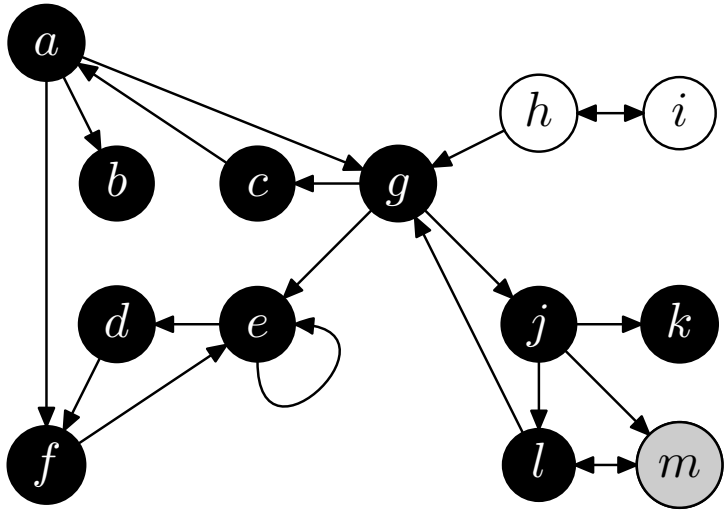


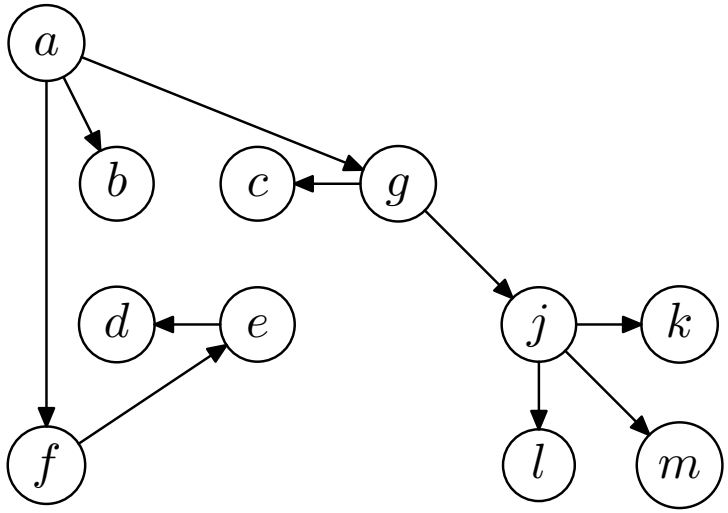


Sor: l, m

Kósa Márk
Pánovics János

Példa szélességi bejárásra

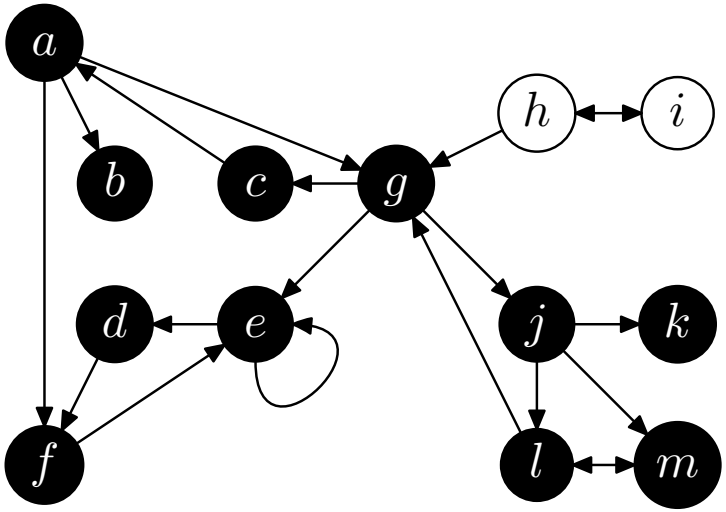


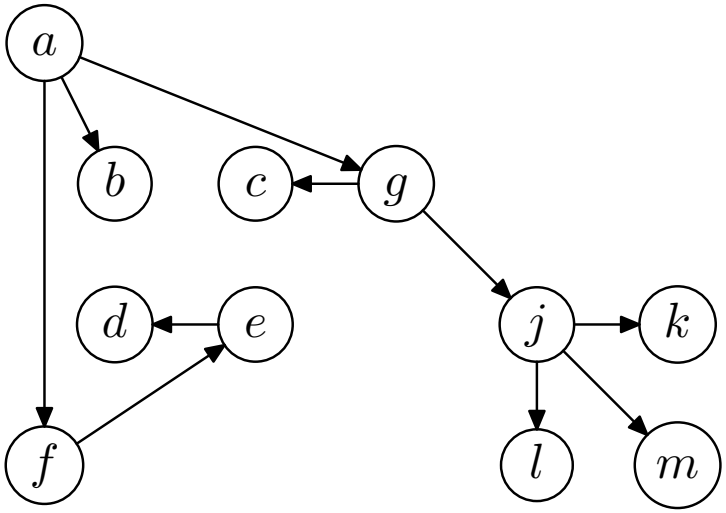


Sor: m

Kósa Márk
Pánovics János

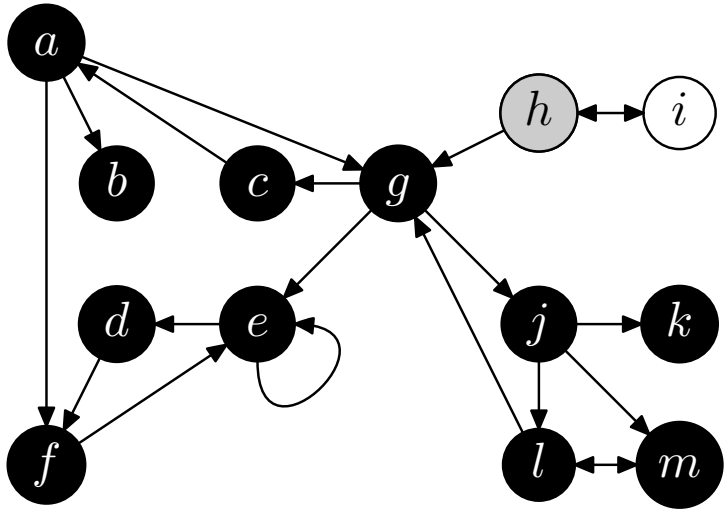
Példa szélességi bejárásra

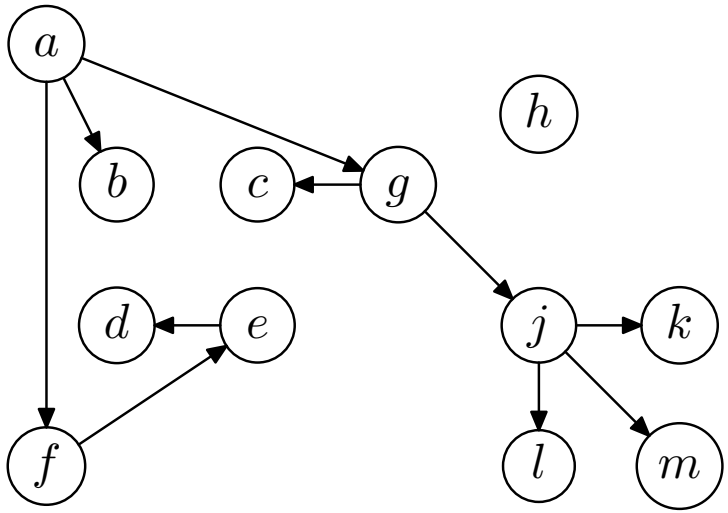




Sor: –

Példa szélességi bejárásra

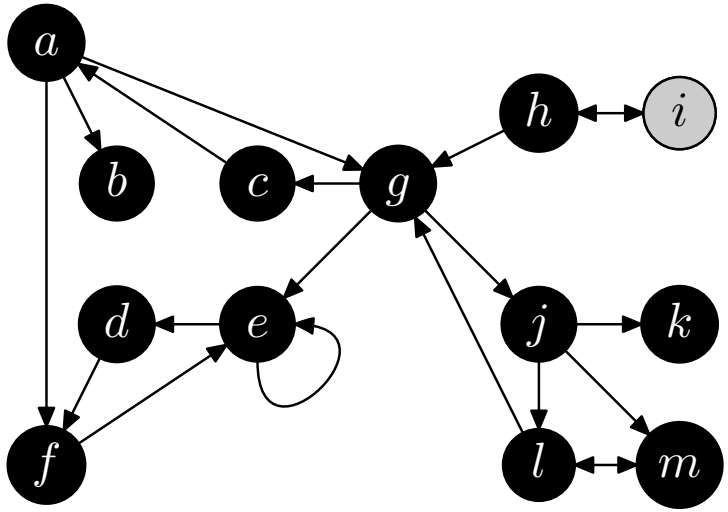


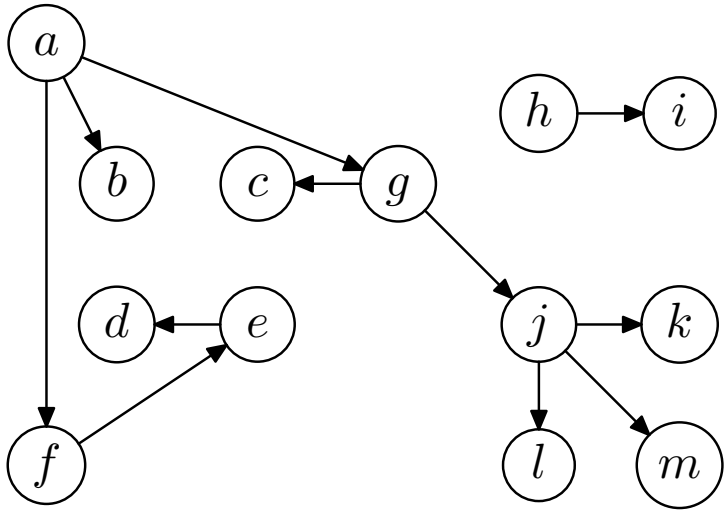


Sor: h

Kósa Márk
Pánovics János

Példa szélességi bejárásra

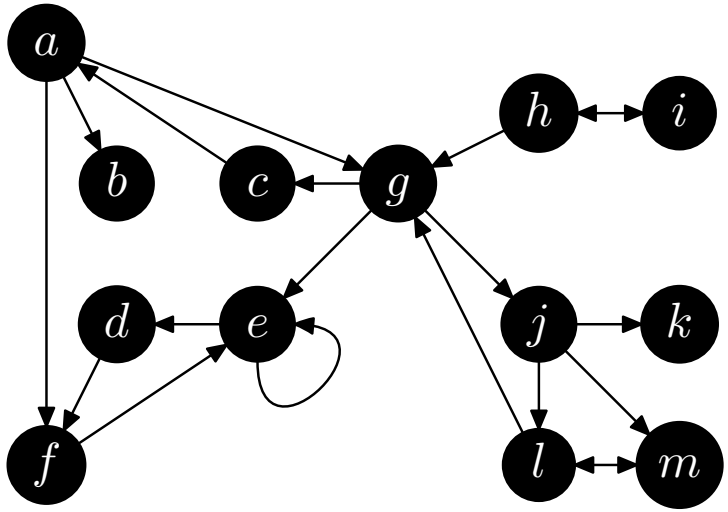


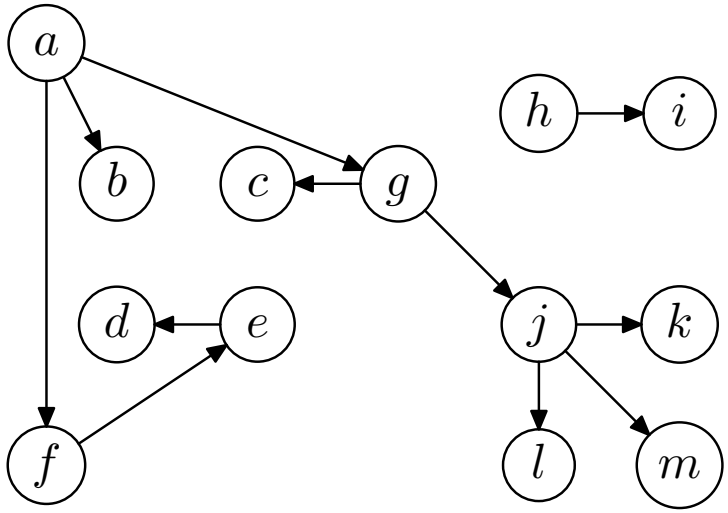


Sor: i

Kósa Márk
Pánovics János

Példa szélességi bejárásra





Sor: –

Kósa Márk
Pánovics János

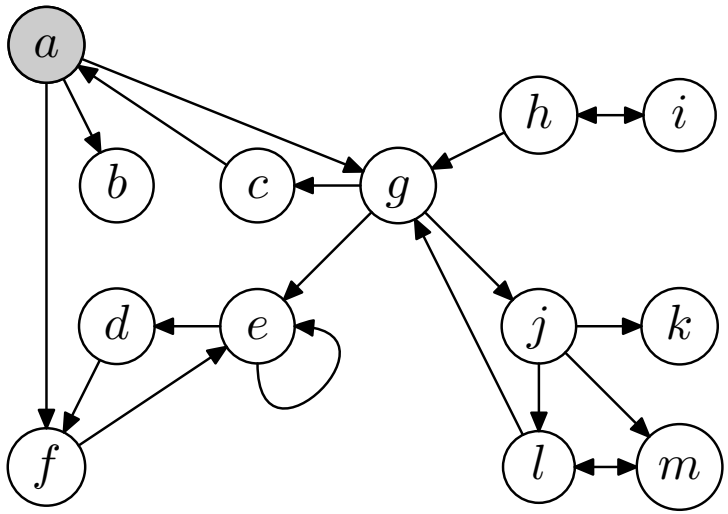
Mélységi bejárás

A bejárás során az adatelemeket fehér, szürke és fekete színűre fogjuk színezni, valamint egy vermet fogunk használni a szürke színű adatelemek tárolására. A bejárás során a feldolgozott elemekből feszítőfá(ka)t építünk fel.

1. Színezzük a háló adatelemeit fehér színűre, és hozzuk létre az üres vermet.
2. Ha minden adatelem fekete színű, a bejárás véget ér.
3. Ha a verem üres, válasszunk tetszőlegesen egy fehér színű elemet, színezzük szürkére, és helyezzük el a veremben. Ennek az elemnek a feldolgozásakor új feszítőfát fogunk elkezdeni építeni.
4. Ha a veremben van elem, vegyük ki a verem első elemét, fehér színű gyermekeit szürkére festve helyezzük el a veremben, majd az adatelemet fessük feketére, és helyezzük el a megfelelő feszítőfában.
5. Folytassuk az algoritmust a 2. lépéssel.

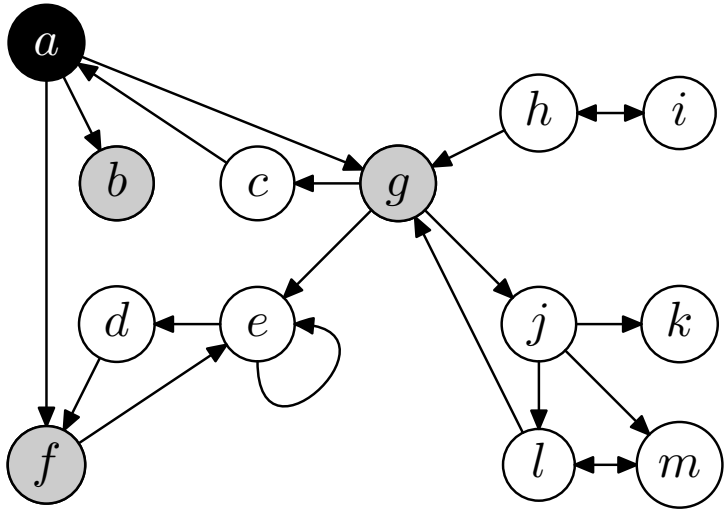
Kósa Márk
Pánovics János

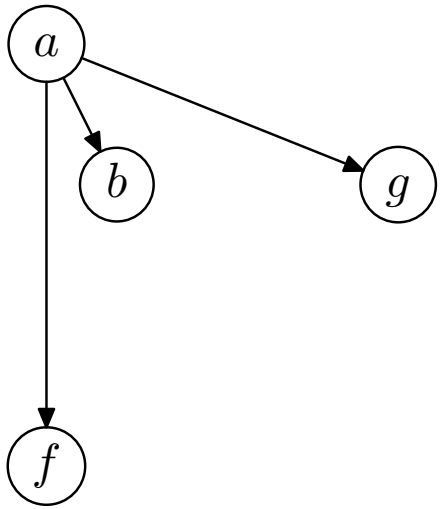
Példa mélységi bejárásra



Verem: a

Példa mélységi bejárásra

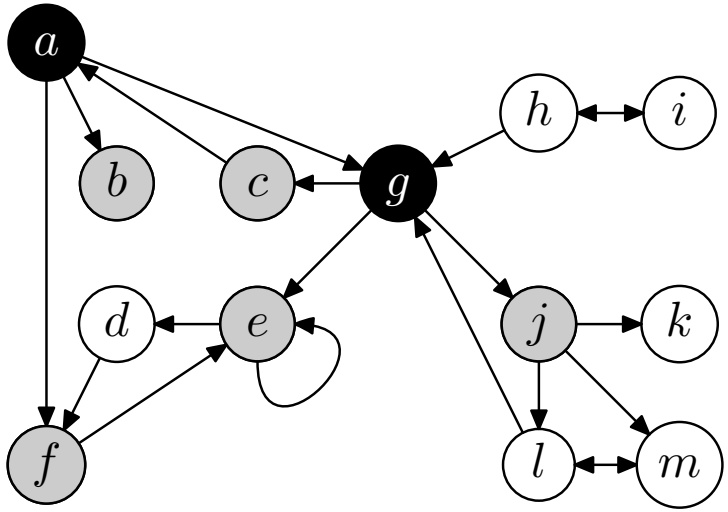


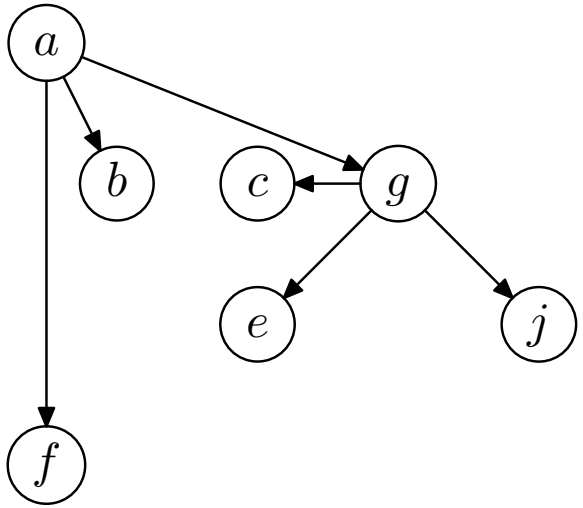


Verem: b, f, g

Kósa Márk
Pánovics János

Példa mélységi bejárásra

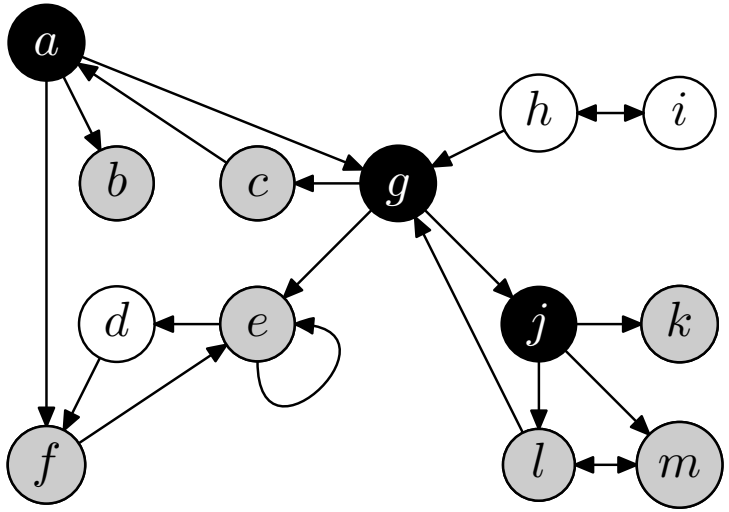


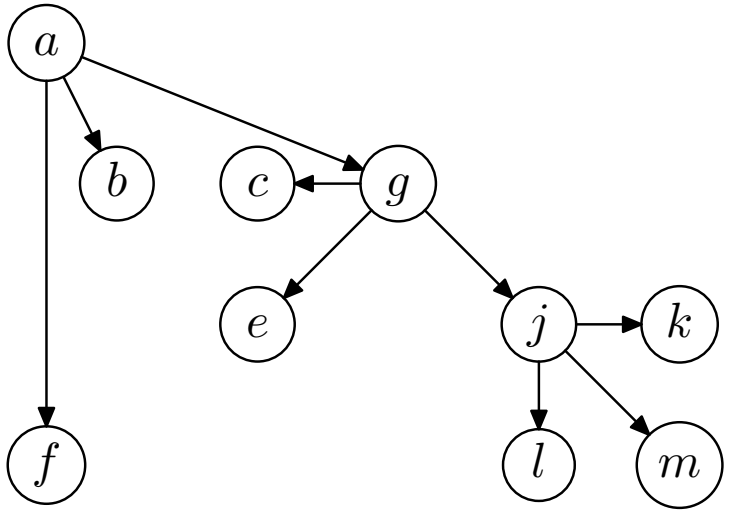


Verem: b, f, c, e, j

Kósa Márk
Pánovics János

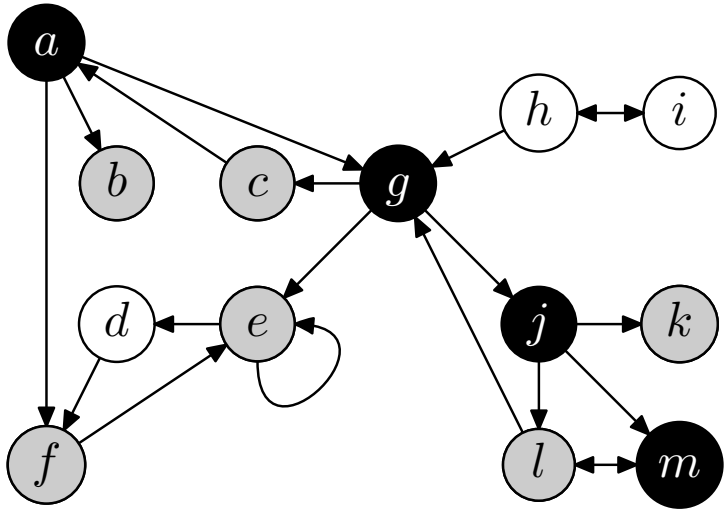
Példa mélységi bejárásra

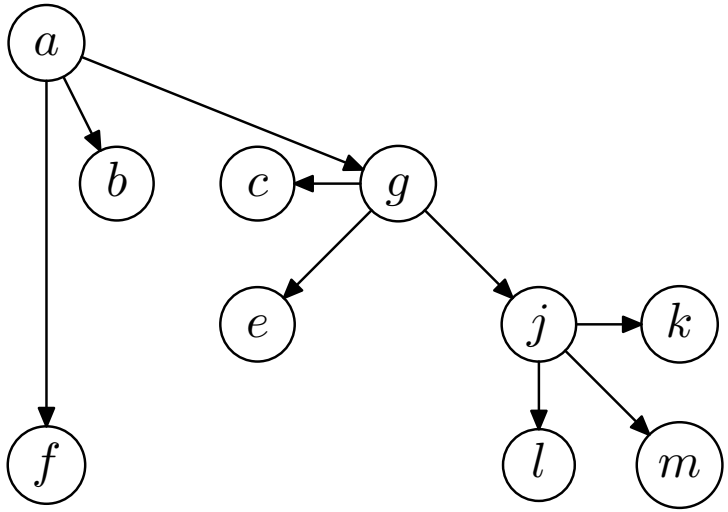




Verem: b, f, c, e, k, l, m

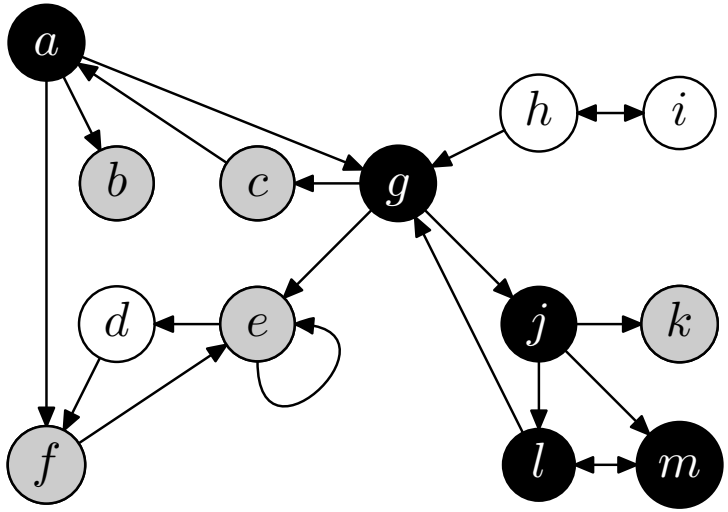
Példa mélységi bejárásra

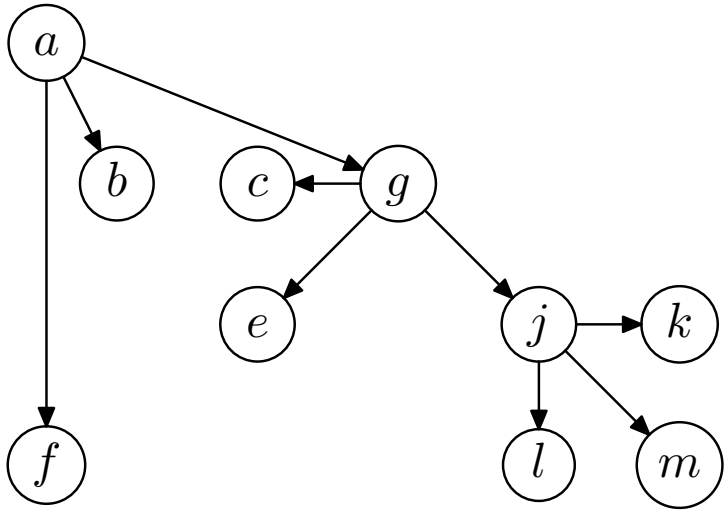




Verem: b, f, c, e, k, l

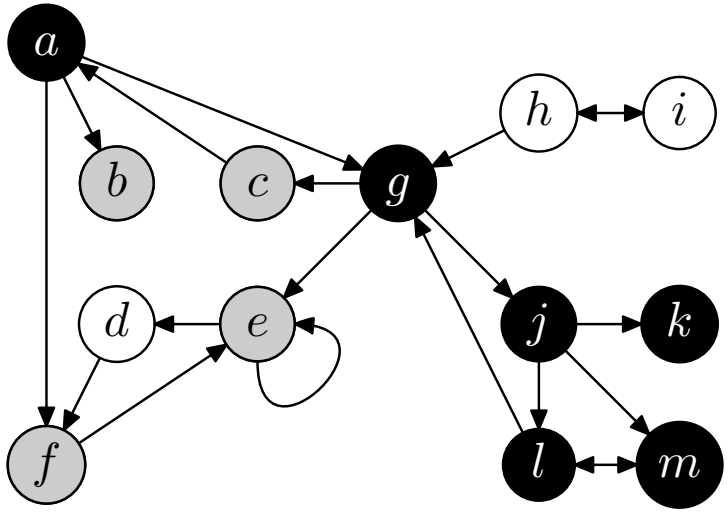
Példa mélységi bejárásra

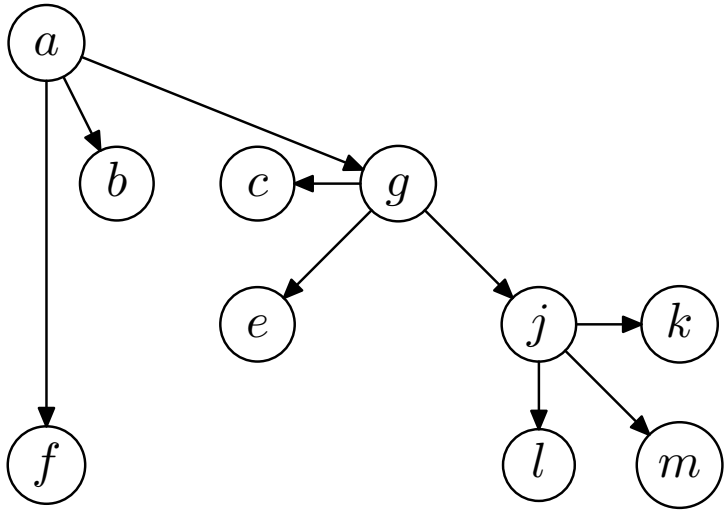




Verem: b, f, c, e, k

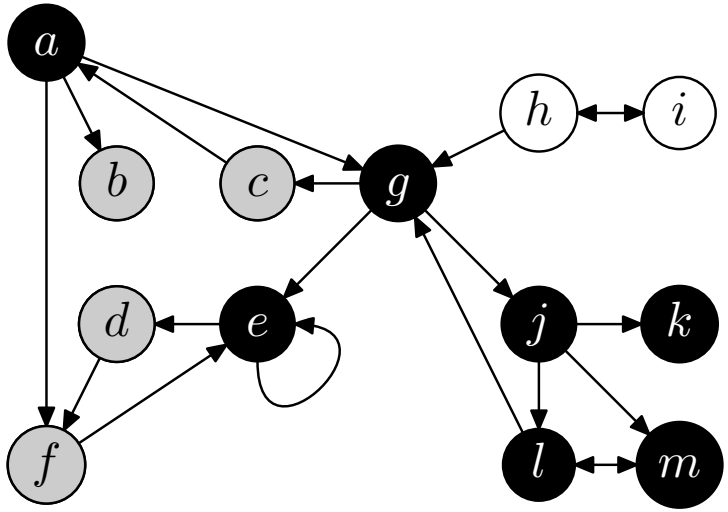
Példa mélységi bejárásra

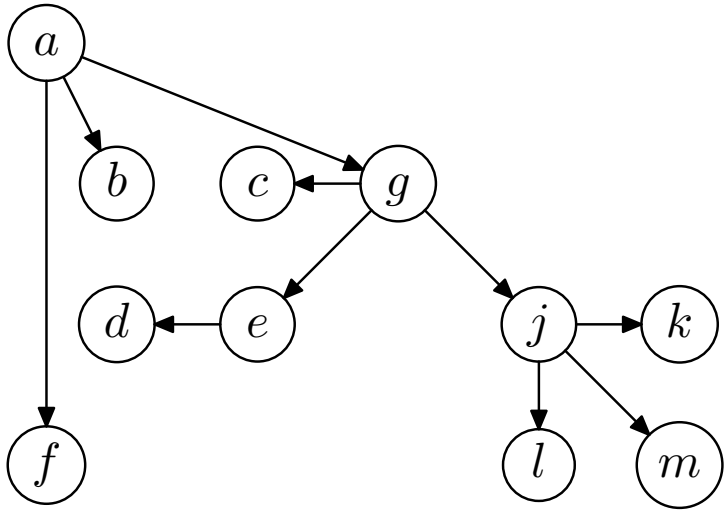




Verem: b, f, c, e

Példa mélységi bejárásra

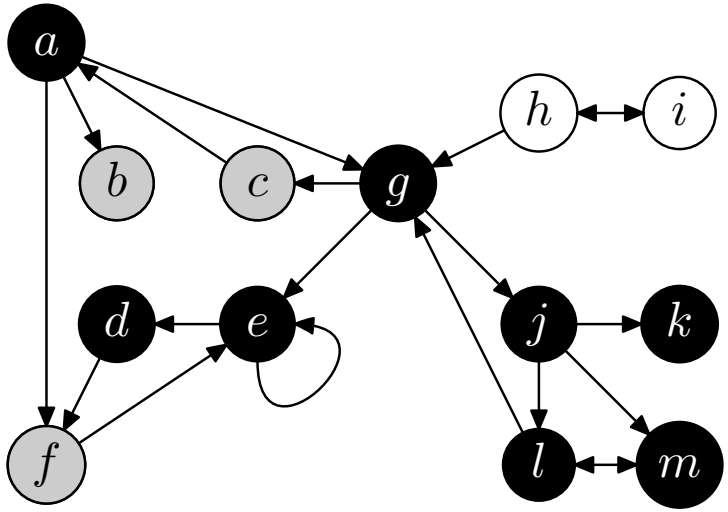


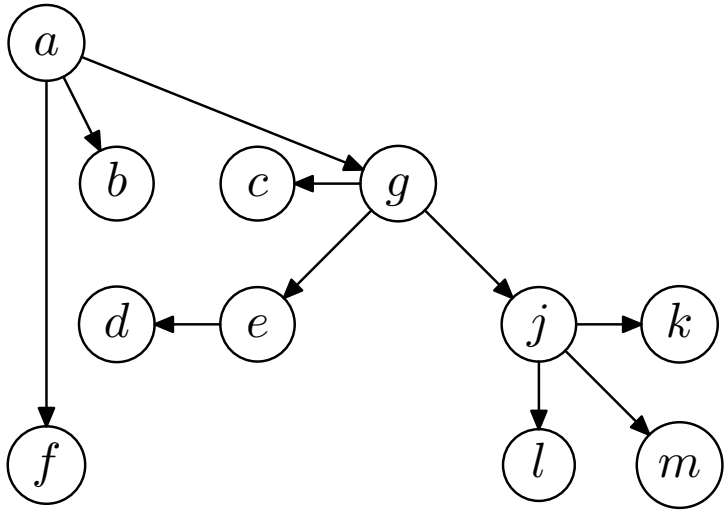


Verem: b, f, c, d

Kósa Márk
Pánovics János

Példa mélységi bejárásra

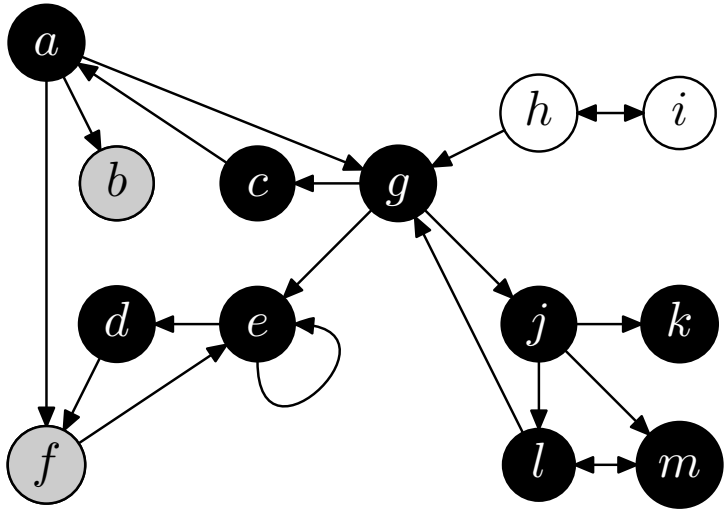


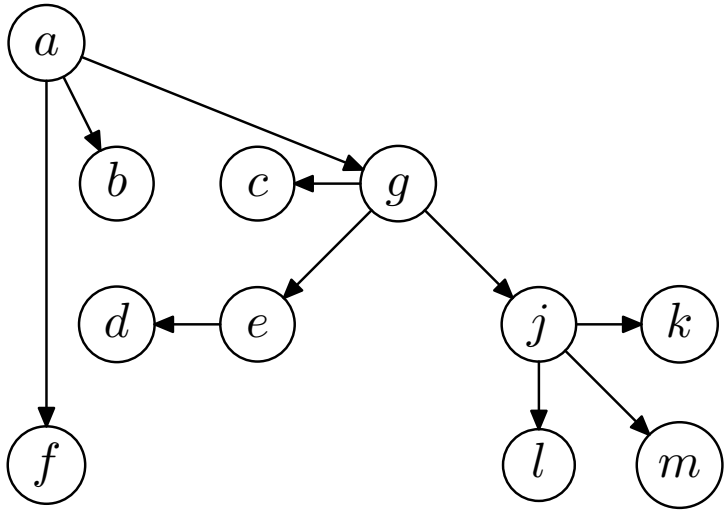


Verem: b, f, c

Kósa Márk
Pánovics János

Példa mélységi bejárásra

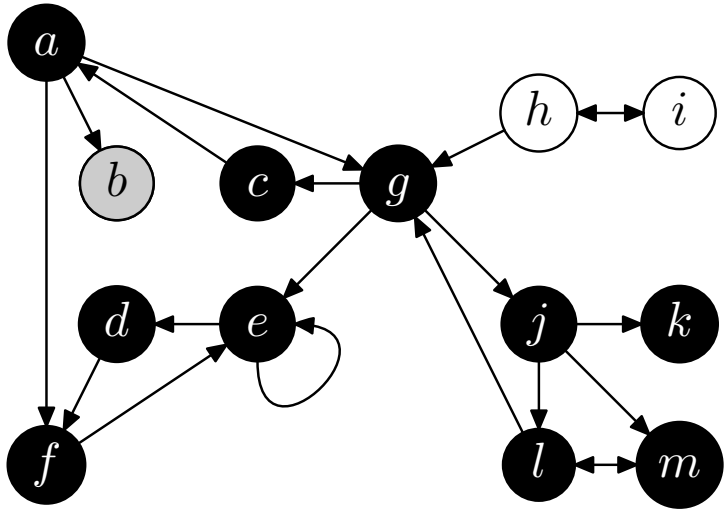


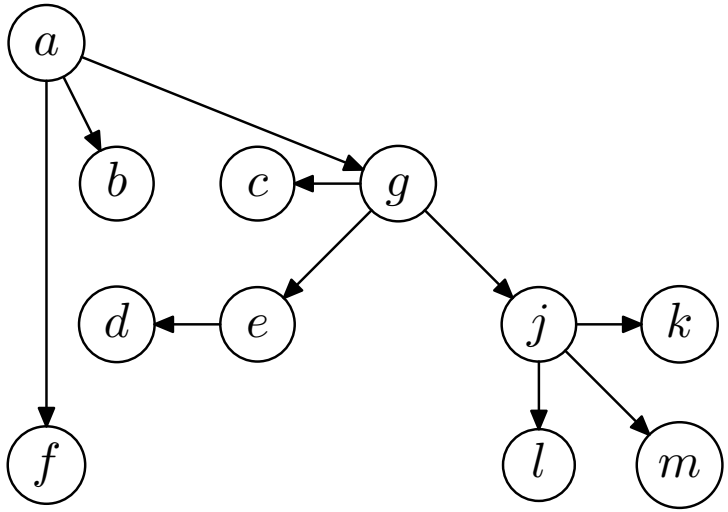


Verem: b, f

Kósa Márk
Pánovics János

Példa mélységi bejárásra

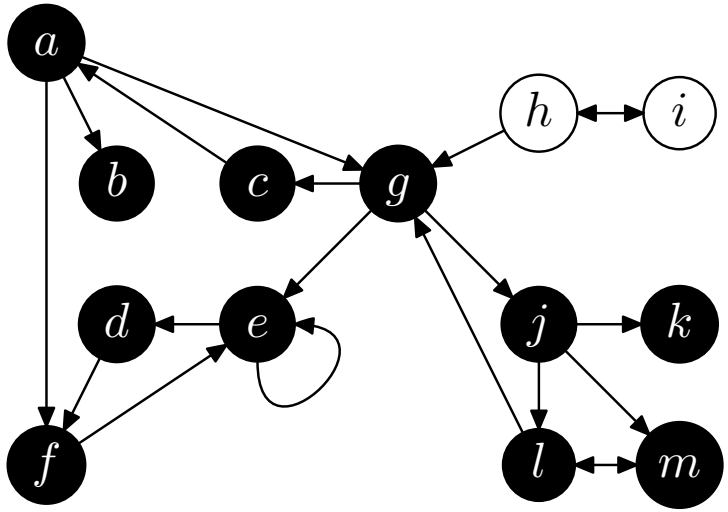


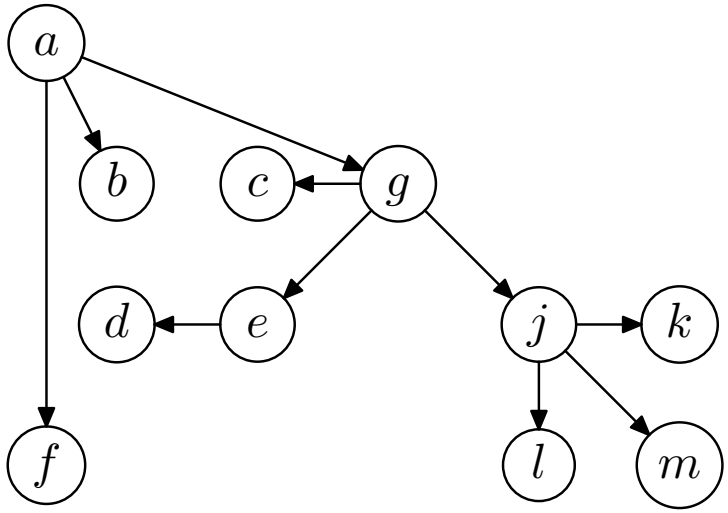


Verem: b

Kósa Márk
Pánovics János

Példa mélységi bejárásra

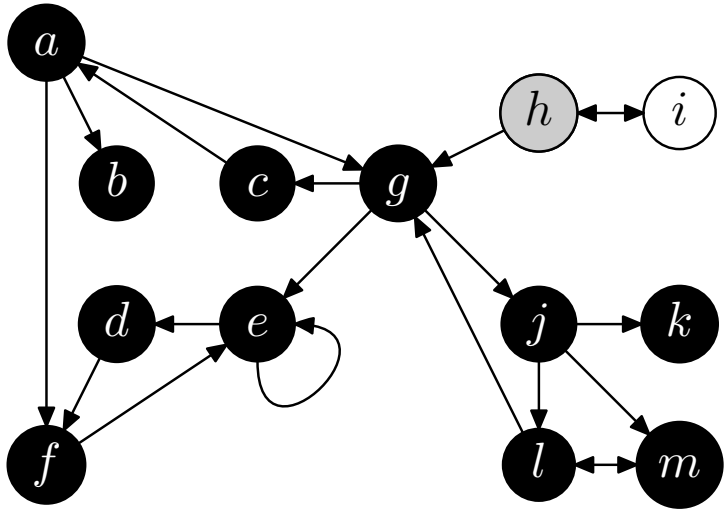


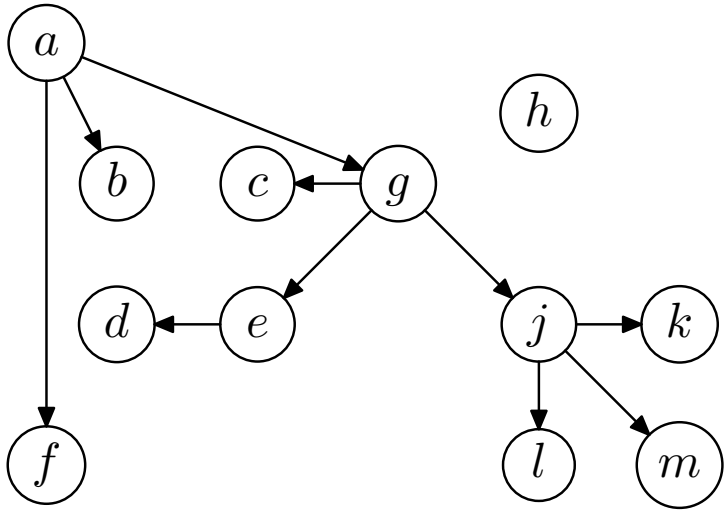


Verem: –

Kósa Márk
Pánovics János

Példa mélységi bejárásra

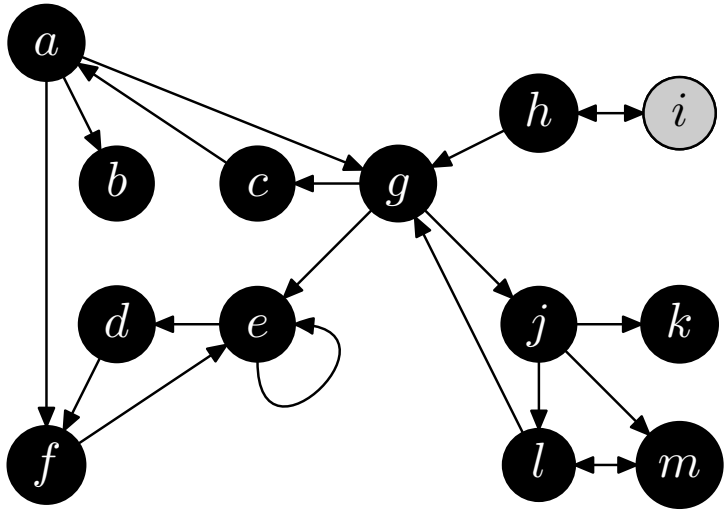


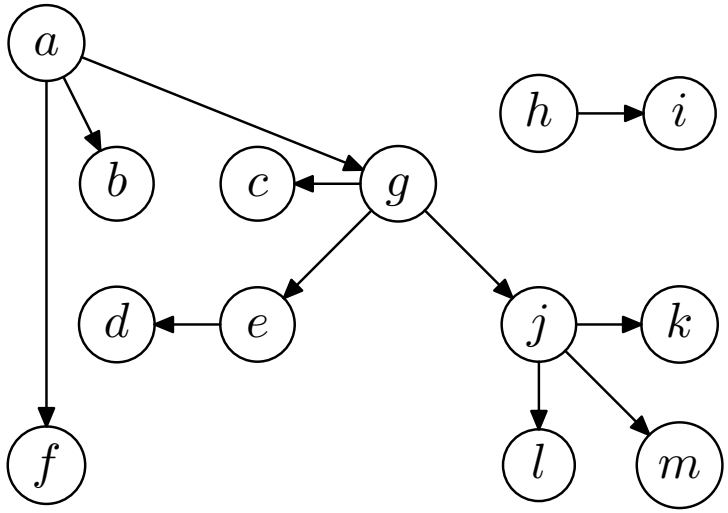


Verem: h

Kósa Márk
Pánovics János

Példa mélységi bejárásra

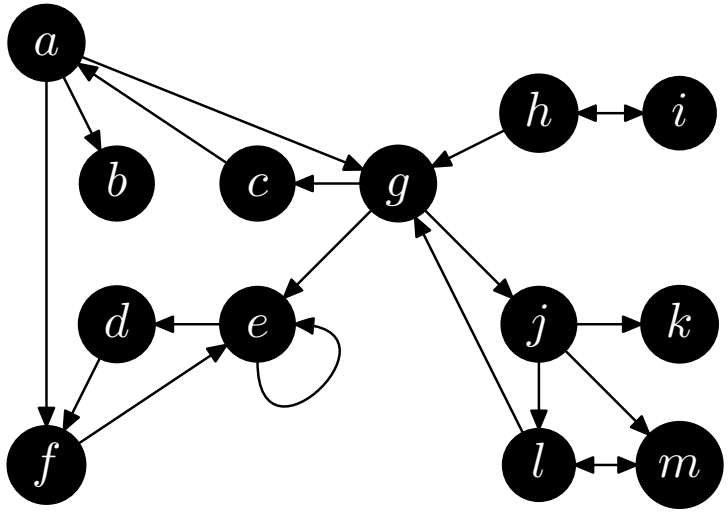


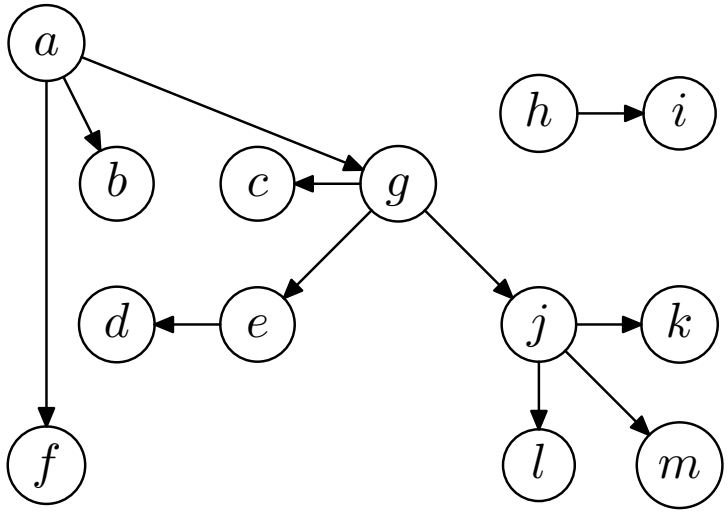


Verem: i

Kósa Márk
Pánovics János

Példa mélységi bejárásra





Verem: –

Kósa Márk
Pánovics János

A háló reprezentációja

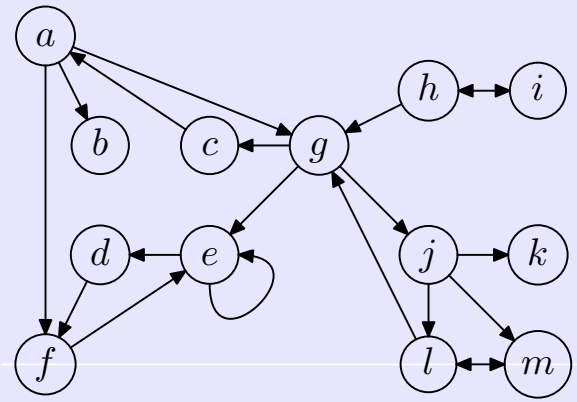
A háló folytonosan szomszédsági mátrixszal (csúcsmátrixszal) reprezentálható. Egy n adatelemből álló háló esetén ez egy $n \times n$-es logikai mátrix, amelynek a sorait és oszlopait az elemekkel címkézzük. Ha egy sor címkéjében szereplő elem megelőz egy oszlop címkéjében szereplő elemet, akkor az adott sor és oszlop metszetébe 1-et írunk, különben 0-t. A szomszédsági mátrixban egy adott elem megelőzőit az elem oszlopából, a rákövetkezőit pedig az elem sorából tudjuk kiolvasni.

Szétszórt reprezentáció is lehetséges, mégpedig multilistával. Minden adatelem mellett megjelenik egy mutatótömb, ami annyi elemű, ahány rákövetkezője lehet maximálisan egy elemnek. Ez az érték legfeljebb annyi, ahány elemből a háló állhat. Ha nem szeretnénk korlátozni a rákövetkezők számát, akkor a mutatókat elhelyezhetjük egy dinamikus vektorban, de felfűzhetjük őket egy egyirányban láncolt listába is.

Kósa Márk
Pánovics János

A háló reprezentációja

Példa



	a	b	c	d	e	f	g	h	i	j	k	l	m
a	0	1	0	0	0	1	1	0	0	0	0	0	0
b	0	0	0	0	0	0	0	0	0	0	0	0	0
c	1	0	0	0	0	0	0	0	0	0	0	0	0
d	0	0	0	0	0	1	0	0	0	0	0	0	0
e	0	0	0	1	1	0	0	0	0	0	0	0	0
f	0	0	0	0	1	0	0	0	0	0	0	0	0
g	0	0	1	0	1	0	0	0	0	1	0	0	0
h	0	0	0	0	0	0	1	0	1	0	0	0	0
i	0	0	0	0	0	0	0	1	0	0	0	0	0
j	0	0	0	0	0	0	0	0	0	0	1	1	1
k	0	0	0	0	0	0	0	0	0	0	0	0	0
l	0	0	0	0	0	0	1	0	0	0	0	0	1
m	0	0	0	0	0	0	0	0	0	0	0	1	0

Kósa Márk
Pánovics János

Gráfelméleti kérdések

Gráfelméleti fogalmak:

- Út: olyan elemek összessége a gráfban, amelyek listát alkotnak.
- Körút: olyan út, amelyben az első elem az utolsó elem rákövetkezője.

Gráfelméleti problémák:

- Annak eldöntése, hogy létezik-e út két adott csomópont között.
- Legrövidebb út keresése két adott csomópont között.
- Annak meghatározása, hogy egy adott csomópontból mely más csomópontokba vezet út.
- Maximális út (a gráf összes csomópontját tartalmazó út) keresése.
- Körút létezésének eldöntése, a leghosszabb körút keresése.

Kósa Márk
Pánovics János

A rekord adatszerkezet

Statikus és heterogén adatszerkezet. Mezőkből áll, ezek sorrendje kötött, mindegyiknek saját neve van.

A rekord atatszerkezet műveletei

- A mezők száma létrehozáskor dől el. Tárhelyeket foglalunk le, melyeket a mezőnevekkel azonosítunk, és az egyes tárhelyekre kezdőértékeket helyezhetünk el.
- Bővítés nincs, a létrehozáskor kialakított mezők száma nem változik. A létrehozáskor üresen hagyott mezők azonban – természetesen – kaphatnak értéket (lásd csere).
- Csere: a mezőnév megadásával a hozzá tartozó érték bármikor cserélhető.
- A törlés csak logikai lehet.
- Az elérés közvetlen, a mezőnevek segítségével.
- A rendezés, a keresés és a bejárás nem értelmezett.
- Feldolgozás: a mezőnevek alapján.

Kósa Márk
Pánovics János

A rekord adatszerkezet

Megjegyzések:

- A rekord reprezentációja általában folytonos.
- A legtöbb programozási nyelv típus szinten közvetlenül támogatja a rekord adatszerkezetet. Egyes programozási nyelvekben a rekord típus dinamikusan kezelhető. A rekordnak ilyenkor van egy fix és egy változó része. A változó rész elemeit (mezőit) általában egy diszkriminátor értéke alapján határozzuk meg, ami a rekord fix részének az egyik mezője.
- A rekord adatszerkezet nem játszik fontos szerepet a memóriában tárolt adatszerkezeteknél, alapvető szerepe van viszont az állományoknál.

Kósa Márk
Pánovics János

Memóriában tárolt adatszerkezetek – végszó

Az egyes adatszerkezetekben tárolt értékek lehetnek:

- atomiak vagy
- összetettek.

Megengedhető, hogy egy adatszerkezetben egy adatelem maga is egy adatszerkezet legyen. Ez az ortogonalitás elve: bármelyik adatszerkezet adateleme lehet bármilyen adatszerkezet.

Kósa Márk
Pánovics János