1. tétel

Az optimális hozzárendelés problémája, Egerváry algoritmusa.

## Magyar módszer

A módszer: maximális párosítás keresése páros gráfban javító utak iterálásával. 

A módszer lényege, hogy javító utat keresünk, majd abban a szerepeket megcseréljük, így eggyel növekszik a párosítások száma. Ha már nincs ilyen út, akkor a párosítás maximális.

**Def.:** A javító út, olyan út, amely nem párosított A-ból indul és nem párosított B-be vezet, úgy, hogy minden második él párosításbeli.

**Def.:** Az alternáló út, olyan út, amely nem párosított A-ból indul és B-be vezet (lehet párosított és nem párosított is, ebben különbözik a javító úttól), úgy, hogy minden második él párosításbeli.

Javító út keresése: BFS algoritmussal. Folyamatosan lépkedünk A és B-beli pontok között,úgy, hogy egyszer M-beli, egyszer pedig nem M-beli éleken haladunk tovább.

### Bizonyítás

Helyesség bizonyítása: legyen a két pontosztály A és B, az algoritmus leállásakor talált párosítás M. Legyen az M által nem lefedett A-beli pontok halmaza U. Az U-ból alternáló elérhető B-beli pontok halmaza legyen T’. T’ A-beli szomszédjai (a párosítás szerint) T. Mivel az algoritmus leállt (nincs több javító út), ezért belátható, hogy T unió U minden szomszédja T’-ben van.

Tudjuk, hogy páros gráfban a maximális párosítás elemszáma és a minimális lefogó ponthalmaz darabszáma megegyezik (König-tétel). Ha vesszük T’-t, és A\(T unió U)-t, akkor egyrészt annyi pontot kaptunk, amennyi elem a talált párosításban van, másrészt lefogó ponthalmazt is találtunk (mert minden A-beli pontra illeszkedő élt, azaz valamennyi élt lefogtuk), ergo az algoritmus leállása után kapott párosítás maximális voltát beláttuk.

## Optimális hozzárendelés

páros gráf, élsúly-függvény. Keressük maximális összsúlyú *M* teljes párosítást: .

A maximális összsúlyú párosítás és a maximális összsúlyú teljes párosítás feladat nem ekvivalens.

különbség a kétféle optimum assignment probléma között

Pl. a gráfban az első esetben 3 az optimum, a második esetben 2.

Az első feladat visszavezethető a másodikra, ha az F,L csúcshalmazok közül a kisebbet kiegészítjük azonos elemszámúra, a hiányzó éleket behúzzuk 0 súllyal és a negatív éleket is kinullázzuk. Elég tehát a második feladatot megoldani.

## Címkézés definíció

páros gráf, címkézés (csúcs súlyfüggény), ha minden élre .

## Lemma

Rendeljünk a páros gráf minden v csúcsához egy címkét úgy, hogy minden élre teljesüljön. Ekkor a maximális összsúlyú teljes párosítás összsúlya legfeljebb .

### Bizonyítás

Legyen *M* egy tetszőleges teljes párosítás. Mivel *M* élei minden pontot egyszer fognak le, ezért

## Lemma

páros gráfban *w* élsúly-függvény, *c* címkézés, *M* teljes párosítás és minden   
 élre teljesül, akkor *M* maximális összsúlyú.

## Egerváry algoritmus

A gráfokat kiegészítjük úgy, hogy létezzen bennük teljes párosítás. Majd végrehajtjuk a következő lépéseket:

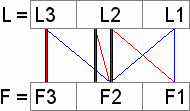
0. lépés: kezdeti címkézés. Legyen ; esetén *c(v)* a *v*-ből induló maximális élsúly, esetén .

1. lépés: *M*-ből kiindulva keresünk egy maximális elemszámú *M*’ párosítást javító utakkal a piros (teljesül rá a ) részgráfban. Ha *M*’ teljes párosítás, akkor STOP (*M’* a keresett párosítás, *c* a keresett címkézés).

2. lépés: Javítóutas algoritmus: *M’* élszáma növekszik vagy címkézés összege csökken. Legyen *F1*az *M’* által le nem fedett *F*-beli pontok halmaza, *L2* az L azon része, melyek az *F1*-ből alternáló úton elérhetők és pirosak (teljesül rá a ). *F2* az eddigi párosítás szerint L*2* párja.

Ekkor , majd az új címkézés:

Folytatjuk 2-től (*M*’ és *c*’-vel).



### Bizonyítás

Áll: c’ címkézés

Biz.: változása:

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  | 0 (-+) |  |
|  |  | 0 |

Csak azt az esetet kell megvizsgálni, amely esetén csökken. definíciója miatt a legkisebbet választottuk ki, így mindegyik él elveszthet -t, még akkor is címkézés marad.

Áll:

Ha az átcímkézés során keletkező piros él végpontja

* L1-beli, nő az M párosítás mérete, mivel ha a másik fele F1-ben van, akkor simán összeköthetjük, ha pedig F2-ben, akkor F1-L2-F2-L1 úton elérhető és ez hosszabb, mint az eredeti.
* L3-beli, akkor F3 és L2 között megszűnik egy piros él, de ez nem befolyásolja a párosítást. Az új piros és L3-beli végpontja elérhető lesz alternáló úton, ezért átkerül L2-be.

L3 legfeljebb n lépés után elfogy, utána biztosan L1-beli lesz a keletkező piros él végpontja és akkor nő M mérete. Legfeljebb n2 iteráció után az algoritmus megáll a teljes párosítással.

2. tétel

A lineáris programozás alapfeladata, kétváltozós feladat grafikus megoldása. Lineáris egyenlőtlenségrendszer megoldása Fourier-Motzkin eliminációval.

## A lineáris programozás alapfeladata

A lineáris programozás alapfeladata: egy egyenlőtlenségrendszer megoldásai közül kiválasztani azt, amely egy szintén lineáris célfüggvény szerint optimális.

*A* *m*×*n*-es mátrix, *x* egy *n* dimenziós, *b* egy *m* dimenziós, *c* egy *n* dimenziós vektor. A LP alapfeladata ekkor:

Megjegyzések:

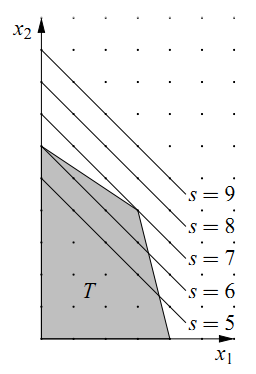
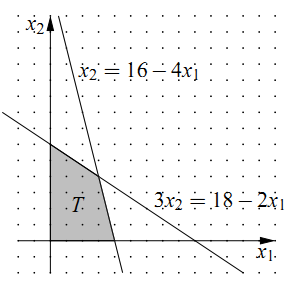
* helyett
* helyett és
* helyett (aztán ellentett-vétel a végleges megoldáshoz)

Kérdések:

1. Van-e Ax≤b-nek megoldása?
2. A megoldások halmazán cx korlátos-e?
3. Melyik x-re maximális cx?

## A kétváltozós feladat grafikus megoldása

* Minden zárt félsíkot határoz meg, határolója egyenes.
* Ha a félsíkok metszete véges, akkor konvex sokszöget kapunk, ezen belül vannak a lehetséges megoldások (bal ábra).
* A célfüggvényt különböző értékekre felrajzolva kiválaszthatjuk azt a célfüggvény – sokszög metszéspontot, amely optimális (jobb ábra; *s*=7, *x1*=3, *x2*=4 a megoldás).



## Fourier-Motzkin elimináció

Az *n* változós esetet mindig eggyel kevesebb változósra vezetjük vissza úgy, hogy a **megoldhatóság tulajdonsága megmaradjon**. Az utolsó lépésnél már csak 1 változó marad, amelyre a megoldhatóság egyszerűen vizsgálható.

Minden lépésben **pozitív számmal való szorzással** olyan alakra hozzuk a mátrixot, hogy 1, 0 vagy -1 legyen az első oszlopban.

Ha *A*– üres, akkor *A*+ sorai teljesülnek. *A*+ sorai ilyen alakúak:

Ekkor *x*1-et olyan kicsire választhatjuk, hogy az összes ilyen sort kielégítse. *A*+ **teljes sorai** így elhagyhatók (nem csak az első oszlopuk), elég A0 sorait tovább vizsgálni a 0 oszlop elhagyásával  
((*n* – 1) változós eset).

Hasonlóan, ha *A*+ üres, akkor *A*– sorai teljesülnek alapján.

Ha sem *A*+, sem *A*– nem üres, akkor *A*+ és *A*– összes {*i*, *j*} sorpárját összeadva az , egyenlőtlenségrendszer megoldható (az első oszlopra 0 adódik, így (*n* – 1) változós esetre áttérhetünk).

A végén egy változó marad. Ekkor ha *b*0-ban van negatív, akkor a rendszer nem megoldható, mert: sosem teljesülhet, egyébként *A*0 sorai teljesülnek (elhagyhatók). Ha *A*+ vagy *A*– üres, akkor a rendszer megoldható (lásd előző bekezdés). Egyébként megoldható, ha esetén:

és .

Azaz

Megjegyzés: A Fourier-Motzkin elimináció futási ideje exponenciális.

3. tétel

Farkas-lemma (két alakban). A lineáris program célfüggvénye felülről korlátosságának feltételei.

## Farkas lemma 1.

Pontosan az egyiknek van megoldása:

### Bizonyítás

Kettőnek egyszerre nincs megoldása, mert: .

Be kell látni még, hogy ha (1) nem megoldható, akkor (2) igen, teljes indukció. Egyváltozós esetben (1) nem megoldható (lásd Fourier-Motzkin egyváltozós eset), ha *b*0-ban van negatív, illetve van egy alakú sorpárja, hogy:

ami akkor nem megoldható, ha:

(azaz a negatív feltétel abszolútértéke nagyobb, mint a pozitívé – nem lesz érvényes intervallum *x*-re).

Ha *y* eze(ke)n a komponens(ek)ben 1, a többin 0, akkor megoldja (2)-t:

, illetve

vagyis létezik megoldás.

*n*-változós eset: ha *y* megoldás, akkor *α*y is (), ezért elérhető. A rendszer tömören felírva: . Más szóval sorainak *y* szerinti (nemnegatív együtthatós) lineáris kombinációja a vektort adja. Jelölje az eliminációs lépés után -t. Az indukciós feltevés miatt megoldható. Egészítsük ki a rendszert az eliminációs lépés végén elhagyott csupa 0 oszloppal: . Ettől továbbra is megoldható marad, kifejezhető vektor pozitív együtthatós lineáris kombinációja. Visszafelé gondolkodva az eliminációs lépésben egy sora vagy egy sora volt (A0 csoportból), vagy két sorának összege (A+ és A- csoportokból), vagyis sorainak nemnegatív együtthatós lineáris kombinációja kiadja vektort.

## Farkas lemma 2.

Pontosan az egyiknek van megoldása:

### Bizonyítás 1.

Kettőnek egyszerre nincs megoldása:

Ha (2) nem megoldható, akkor (1) igen.

(2) nem megoldható, így:

nem megoldható (Farkas lemma 1. n változós esetnél ismertetettek miatt -1 is megoldás lenne)

Ekkor y sorvektor, ezt írjuk át y’ oszlopvektorra:

Erre alkalmazzuk a Farkas lemma 1. alakját. Ez az (1)-el analóg. Mivel ez nem megoldható, így (2) igen:

Ekkor x’ sorvektor, így térjünk át x oszlopvektorra:

x-et írjuk fel a következő alakban:

Ekkor:

Legyen:

Így:

Ami a Farkas lemma 2. alakjának (1)-es állítása, ami a feltételekből következően megoldható. Tehát bizonyítottuk az állítást.

### Bizonyítás 2.

(1)-et átírjuk egyenlőtlenségrendszerre, amire alkalmazható a Farkas lemma 1. (E: egységmátrix)

(Farkas lemma 1. alak (1), Farkas lemma 2. alak (1)-ből)

Ennek megfelelő másik egyenlőtlenségek:

ahol

Legyen:

Így megkaptuk a Farkas lemma 2. alakjának (2)-nek megfelelő egyenlőtlenségeket, ezzel bizonyítva az állítást.

## Célfüggvény korlátosság

megoldható, ekkor az állítások ekvivalensek:

1. Az megoldáshalmazon *cx* felülről korlátos.
2. Nincs megoldása az rendszernek
3. Van megoldása az rendszernek.

### Bizonyítás

(1)🡪(2) indirekt. Legyen *x*0 megoldása -nek, és mégis létezik *z*, hogy: . Ekkor tetszőleges is megoldás, mert . Továbbá nem lehet korlátos felülről, mert és *λ* tetszőlegesen nagy lehet.

(2)🡪(3). Farkas lemma 2-ből adódik ((2)=(2), (3)=(1) helyettesítés), (2)-nél a *z* helyett (*–z*)-t vesszük

(3)🡪(1). , minden *x, y*-ra *yb* felső korlát *cx* értékére.

4. tétel

A lineáris programozás dualitástétele (két alakban). A lineáris programozás alapfeladatának bonyolultsága (biz. nélkül).

## Célfüggvény korlátosság

megoldható, ekkor az állítások ekvivalensek:

1. Az megoldáshalmazon *cx* felülről korlátos.
2. Nincs megoldása az rendszernek
3. Van megoldása az rendszernek.

### Bizonyítás

(1)🡪(2) indirekt. Legyen *x*0 megoldása -nek, és mégis létezik *z*, hogy: . Ekkor tetszőleges is megoldás, mert . Továbbá nem lehet korlátos felülről, mert és *λ* tetszőlegesen nagy lehet.

(2)🡪(3). Farkas lemma 2-ből adódik ((2)=(2), (3)=(1) helyettesítés), (2)-nél a *z* helyett (*–z*)-t vesszük

(3)🡪(1). , minden *x, y*-ra *yb* felső korlát *cx* értékére.

## Dualitás 1.

Ha primál program megoldható és felülről korlátos, akkor

1. duális program is megoldható és alulról korlátos
2. a primál programnak létezik maximuma, a duálisnak minimuma
3. maximum = minimum

### Bizonyítás

**(1) bizonyítása:**

A célfüggvény korlátosságánál ekkor (1) teljesül a dualtitás 1. alapfeltevése miatt. Mivel (1) teljesül, így (3) is. Tehát dualitás 1. (1) tényleg megoldható és miatt van alsó korlát.

**(2) és (3) bizonyítása:**

*Állítás:* Tegyük fel, hogy megoldható és *t* tetszőleges. Ha -nek nincs olyan megoldása, amelyre teljesülne, akkor -nak van megoldása, amelyre teljesül.

*Bizonyítás:* Átírva: rendszer nem megoldható, alkalmazzuk rá a Farkas lemma 1-et. Az abban szereplő (2) rendszer *y* vektor utolsó komponensét (ami a hozzáadott egyenlőtlenség) válasszuk külön, jelölje *λ*. Ennek megfelelően kifejtve a lemma (2) egyenleteit:

Ha lenne, akkor teljesülne, így a Farkas lemma alapján nem megoldható, ez ellentmond az állításnak. Ezért , így bevezethetjük . Erre , , teljesül, tehát *y'* teljesíti az állítást.

**(2) bizonyítása primálra:**

Indirekt módszerrel. Nem létezik maximum, de minden felülről korlátos halmaznak van szuprémuma (legkisebb felső korlátja), jelöljük: . Mivel *t* nem maximum, ezért -nek nincs -t teljesítő megoldása. Az előbbi állítás miatt rendszernek van -t teljesítő megoldása. Ekkor: . Itt *yb* *t*-nél kisebb felső korlát *cx*-re, ami ellentmondás, mert *t* a szuprémum.

**(2) bizonyítása duálisra:**

Egyszerűen átírjuk: alakra.

**(3) bizonyítása:**

fennáll miatt. Indirekt tegyük fel, hogy itt nem egyenlőség áll és legyen . Indirekció miatt -nek nincs megoldása, a fenti tétel szerint ekkor a duálisnak van megoldása, amire , ami nyilvánvalóan nem lehet, mert *t* minimális.

## Ekvivalens alak

Ha (primál) program megoldható és felülről korlátos, akkor a  
 (duális) program is megoldható és alulról korlátos; a primál programnak létezik maximuma, a duálisnak minimuma, és ezek megegyeznek.

## LP bonyolultsága

* -t kiegyenlítő *x* megoldások közül van-e, ami felülről korlátos *cx*-en?
  + NP-beli: *x* tanú
  + coNP-beli: duális megoldása tanú
* szimplex módszer (1947) nem polinomiális, de gyors
* ellipszoid módszer (1979, Hacsijan) polinomiális, de lassú
* belső pontos módszerek (1984, Karmakar) polinomiális, de lassú

5. tétel

Egészértékű programozás: a feladat bonyolultsága, korlátozás és szétválasztás (Branch and Bound)

## Egészértékű programozás

* IP alapfeladat:
* IP duálisa:

## A feladat bonyolultsága

* -t kiegyenlítő *x* egész megoldások közül van-e, ami felülről korlátos *cx*-en?
  + NP-beli: *x* tanú
  + nincs dualitástétel: nem adódik a co-NP-beli.
  + NP-teljes: visszavezetés a 3-SAT problémára.

## Tétel

Az (IP) probléma NP-teljes.

### Bizonyítás

Mivel a feladat NP-beliségét már láttuk, elég lesz a 3-SAT problémát visszavezetni az (IP) problémára. Megadunk tehát egy eljárást, amely tetszőleges

alakban megadott logikai függvényhez megad egy olyan (IP) problémát, melyre akkor és csak akkor igenlő a válasz, ha f kielégíthető. Az (IP) probléma ismeretlenei legyenek z1,z2,…,zk, az egyenlőtlenségek:

(1a) minden i=1,2,…,k-ra

(1b) minden i=1,2,…,k-ra

továbbá minden diszjunkcióra is felírunk egy egyenlőtlenséget:

* ha egy diszjunkció alakú (vagyis minden változó ponáltan (negálatlanul) szerepel benne), akkor  
   (2a)
* ha egy diszjunkció alakú, akkor  
   (2b)
* ha egy diszjunkció alakú, akkor  
   (2c)
* ha egy diszjunkció alakú, akkor  
   (2d)

Azonnal látszik, hogy ha f kielégíthető, akkor a választással az egyenlőtlenségrendszernek megtaláltuk egy egészértékű megoldását.

Megfordítva, ha az egyenlőtlenségrendszert egész értékekkel ki tudjuk elégíteni, akkor – mivel minden egész vagy 0, vagy 1 – az választással f-et kielégítő értékeket adtunk a logikai változóknak. Eközben a összeg értéke valahol 0 és k között adódik, tehát c=(1,1,…,1) választással tulajdonképpen a „Van-e az (1a-1b) és a (2a-2d) feltételeket kielégítő, egész vektorok között olyan, melyre ? ” kérdést tesszük fel. Az így kapott (IP) problémára a válasz akkor és csak akkor igenlő, ha f kielégíthető.

## A Branch and Bound algoritmus

Feladat: , *f* és *g* tetszőleges egész vektorok.

### Jelölések

* részfeladatok
* *w(i): IP(i)* maximumértéke ennél nagyobb nem lehet
* *z\**: eddigi legjobb célfüggvényérték
* *x\** eddigi legjobb megoldás ()

### Az algoritmus lépései

1. lépés: eredeti feladat, , nem definiált.
2. lépés: Ha üres, akkor STOP (megoldás *z\**, *x\** helyen). Egyébként vegyünk egy feladatot -ből (és töröljük onnan).
3. lépés: Ha : IP*(i)* nem lehet megoldás, GOTO 1. (Bound lépés.)
4. lépés: IP*(i)* helyett LP*(i)* relaxált feladat megoldása. Ha nincs megoldás, akkor GOTO 1. Egyébként maximum *z(i)*, maximumhely *x(i)*.
5. lépés
   1. Ha , akkor GOTO 1
   2. Ha , *x(i)* egész vektor. Ekkor z\* és x\* fölülírjuk ezeket az értékekkel. GOTO 2
   3. Ha , *x(i)* nem egész vektor. Ekkor válasszunk *x(i)*-ben egy elágazási változót: xj, és egy közbülső értéket: . Két új feladatot hozunk létre, módosítjuk a korlátokat a megadott komponensre, egyikben: , másikban . A két új célfüggvény felső korlát: . Adjuk hozzá -hez a két új feladatot. GOTO 1. (Branch lépés.)

### Az algoritmus hatékonysága

*Állítás:* Az algoritmus véges sok lépésben leáll és megtalálja a feladat optimumát.

*Bizonyítás:* Véges: *f* és *g* miatt, mert csak véges sok egész *x* lehet úgy, hogy .

Optimális: indirekt bizonyítás. Legyen *z0* az optimum, de az eljárás csak -t találta. -ben mindig van feladat, aminek az optimuma *z0*. Ez a legelején (0. lépésnél) fönnáll. Ilyenkor 4b vagy 4c lépéshez jut az algoritmus, mert még nincs optimális megoldás. 4b-nél meg is találja 🡪 ellentmondás. 4c esetén az egyik alfeladatban benne van az optimumhely és így optimumérték (*z0*) is. Így viszont sosem ürülne ki, és az algoritmus sosem állna le, ellentmond az előző bekezdéssel.

### Gyakorlati tapasztalatok az algoritmus használatára

* LIFO alapján válasszunk új feladatot -ből, mert a megoldás várhatóan mélyen van a fában, és LIFO-val tudunk a legmélyebbre hatolni.
* elágazásnál a legkevésbé egész *xj*-t válasszuk elágazási változónak (azaz amelyiknek a tört része 0,5-höz legközelebbi), közbülső értéknek pedig ennek egészrészét.

6. tétel

Totálisan unimoduláris mátrix fogalma, példák. Egészértékű programozás totálisan unimoduláris együtthatómátrixszal. (biz. nélkül) Alkalmazás páros gráfokra és intervallumgráfokra.

## Totális unimoduláris (TU) mátrix

Minden négyzetes részmátrixának determinánsa 0, 1 vagy –1. Szükséges (de nem elégséges), hogy a mátrix elemei is csak 0, 1 vagy –1 értékűek lehetnek.

## Egészértékű totálisan unimoduláris együtthatómátrixszal

Legyen *A* totálisan unimoduláris mátrix, *b* egész koordinátájú vektor. A (LP) feladat megoldható és maximuma véges. Ekkor a (IP) feladat is megoldható, és maximuma megegyezik (LP) feladat maximumával. A maximumhelyek ekkor egészek.

, akkor ha c is egész vektor

## Lemma

Egy mátrix totális unimoduláris marad, ha

1. egy sorát/oszlopát (–1)-gyel szorozzuk
2. egységvektort hozzáveszünk sorként/oszlopként
3. egyik sorát/oszlopát új sorként/oszlopként hozzávesszük
4. transzponáljuk

### Bizonyítás

Nyilván csak azoknak a négyzetes részmátrixoknak változhat a determinánsa, amelyikeket érint a változás.

1. A determináns (–1)-szeres lesz.
2. Ha a mátrixhoz egységvektort veszünk hozzá például oszlopként, és egy kiválasztott négyzetes részmátrixában ez az oszlop szerepel, akkor az új oszlop szerinti kifejtésből azonnal látszik, hogy a determináns megegyezik az eredeti mátrix egy négyzetes részmátrixának determinánsával vagy annak ellentettjével, így értéke 0, 1 vagy -1.
3. Ha a régi és új sor/oszlop is szerepel a kiválasztott részmátrixnak, akkor a determináns 0. Ha csak az egyik (vagy a régi vagy az új), akkor előáll az eredeti mátrixból képzett részmátrix, determináns marad 0, 1 vagy –1.
4. Determináns definíciójának következménye.

## Illeszkedési mátrix

Legyen az n-pontú G gráfnak e éle és definiáljuk az n\*e méretű B(G)=(bij) mátrix elemeit úgy, hogy:

Legyen akkor is, ha a j-edik él az i-edik ponthoz illeszkedő hurokél. Irányítatlan esetben is ez a definíció, csak ott a j-edik él mindkét végpontjának megfelelő mátrixelem 1. Ezt a B(G) mátrixot a G gráf illeszkedési mátrixának nevezzük.

## Tétel

Minden irányított gráf illeszkedési mátrixa totálisan unimoduláris.

### Bizonyítás

Mivel irányított illeszkedési gráfról beszélünk, egy oszlop azt adja meg, hogy egy adott él mely pontokra illeszkedik (1 ott, ahonnan indul, és -1 ott, ahova mutat; ha hurokél, akkor az egyetlen pontjánál 1 van).

Teljes indukció: *M* egy méretű részmátrix.

* Ha , akkor nyilvánvaló, mert minden elem 0, -1 vagy 1.
* Ha , és
  + *M*-nek van olyan oszlopa, amelyben legfeljebb egy nemnulla elem van, akkor fejtsük ki detM-et e szerint az oszlop szerint, ekkor az indukciós feltétel szerint készen vagyunk.
  + Egyébként minden oszlopban egy +1 és egy –1 elem van, ekkor *M* sorainak összege nullvektor, a determináns 0.

## Tétel

Páros gráf illeszkedési mátrixa totálisan unimoduláris.

### Bizonyítás

Felhasználjuk az irányított gráf tételét: páros gráf éleit irányítsuk úgy, hogy minden él *A*-ból *B*-be mutasson, ekkor B(G) TU. A *B*-hez tartozó sorokat negáljuk, de ez nem változtat TU tulajdonságon.

## Maximális összsúlyú párosítás IP feladatként

Jelölések: *x* indikátor, hogy egy él benne van-e a párosításban. *w* tetszőleges (él)súlyfüggvény, *B* illeszkedési mátrix (páros gráf lévén TU).

### Feladat

. Az feltétel miatt *B*-t kiegészítjük az -es egységmátrix ellentettjével, de ez nem változtat TU tulajdonságon. A w adja meg az élsúlyokat. A azt jelenti, hogy az egy csúcsból kiinduló élek közül legfeljebb egyet választunk ki 🡪 párosítás. Mivel *B* TU mátrix és *b* egész vektor, ezért *x* egész vektor (IP feladat) (pontosabban 0 vagy 1 értékű).

### Duális

. Ekkor *y* megoldás minden *v* csúcshoz egy *c(v)* címkét rendel. A azt jelenti, hogy minden élre (Egerváry Jenő).

## Intervallumgráf

A számegyenes véges sok intervalluma alkossa egy gráf csúcshalmazát, és két csúcs akkor legyen szomszédos, ha a megfelelő intervallumok metszők; az így előálló gráfokat intervallumgráfoknak nevezik. Könnyű végiggondolni, hogy a fenti meghatározásban feltehető, hogy a gráfot meghatározó intervallumok valamely n-re az [1,n] egész végpontú, zárt részintervallumai; ezt az alábbiakban végig fel is tesszük.

Legyen tehát adott az I = {I1, I2, ... , Im} intervallumrendszer. Ehhez a következő n x m-es A(I) mátrixot rendelhetjük: A(I) sorai feleljenek meg az 1,2,…,n egészeknek, oszlopai pedig az I intervallumainak; az i-edik sor és a j-edik oszlop kereszteződésében akkor álljon 1-es, ha , és minden más helyen álljon 0.

## Tétel

Az így definiált A(I) mátrix totálisan unimoduláris.

### Bizonyítás

Kiválasztunk egy tetszőleges *k*×*k* részmátrixot, teljes indukció 1-esek darabszáma szerint.

0 db egyesből álló nyilván TU, magasabb darabszámú:

* Ha van benne felülről nézve két olyan oszlop, melyben az első 1-es azonos helyen áll, akkor a nagyobb 1-es darabszámúból kivonjuk a kisebb darabszámút. Ez az 1-esek darabszámát csökkenti, determinánst nem változtatja.
* Ha nincs ilyen oszlop, de van csupa 0 oszlop, akkor a determináns 0.
* Ha egyik sem teljesül, akkor pedig be tudjuk rendezni az oszlopokat úgy, hogy alsó háromszögmátrixot kapunk, itt a determináns +1 vagy -1.

## Tétel

(intervallumgráfok tetszőleges *k* színre megszínezhetők egyenletesen)

Az [1,n] egész végpontú, zárt I1, I2, … , Im részintervallumai minden k pozitív egészre megszínezhetők k színnel úgy, hogy a színezés minden esetén „egyenletes” legyen. Itt az egyenletes színezés azt jelenti, hogy ha az i-t tartalmazó intervallumok száma di, akkor ezek közül minden felhasznált szín esetén az ilyen színű intervallumok száma vagy . ( alsó egészrészt, felső egészrészt jelöl.)

### Bizonyítás

Bármely *i* pozícióban lévő intervallumok közül kiválaszthatunk egyenletes darabszámút (ez a tétel bizonyításához már elég). Felírjuk először LP feladatként, majd konstatáljuk, hogy mivel a mátrix TU, a *b* pedig egészértékű (csupa 1-es), ezért lesz egészértékű megoldás is.

7. tétel

A lineáris és egészértékű programozás alkalmazása hálózati folyamproblémákra.

## Jelölések

irányított gráf

* két kitüntetett csúcs
* nemnegatív *kapacitásfüggvény*. tetszőleges függvény.
* : *v*-be belépő élek összege *x* szerint
* : *v*-ből kilépő élek összege *x* szerint
* Egy *x* függvény akkor *folyam*, ha minden esetén
* *x* függvény megengedett, ha minden *e* élre:
* A folyam értéke .

## Ford-Fulkerson tétel

A maximális folyam értéke megegyezik a minimális vágás értékével.

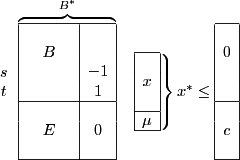
## Lemma

Legyen olyan függvény, amely minden esetén , továbbá . Ekkor *x* folyam.

### Bizonyítás

Minden pontból vegyünk fel egy új élt, az új gráf legyen *G’*. Minden új *e* élhez rendeljük hozzá az értéket (amennyivel több a belépő, mint a kilépő), a régi éleken maradjon az érték . A kapott *x’* függvény *G’*-ben nyilván folyam, ezért . Másrészt és . A lemma feltételei alapján: . Mivel csak egyenlőség állhat, ezért , ami csak akkor teljesülhet, ha *x* folyam.

## LP felírás

Egészítsük ki G gráfot egy pszeudó éllel, (értéke lesz majd a folyam értéke, jelöljük *µ*-vel), az így kapott gráf illeszkedési mátrix legyen *B\**. Minden csúcshoz tartozik egy sor, melyre teljesül (ez folyamnál azt jelenti, hogy a belépő élek összege nem kisebb, mint a kilépőké; ). rendszer alapján , illetve , összevontan: . Az előző lemma miatt , vagyis a folyam értéke *µ*.

Keressük a maximális folyamot, vagyis . Az feltételt hozzávesszük a mátrixhoz (ez lesz az *E* egységvektor és a hozzá tartozó *c* rész a *b* vektorban) az alábbi formában, ezt jelöljük *M*-mel.

## Duális

Írjuk fel másképp *y*-t:

Ekkor:

1. ,
2. minden él esetén

A duális változói közül a csúcsokhoz, az élekhez rendel értéket.

Ekkor a cél minimum.

## Állítás

A értéke megegyezik a hálózati folyam minimális vágásának értékével (*mC*).

### Bizonyítás

Egy *mC* vágáshoz könnyen készíthető olyan *π* és *w*, amelyre , a következők szerint. Legyen *S* () és *T* () diszjunkt halmazok, akkor esetén , esetén legyen . Minden *e* él, ami *S*-ből *T*-be mutat: , egyébként . A teljesül, ebből adódik, a másik irányú egyenlőtlenséget kell belátni.

Az M mátrix TU, ezért (Egészértékű totálisan unimoduláris együtthatómátrixszal) alapján *y* is egészértékű (mivel rendszerbeli *c* is egészértékű). Legyen adott (*π, w*) optimális, egészértékű megoldás, ebből kiindulva elkészítünk egy (*π’, w’*) 0 vagy 1 értékű optimális megoldást. Legyen

Ekkor (*π’, w’*) (1) és (3) nyilván teljesül

(2) indirekt: él esetén , ekkor a 0 – 1 értékűség miatt , esetben valósulna meg, ekkor definíciója miatt  
, ami ellentmondana (2)-nek, mert *w’* definíciója miatt .

Optimális, mert , mert .

Legyen és a fentiek szerinti *S* () és *T* (). Ha egy élre és , akkor (2) miatt . Minden más élen *w’(e)* csak akkor lehet 1, ha , mert -ra változtatása után a feltételek továbbra is fennállnának, és csökkenne. Ezért az S és T között feszülő élek alkotta vágás értéke , így  
.

## Alkalmazás minimális költségű folyam keresésére

Minden élhez rendelhető egy *k* költség is. Ekkor kitűzhető egy olyan feladat, ami a legalább *M* nagyságú folyamok között keres minimális költségűt.

Megfogalmazása lineáris programozási feladatként:

## Alkalmazás többtermékes folyamra

Legyen adott egy G=(V,E) irányított gráf és abban k darab pontpár: (s1,t1),(s2,t2),…,(sk,tk). G-t továbbra is képzelhetjük út- vagy csőhálózatnak, az (si,ti) pontpárok pedig a k darab szállítandó termék termelő-, illetve fogyasztóhelyének felelnek meg. Végül legyen adott a kapacitásfüggvény. A feladat egy megoldása abból áll, hogy minden élhez k darab számot fogunk hozzárendelni.

Ezekre is alkalmazható a LP, a legalább 2 termékes folyamoknál már az egyetlen ismert hatékony algoritmus. IP ezekre már NP nehéz.

8. tétel

Matroid definíciója, alapfogalmak (bázis, rang, kör). Példák: lineáris matroid (mátrixmatroid), grafikus matroid, uniform matroid. A rangfüggvény szubmodularitása.

## Matroid definíció (mohó algoritmus)

Egy *E* alaphalmazon értelmezett nemüres, leszálló halmazrendszer matroid, ha tetszőleges nemnegatív súlyfüggvényre a mohó algoritmus optimális – maximális súlyú – megoldást ad.

**Mohó algoritmus:** olyan algoritmus, amely minden lépésében olyan elemet választ, melyre az alábbi állítás igaz:

## Matroid definíció (függetlenségi axiómák)

Legyen *E* tetszőleges véges halmaz, matroid, ha:

1. Ha és , akkor (leszálló halmaz)
2. Ha és , akkor létezik olyan , amelyre

**Tétel:** A fenti 2 definíció ekvivalens.

## Alapfogalmak

Az matroidban az alaphalmaz -hez tartozó részhalmazait *független* halmazoknak nevezzük. A maximális (nem bővíthető) független halmazok a matroid *bázis*ai. Egy minimálisan összefüggő (egy elem elvételével már független) halmazt *kör*nek nevezünk. Az egyelemű kör *hurok*. Az *M* matroidban egy halmaz *rang*ja *r(X)*: egy *X*-beli maximális független halmaz mérete. A matroid ranga *r(E)*.

**Def.**: (E,F) matroidban ha X⊆E és X∈F, akkor X független.   
**Def.**: (E,F) matroidban ha X⊆E és X∉F, akkor X összefüggő.   
**Def.**: ha X maximális elemszámú független halmaz, akkor bázisnak hívjuk.   
**Def.**: ha X minimális elemszámú összefüggő halmaz, akkor körnek hívjuk.   
**Def.**: X rangja r(X) = az X által tartalmazott maximális független részhalmazok (közös) elemszáma.

## Lemma

Legyen matroid, . Ha *X1* és *X2* maximális független halmazok *A*-ban, akkor .

### Bizonyítás

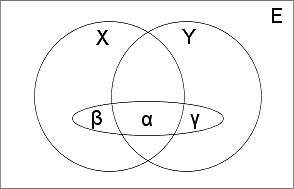
Indirekt: . (F3) miatt létezik olyan , amelyre , vagyis *X1* nem lehetett maximális.

## Példák

* *Grafikus matroid* (körmatroid): *G* gráf által indukált matroid. , {*G*-beli erdők}
* *Lineáris matroid*: *A* mátrix által indukált matroid. , {*A* lineárisan független oszlopai}.
* *Uniform matroid*: *E*: tetszőleges véges halmaz, jelölje *n* az elemszámát. {*E* legfeljebb *k*-elemű részhalmazai} (), jelölése: *Un,k*. Speciálisan: *Un,n*: teljes/szabad matroid, *Un,0*: trivális matroid.

Egy uniform matroid grafikus, ha *Un,0 , Un,1 , Un,n-1 , Un,n alakú.*

## Rangfüggvény szubmodularitás

Legyen *r* egy matroid rangfüggvénye. Ekkor minden halmazpárra:

1. .

Fordítva: ha *r* egy egészértékű függvény *E* részhalmazain, amelyre teljesül (R1) – (R4), akkor *r* egy matroid rangfüggvénye, ahol .

### Bizonyítás

(R1) – (R3) a rangfüggvény definíciójából adódik.

(R4): adott *X, Y* halmazpárra legyen *A* maximálisan független, , elemszáma . Az *A* halmaz kiterjeszthető egy olyan *B* halmazzá, amely független lesz -ban. *X*-ből *β*, *Y*-ból *γ* új elem kerül *B*-be. A rangfüggvények a következőképp alakulnak:

Összesítve:

9. tétel

Mohó algoritmus matroidon. Matroid megadása rangfüggvényével, bázisaival (biz. nélkül). Matroid duálisa, a duális matroid rangfüggvénye.

## Mohó algoritmus

Legyen matroid, nemnegatív súlyfüggvény. Keressük a maximális összsúlyú független halmazt, azaz: . Tetszőleges matroidra és súlyfüggvényre optimális megoldást (maximális összsúlyú) ad a mohó algoritmus.

### Bizonyítás

Indirekt. A mohó algoritmus megoldást adta, de az optimális . Mivel mindkét halmaz maximálisan független, ezért (F3) miatt . Legyen mindkét halmaz költség szerint csökkenő sorrendbe rendezve, vagyis és . Tudjuk, hogy a mohó algoritmus a legnagyobb súlyú elemmel kezd, ezért .

Legyen *i* a legkisebb index, amire Ilyen mindenképp létezik, különben a mohó algoritmus optimális megoldást adott volna. Legyen és . Alkalmazzuk (F3)-at erre:

Létezik , amelyre . Feltettük, hogy csökkenő sorrendbe vannak rendezve a halmazok elemei, vagyis , továbbá miatt az algoritmus *bj*-t választotta volna, nem *ai*-t.

## Matroid definíció (függetlenségi axiómák)

Legyen *E* tetszőleges véges halmaz, matroid, ha:

1. Ha és , akkor (leszálló halmaz)
2. Ha és , akkor létezik olyan , amelyre

## F3 vs mohó

Legyen matroid, nemnegatív súlyfüggvény. (F1) – (F2) teljesül, továbbá tetszőleges *w* költségfüggvényre a mohó algoritmus maximális súlyú megoldást ad: . Ekkor (F3) is teljesül.

### Bizonyítás

Indirekt: mohó algoritmus OK, de (F3) nem teljesül. Ilyenkor , amelyeknél fennáll, de nem létezik olyan , amelyre teljesülne. Legyen és súlyfüggvény a következő:

Az algoritmus először kiválasztja *Y* elemeit. Ezt követően az indirekt feltevés miatt már nem választhat *X – Y*-beli elemeket, csak 0 súlyúakat. Ekkor az összsúly lesz. Azonban *X* összsúlya ennél nagyobb, tehát a mohó algoritmus nem találta meg az optimális megoldást 🡪 ellentmond a kezdeti feltevésnek.

## Bázisos megadás

Legyen egy matroid bázisainak a halmaza. Ekkor

1. minden -re
2. Ha és , akkor létezik olyan , melyre .

Fordítva: ha egy halmazrendszer, amelyre teljesül (B1) – (B3), akkor matroid, ahol .

## Rangfüggvény megadás

Legyen *r* egy matroid rangfüggvénye. Ekkor minden halmazpárra:

1. .
2. , ha

Fordítva: ha *r* egy egészértékű függvény *E* részhalmazain, amelyre teljesül (R1) – (R4), akkor *r* egy matroid rangfüggvénye, ahol .

## Duális matroid definíció

matroid bázisai , akkor a duális matroid bázisai . Ebből már adódik

## Duális matroid tétel

Az matroid.

### Bizonyítás

(F1) – (F2) nyilvánvalóan teljesül, (F3)-at kell belátni: ha és , akkor létezik olyan , amelyre . Legyen és bázis az eredeti matroidban. Ha van olyan elem *X*-ben, ami nincs benne *Y*-ban, sem *BY*-ban, akkor ezt *Y*-hoz hozzávéve ismét -beli elemet kapnánk.

Egyébként nem ilyen egyszerű a helyzet, azaz . Ekkor a feltevés miatt. Mivel *BX* és *BY* ugyanakkora, ebből következik. Alkalmazzuk erre a két halmazra (és *M* matroidra) (F3)-at: létezik olyan elem, amelyre független halmaz *M*-ben. Ezt a független halmazt egészítsük ki bázissá úgy, hogy *BY*-ból veszünk hozzá új elemeket, jelöljük *B’*-vel.

*B’* tartalmaz elemet -ból, tehát létezik olyan elem *BY*-ban – jelöljük *u*-val –, ami nincs B’-ben. Ekkor elemre teljesül: miatt .

## Duális rangfüggvény

.

### Bizonyítás

10. tétel

Elhagyás és összehúzás. Matroidok direkt összege, összefüggősége. T test felett reprezentálható matroid duálisának T feletti reprezentálhatósága.

## Elhagyás

matroid, elhagyása: matroid, ahol  
. Pl.: grafikus matroid esetén élek elhagyása.

## Összehúzás

matroid, összehúzása: matroid, ahol *M/X* rangfüggvénye: .

## Tétel

Az elhagyások és összehúzások fölcserélhetők. Minden *M* matroid *N* minora (elhagyások és összehúzások sorozata) előáll alakban, ahol *A* és *B* diszjunkt halmazok.

## Tétel

Elhagyás és összehúzás duális művelet: és .

### Bizonyítás

Elég az elsőt bizonyítani. A rangfüggvénye:

rangfüggvénye:

ahol az *M\X* matroid alaphalmaza. Ebből adódik.

## Matroidok direkt összege

Legyen és két matroid a diszjunkt *E*1 és *E*2 nemüres alaphalmazon. A két matroid direkt összege az az matroid, melynek alaphalmaza , és egy halmaz akkor független *N*-ben, ha és független *M1*-ben, illetve *M2*-ben.

## Matroidok összefüggősége

Egy matroid összefüggő, ha nem áll elő matroidok direkt összegeként. Egy grafikus matroid akkor összefüggő, ha a gráf kétszeresen összefüggő.

## T test felett reprezentálható

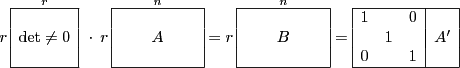
Az matroid reprezentálható *T* test felett, ha *E* minden eleme *T* feletti vektor.

### Definíció2

Az matroid reprezentálható (koordinátázható) *T* test felett, ha létezik olyan mátrix, amelynek oszlopai *T* feletti vektorok, és ezek által meghatározott lineáris matroid izomorf *M*-mel. (E minden eleme T feletti vektor)

Legyen r=r(E) és n=|E|. M(E,F) leírható egy r×n-s A mátrixszal, aminek sorai lineárisan függetlenek.   
r sor mindenképpen szükséges, ha pedig több sorból áll a mátrix, kiválaszthatunk r lineárisan függetlent, és elhagyhatjuk a maradékot, a matroid ugyanaz marad.

A kapott mátrix pedig egy alkalmas nemszinguláris r×r-es mátrixszal való szorzással leképezhető úgy, hogy a baloldalán egységmátrix legyen. A transzformált mátrix ugyanazt a matroidot koordinátázza.



## Duális reprezentációja T test felett

Ha matroid reprezentálható *T* test felett, akkor *M\** is.

### Bizonyítás

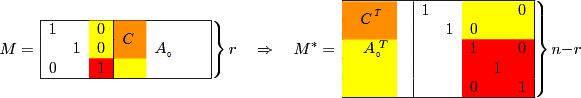
Elég belátni, ha baloldalt egy r×r-es részmátrix nemszinguláris, akkor jobboldalt a megfelelő   
(n-r)×(n-r)-es részmátrix is nemszinguláris.

Az ábrán a két narancssárga részmátrix megegyezik, a két piros részmátrix oszlopai pedig valamilyen sorrendben mindkét oldalon egységmátrixot alkotnak ⇒ ha a bázisnak megfelelő baloldali részmátrix nemszinguláris, akkor a jobboldali is. A bázisok 1-1 megfeleltethetők egymásnak ⇒ M és M\* egymás duálisai.

Induljunk ki a reprezentálás definíciójából: az alaphalmaz elemszámát jelöljük , *M* rangját *r*-rel, *M*-mel izomorf lineáris matroid pedig legyen *A*. Ekkor írjuk föl alakban, *Er* az -es egységmátrix (*M* egy bázisa), *A*0 a többi eleme. Belátjuk, hogy reprezentálja *M\**-ot.

Mindkét mátrixnak *n* oszlopa van, így az alaphalmazok megfeleltethetők egymásnak, a rangok is rendben vannak. Válasszuk ki *M* egy *B*1 bázisát (nyilván *r* oszlop), ezt válasszuk úgy, hogy *Er* utolsó *t* darab oszlopa, és *A*0 első *r – t* oszlopa (ábrán a színes). Világos, hogy az egységmátrixból vett oszlopokhoz tartozó oszlopok és sorok (sárgával) determinánsa nem lehet 0, egyedül *C* részmátrix kérdéses. Mivel *B*1 bázis, ezért az oszlopai függetlenek *C* determinánsa sem 0.

A duális matroidban azok az oszlopok alkotnak bázist, amik eredetileg nem, mert . Vagyis az első *r – t* és az utolsó *n – 2r + t* oszlop. Ez esetben sem kérdéses az egységmátrixhoz tartozó sorok és oszlopok függetlenek (determinánsuk ≠ 0), a maradék (narancssárga) rész a kérdéses. Azonban a konstrukció miatt ez pont *C* transzponáltja, tehát determinánsa nem lehet 0, ezért valóban bázis.



11. tétel

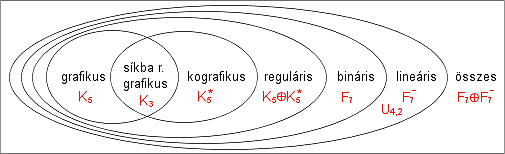
Grafikus, kografikus, reguláris, bináris és lineáris matroid fogalma, ezek kapcsolata (ebből bizonyítás csak a grafikus és reguláris matroidok közötti kapcsolatra), példák. Fano-matroid, példa nemlineáris matroidra. Bináris, reguláris és grafikus matroidok jellemzése tiltott minorokkal: Tutte tételei (biz. nélkül). Seymour tétele (vázlatosan, biz. nélkül)

## Matroid osztályok

*Grafikus matroid* (körmatroid): *G* gráf által indukált matroid. , {*G*-beli erdők}. Grafikus matroid duálisa *kografikus*.

*Reguláris* matroid, ha minden test felett reprezentálható. A *bináris* matroid a bináris test felett reprezentálható. Ha van olyan test, ami felett reprezentálható, akkor *lineáris* matroid.

**Def.**: M(E,F) grafikus matroid, ha ∃G gráf, hogy E=E(G) és F a G-beli erdők halmaza.   
**Def.**: a grafikus matroidok duálisát kografikus matroidnak hívjuk.   
**Def.**: M reguláris matroid, ha minden test felett koordinátázható.   
**Def.**: M bináris matroid, ha koordinátázható a bináris test felett.   
**Def.**: M lineáris matroid, ha van olyan test, ami felett koordinátázható.



K3,3\*

## Tétel

Grafikus matroid bármilyen test felett reprezentálható (reguláris).

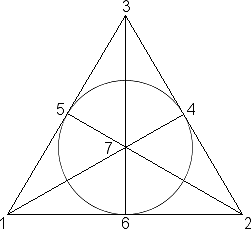
### Bizonyítás

Rendeljünk a gráf minden pontjához egy-egy különböző *n*-dimenziós egységvektort (n-pontú gráf esetén). Az élekhez pedig a két végpontja közti különbséget (irányítás lényegtelen). Az élek vektorában csak 0, 1 és –1 szerepelhet, így minden test felett reprezentálható.

Ha egy kör éleit ±1 együtthatós lineáris kombinációval összeadjuk, akkor nullvektort kapunk (–1-re azért van szükség, mert nem törődtünk az élek irányításával), mert a megfelelő nemnulla komponensek kiejtik egymást.

Fordítva: vegyünk egy összefüggő vektorhalmazt, ahol egyik együtthatója sem 0, jelölje *X*. Az élvektorok azokon a koordinátákon nemnulla, amelyik pontokat összeköt. Ahhoz, hogy a nullvektor kijöjjön, minden egy ponthoz két nemnulla koordináta kell, azaz egy pontra két él illeszkedik (a foka legalább 2). Ha valamely részgráf, ahol minden pont foka 2 (vagy több), akkor biztos van benne kör.

## Fano matroid

Adott a hételemű halmaz: . A legfeljebb kételemű részhalmazok függetlenek, továbbá a háromeleműek közül azok, amik nincsenek egy egyenesen vagy körön, ez a Fano-matroid (*F*7). A legfeljebb kételemű részhalmazok függetlenek, továbbá azok a háromeleműek, amik nincsenek egy egyenesen, ez az anti-Fano matroid ().

### Következmény

A Fano-matroid pontosan azon testek felett reprezentálható, amelynek karakterisztikája 2 (pl bináris matroid). Az anti-Fano-matroid pontosan azon testek felett reprezentálható, amelyek karakterisztikája nem 2. Ezért a két matroid direkt összege nem reprezentálható semmilyen test felett 🡪 nem lineáris.

## Tutte tételei

* M matroid *bináris* ⇔ nem tartalmaz minorként *U*4,2 matroidot.
* M matroid *regurális* ⇔ nem tartalmazza minorként *U*4,2, *F*7 és *F*7\* matroidokat.
* M matroid *grafikus* ⇔ nem tartalmazza minorként *U*4,2, *F*7, *F*7\*, *M*\*(*K*5) és *M*\*(*K*3,3) matroidokat.

## Seymour tétel

M reguláris ⇔ előáll 1 grafikus, 1 kografikus és egy R10 matroid néhány példányából a direkt összeg, 2-összeg és 3-összeg műveletek segítségével.

Az R10 matroid:

Ez egy 5 rangú matroid egy 10 elemű halmazon. Nem grafikus és nem kografikus.

12. tétel

Matroidok összege. k-matroid-metszet probléma, ennek bonyolultsága k ≥ 3 esetén.

## Matroid összege def

és matroidok összege , ahol , hogy és , illetve . (*X* előáll egy -beli és egy -beli elem uniójaként)

## Tétel

Matroidok összege is matroid.

### Bizonyítás

(F1), (F2) nyilvánvaló, (F3) indirekt: és , akkor nem létezik olyan , amelyre

Definíció szerint *X*, *Y* halmazoknak létezik (esetleg többféle) és felbontása, ahol , illetve . Válasszunk olyat, hogy a részhalmazok diszjunktak legyenek és érték legyen minimális, továbbá legyen . Ekkor *M*1 matroidra alkalmazva (F3)-mat, létezik , amelyre . Ha , akkor teljesülne, ami ellentmond az indirekciós feltevésnek. Tehát . Ekkor egy másik felbontás, ráadásul összeg eggyel kevesebb, mint az mint érték, ami ellentmondás (minimálisra választottuk).

## 2-matroid metszet probléma

k-matroid metszet probléma (MMPk): adott *k* darab matroid közös alaphalmazon: . Létezik-e valamely konstans *p*-re *p*-méretű halmaz -ben?

Az eredmény nem minden esetben matroid.

**Tétel**: 2-matroid metszet probléma: két matroid esetén,   
**Biz**: M1=(E,F1), M2=(E,F2), p egész. Ha p>min(r1,r2), nem a válasz. Csonkoljuk a matroidokat addig, amíg a rangjuk min(r1,r2,p)-re nem csökken. Ezzel a problémát redukáltuk közös bázis keresésére. r1=r2=p-re a válasz akkor és csak akkor igenlő, ha M1∨M2\*=(E,2E).

* ⇒: ha M1-nek és M2-nek van közös B bázisa, akkor M2\*-ban E-B bázis, E=B∪(E-B) egy felbontása az összegnek, azaz az összegben E független, tehát M1∨M2\* szabadmatroid.
* ⇐: legyen E=C∪D egy olyan felbontása a szabadmatroidnak, ahol C∈F1 és D∈F2\*.   
  |E| ≤ |C|+|D| = r1(C)+r2\*(D) ≤ r1(E)+r2\*(E) = r1(E)+|E|-r2(E) = |E|.   
  C és D diszjunkt bázisok ⇒ C közös bázisa M1-nek és M2-nek.

**Def.:** Egy M=(E,F) matroidot szabadmatroidnak nevezzük, ha E bármely részhalmaza független.

## MMPk bonyolultsága

MMPk NP-teljes (k≥3) esetén

### Bizonyítás

NP-beli, mert *p*-elemű közös bázis tanú

MMP3 visszavezethető irányított Hamilton-út keresésére, ami NP-teljes.

* M1=(E,F1)-ben X⊆E-re X∈F1 ⇔ az X részgráfban minden pont be-foka legfeljebb 1 és u be-foka 0.
* M2=(E,F2)-ben X⊆E-re X∈F2 ⇔ az X részgráfban minden pont ki-foka legfeljebb 1 és v ki-foka 0.
* M3 a gráf körmatroidja.

M1, M2 és M3 közös |V|–1 elemű bázisai G irányított Hamilton-útjai.

**Tétel**: MMPk k>3-ra NP-teljes.   
**Biz.**: az előző bizonyításban definiálit M1, M2 és M3 matroidokhoz vegyünk hozzá k-3 db szabadmatroidot.

13. tétel

A k-matroid partíciós probléma, ennek algoritmikus megoldása. A 2-matroid-metszet feladat visszavezetése matroid partíciós problémára.

## k-matroid partíciós probléma

**MPPk**: adott k matroid (Mi=(E,Fi), i=1...k). Kérdés: a matroidok összege a szabadmatroidot adja-e, vagyis E előáll-e E1∪...∪Ek alakban úgy, hogy Ei∈Fi ∀i-re. Feltehető, hogy az Ei halmazok diszjunktak, ezért hívják a feladatot matroid partíciós problémának.

**Lemma**: MPPk NP-beli, mert tanú rá egy particionálás, és a tanú lineáris időben ellenőrizhető.   
**Lemma**: MPPk coNP-beli, mert tanú rá egy X⊆E halmaz, ami biztosan összefüggő az összegben, azaz ∑ri(X)<|X|.

**Lemma**: MPPk P-beli.

### Algoritmus

Induljunk ki az ∀i Ei=∅ állapotból. Ekkor Ei∈Fi. Az Ei halmazokat addig bővítjük, amíg az uniójuk E nem lesz, vagy ha nem bővíthető, mutatunk egy X tanút.

A bővítéshez bevezetünk egy n+k pontú irányított segédgráfot, amelynek

* Csúcsai E elemei ∪ {p1, ..., pk}. pi az Ei partíció segédpontja.
* (x→pi) ∈ E(G), ha x∉Ei és Ei∪{x}∈Fi.   
  Az ilyen típusú élek azt jelképezik, hogy az Ei partícióba felvehető x a függetlenség megsértése nélkül.
* (x→y) ∈ E(G), ha ∃i x∉Ei, y∈Ei, Ei∪{x}∉Fi, de Ei∪{x}-{y}∈Fi.   
  Az ilyen típusú élek azt jelentik, hogy az Ei partícióban az y elem kicserélhető x-re a függetlenség megsértése nélkül.

1. Megkeressük a legrövidebb irányított utat (E-∪Ei)-ből {p1, ..., pk}-ba.
2. Ha van ilyen út, javítunk az út mentén, azaz végrehajtjuk a cseréket. ∪Ei mérete 1-gyel nő. Azért kell a legrövidebb úton végigmenni, mert különben nem garantált, hogy a cserék során nem sérül a partíciók függetlensége.
3. Különben STOP, nemleges a válasz, és a tanú az (E-∪Ei)-ből irányított úton elérhető pontok halmaza.

## 2-matroid-metszet probléma

**MMPk**: adott k matroid (Mi=(E,Fi), i=1...k) és egy p egész szám. Kérdés: létezik-e Fi-knek legalább p méretű közös elemük?

**Tétel**: MMP2∈P.

**Biz**: M1=(E,F1), M2=(E,F2), p egész. Ha p>min(r1,r2), nem a válasz. Csonkoljuk a matroidokat addig, amíg a rangjuk min(r1,r2,p)-re nem csökken. Ezzel a problémát redukáltuk közös bázis keresésére. r1=r2=p-re a válasz akkor és csak akkor igenlő, ha M1∨M2\*=(E,2E).

* ⇒: ha M1-nek és M2-nek van közös B bázisa, akkor M2\*-ban E-B bázis, E=B∪(E-B) egy felbontása az összegnek, azaz az összegben E független, tehát M1∨M2\* szabadmatroid.
* ⇐: legyen E=C∪D egy olyan felbontása a szabadmatroidnak, ahol C∈F1 és D∈F2\*.   
  |E| ≤ |C|+|D| = r1(C)+r2\*(D) ≤ r1(E)+r2\*(E) = r1(E)+|E|-r2(E) = |E|.   
  C és D diszjunkt bázisok ⇒ C közös bázisa M1-nek és M2-nek.

**Def.:** M = (E,F) csonkoltja M′ = (E,F′) ahol F′ az F elemeit tartalmazza a bázisokon kívül. A matroid rangja ettől eggyel csökken.

14. tétel

k-polimatroid rangfüggvény fogalma. A 2-polimatroid-matching probléma, ennek bonyolultsága, Lovász tétele (biz. nélkül).

## k-polimatroid rangfüggvény

**Def.**: f:2E→**N** egy k-polimatroid rangfüggvénye, ha teljesülnek rá a következő axiómák:

1. r(∅)=0
2. r({x})≤k
3. X⊆Y ⇒r(X)≤r(Y)
4. r(X)+r(Y)≥r(X∪Y)+r(X∩Y)

**Speciális eset**: r({x})≤1 (k=1), ekkor r egy matroid rangfüggvénye.

**Megj.**: r({x})≤k-val ekvivalens az r(X)≤k|X| illetve az r(X∪Y)≤r(X)+k|Y| axióma.

## k-polimatroid matching probléma

**Def.**: X⊆E k-polimatroid matching, ha r(X)=k|X|.

**Def.**: k-polimatroid matching probléma: adott r és t∈**N**. Kérdés: van-e legalább t elemű k-polimatroid matching?

**Speciális esetek**:

* Input: tetszőleges G gráf, t∈**N**. Kérdés: ν(G)≥t?   
  2-polimatroid matchingként megfogalmazva: r(X)=|X által lefedett pontok halmaza|≤2|X|
* Input: két matroid, t. Kérdés: létezik-e X⊆E, |X|≥t, X∈F1∩F2 (matroid metszet probléma).   
  2-polimatroid matchingként megfogalmazva: f(X)=r1(X)+r2(X)≤2|X|
* Az utolsó két probléma közös speciális esete: páros gráfban ν(G)≥t?
  + 2. probléma leképzése a 3.-ra.: M1 grafikus matroidban e1 és e2 párhuzamos élek, ha e1-nek és e2-nek felül van egy közös pontja. M2-t hasonlóan értelmezzük a páros gráf alsó ponthalmazán.

### A 2-polimatroid matching probléma bonyolultsága

**Tétel**: A matroidpárosítási probléma (2-polimatroid matching probléma) teljes általánosságban nem oldható meg polinomidőben. A teljes általánosságban kifejezés a függvény megadási módjára vonatkozik: azt jelenti, hogy bármely részhalmazra egy egységnyi idő alatt megtudhatjuk r(X) értékét , de ettől eltekintve a 2-polimatroid rangfüggvényről semmit nem tudunk.

**Biz.**:

Legyen E egy 2n elemű halmaz és E0 egy rögzített n elemű részhalmaza. Két 2-polimatroid rangfüggvényt definiálunk: minden részhalmazra legyen és minden részhalmazra legyen , a pontosan n elemű részhalmazokon azonban r1 legyen azonosan 2n-1, r2 viszont az E0 halmazon legyen 2n és a többin 2n-1 értékű. Mindkét függvény 2-polimatroid rangfüggvény lesz (a szubmodularitás bizonyítását nem részletezzük).

Legyen most p=n. Arra a kérdésre, hogy van-e legalább p-elemű 2-matching, a válasz nyílván nemleges lesz az r1 és igenlő az r2 input függvény esetén. Ha egy algoritmus teljes általánosságban meg tudja oldani a matroidpárosítási problémát, akkor a legrosszabb esetben mind a darab n elemű részhalmazra meg kell kérdeznie az ri függvény értékét, hisz, ha csak egy híján mindegyikre kérdezné meg és mindig 2n-1 választ kapna, akkor még nem tudhatná, hogy melyik esetünkről van szó. Ez pedig már n függvényében exponenciálisan sok lépés (az analízisből ismert, hogy aszimptotikusan egyenlő -nel.

## ****Lovász-tétel****

Vegyünk egy k\*2n méretű valós M mátrixot, oszlopai legyenek rendre M=(a1,b1,a2,b2,…,an,bn), majd definiáljuk az I={1,2,…,n} indexhalmazon egy r függvényt úgy, hogy X⊆I esetén legyen r(X) az   
∪i∈X{ai,bi} vektorhalmaz által kifeszített altér dimenziója. Könnyű látni, hogy ilyenkor r egy 2-polimatroid rangfüggvény.

Lovász-tétel:

A matroidpárosítási probléma polinomidőben megoldható, ha a 2-polimatroid rangfüggvény egy adott valós elemű M mátrixból a fent leírt módon nyerhető.

15. tétel

NP-nehéz feladatok polinomiális speciális esetei: algoritmus a maximális független ponthalmaz problémára és az élszínezési problémára páros gráfokon. Additív hibával közelítő algoritmusok színezési problémákra. A Hamilton-kör probléma visszavezetése a leghosszabb kör probléma additív közelítésére. k-approximációs algoritmusok, példák: minimális lefogó ponthalmaz, maximális páros részgráf keresése.

## Független és lefogó halmazok

*Független élhalmaz*nak nevezzük egy gráf éleinek olyan halmazát, amelyben semelyik két élnek nincs közös pontja. *Független ponthalmaz*nak pedig a pontok olyan halmazát, amelyben semelyik két pont nincs közös élen.

A gráf pontjainak egy halmaza *lefogó ponthalmaz*, ha a gráf minden élének legalább az egyik végpontját tartalmazza; hasonlóan, élek egy halmaza *lefogó élhalmaz*, ha a gráf minden pontjára legalább egy eleme illeszkedik.

Az alábbi jelölések használatosak a lényeges él-, illetve ponthalmazok elemszámára:

* a legnagyobb független élhalmaz elemszáma
* a legkisebb lefogó ponthalmaz elemszáma
* a legkisebb lefogó élhalmaz elemszáma
* a legnagyobb független ponthalmaz elemszáma

Minden G gráfra:

## Gallai tétel

Minden hurokmentes G gráfra, azaz a legkisebb lefogó és a legnagyobb független ponthalmaz elemszámának összege egyenlő a gráf pontjainak számával.

Minden olyan G gráfra, amely nem tartalmaz izolált pontot, , azaz a legnagyobb független és a legkisebb lefogó élhalmaz elemszámának összege egyenlő a gráf pontjainak számával.

## Kőnig tétel

A Kőnig-tétel a gráfelméletben egy páros gráf maximális párosítása és a minimális lefogó ponthalmaza közötti ekvivalenciát mondja ki.

Legyen G egy páros gráf. Ekkor a tétel szerint (azaz a legnagyobb független élhalmaznak ugyanannyi eleme van, mint a legkisebb lefogó ponthalmaznak), és ha G-ben nincs izolált pont, akkor (azaz a legkisebb lefogó élhalmaz azonos méretű a legnagyobb független ponthalmazzal).

***Magyar módszer*** fogja bizonyítani: algoritmus, mely megtalálja mind a négy halmazt, a legkisebb lefogó ponthalmazt, a legkisebb lefedő élhalmazt, a legnagyobb független élhalmazt és a legnagyobb független ponthalmazt. Ezzel bizonyítja König tételét.

## Kőnig tétel

Legyen G = (A,B;E) páros gráf.Ekkor

### Bizonyítás

Megadunk egy (viszonylag) egyszerű algoritmust, amely a G páros gráf éleit megszínezi darab színnel; ebből természetesen a tétel állítása következik.

Jelölje a G éleit e1, e2, . . . , em. Az algoritmus egymás után foglalkozik a G éleivel és a sorra kerülő élt mindig megszínezi a rendelkezésre álló szín valamelyikével (mégpedig helyesen, vagyis szomszédos élek különböző színt kapnak). Eközben előfordulhat, hogy a korábban már megszínezett élek egy részét át kell színezni más színűre, de semelyik él nem változik vissza színtelenné.

Tegyük fel tehát, hogy az e1, e2, . . . , ei−1 élek már kaptak valamilyen színt és legyen ei = {u, v}, ahol és v. Mivel u-ra és v-re is legfeljebb él illeszkedik, de ezek között még van színtelen (nevezetesen ei), ezért mindkét csúcsnak van „szabad színe” - vagyis olyan szín, amilyen színű él még nem illeszkedik a csúcsra. Ha u-nak és v-nek van közös szabad színe, akkor persze ei megkaphatja ezt a színt és készen vagyunk. Tegyük fel, hogy nem ez a helyzet. Legyen u egy szabad színe a piros, v egy szabad színe a kék. Legyen C a G-nek az a részgráfja, amely (G összes csúcsából és) a jelenlegi színezés szerint pirosra vagy kékre színezett élekből áll. Ekkor C-ben minden pont foka legfeljebb 2, hiszen minden csúcsra legfeljebb egy piros és egy kék él illeszkedhet. Így a C diszjunkt utakból és körökből áll. Ráadásul u és v foka C-ben csak 1 lehet, hiszen u-nak a piros, v-nek a kék szabad színe. Vagyis u és v is egy-egy C-beli út végpontja.

Legyen Pu az a C-beli út, amelynek a végpontja u. Állítjuk, hogy v nincs rajta Pu-n. Ha ugyanis rajta volna, akkor nyilván v volna a Pu másik végpontja. Ekkor Pu páratlan sok élből állna, hiszen a páros gráf egyik osztályából (A-ból) indul és a másikban (B-ben) ér véget. Mivel Pu-n nyilván váltakozva következnek kék és piros élek és az első (u-ra illeszkedő) éle kék (hiszen u-nak a piros szabad színe), ezért az utolsó éle is kék kell legyen (mert Pu páratlan sok élű). Így Pu végpontja mégsem lehet v, mert arra nem illeszkedik kék él.

Fessük át a Pu kék éleit pirosra és a piros éleit kékre. Ezzel a színezést nyilván nem rontottuk el, viszont mostantól az u-nak a kék szabad színe lesz. Mivel a Pu élein zajló színcsere (a fentiek szerint) a v-re illeszkedő éleket nem érintette, ezért a kék v-nek is szabad színe maradt. Így az ei él kékre színezésével továbbra is helyes színezést kapunk.

Megjegyezzük, hogy a fenti bizonyításban leírt algoritmus hatékony is: a segítségével nagyméretű páros gráfok élei is „gyorsan” megszínezhetők a lehető legkevesebb, vagyis színnel.

## Probléma osztályok

A probléma osztályok:

* P: polinomiális időben megoldható
* NP: eldöntési probléma, létezik az igen válaszra ellenőrizhető tanú
* co-NP: eldöntési probléma, létezik a nem válaszra ellenőrizhető tanú
* NP-nehéz: bármely NP-beli probléma visszavezethető rájuk

NP

co-NP

NP-teljes

P=NP ???

P

NP-nehéz

|  |  |  |
| --- | --- | --- |
| **feladat** | **probléma osztály** | **algoritmus** |
| összefüggőség | P | mélységi keresés (DFS), szélességi keresés (BFS) |
| van-e TP páros gráfban | P | javító utas algoritmus |
| van-e 2-színezése | P | BFS |
| van-e 3-színezése | NP-teljes | - |
| síkbarajzolhatóság | P | Kuratowski-tétel |
| klikk | NP-nehéz | - |
| MAXFTL | NP-nehéz | - |
| MAXFTL páros gráfon | P | magyar módszer |
| Hamilton-kör | NP-teljes | - |
| utazóügynök probléma | NP-nehéz | - |
| a maximális élszámú (s,t)-vágás keresése | NP-nehéz | - |
| a maximális élszámú (s,t)-vágás keresése, ha a hálózat síkba rajzolható | P | - |
| meghatározása | NP-nehéz | - |
| meghatározása, ha a G gráf páros | P | - |
| leghosszabb irányított út keresése | NP-nehéz | - |
| leghosszabb irányított út keresése, ha a gráfban nincs irányított kör (Directed Acyclic Graph, DAG gráf) | P | szélességi keresés (BFS) |

## Additív hiba

*f(x)* a minimalizálandó célfüggvény az *XI* halmazon, amely *I* inputon értelmezett. Egy algoritmus *C* additív hibával közelítve old meg egy minimalizálási problémát, ha minden *I* inputra polinom időben ad egy megoldást, amire:

Példa: Egyszerű gráf (Olyan gráf, amely nem tartalmaz se hurokélt, se irányított élt, se többszörös élt.) élkromatikus számának meghatározása. Ha a legnagyobb fokszámú pont fokszáma K, akkor Vizing tételéből tudjuk, hogy az élkromatikus szám K vagy K+1. Ezt eldönteni NP-teljes, de a tétel bizonyítása alapján K+1-es színezés polinomiális időben megkapható, azaz 1 additív hibájú algoritmusunk van.

Példa 2: Kromatikus szám (színezés) keresése. Az ötszíntétel alapján 5 színre beszínezhetünk polinomiális időben.

Optimális színezés síkgráfokon:

* üres-e (igen -> 1 szín)
* páros-e (igen -> 2 szín)
* 5 színezés algoritmus (ekkor )

Ugyanakkor megmutatjuk, hogy egy gráfban a leghosszabb út megkeresésének problémája (amely NP-teljes, hiszen a Hamilton-út keresését magába foglalja) semmilyen additív C konstanssal közelítő algoritmussal nem oldható meg (hacsak P=NP nem teljesül). Tegyük fel, hogy lenne egy olyan Ac algoritmus, mely tetszőleges G gráfra eldöntené, van-e benne egy olyan út, melynek pontszáma legfeljebb C-vel kisebb a leghosszabb út hosszánál. Megmutatjuk, hogy erre polinomrendben visszavezethető lenne az a probléma, hogy adott H gráfról döntsük el, van-e benne Hamilton-út. Egészítsük ki ugyanis a H gráfot C darab izolált ponttal, majd az így kapott G gráfra alkalmazzuk az Ac algoritmust. Nyílván akkor és csak akkor találna az algoritmus megfelelő utat G-ben, ha az annak megfelelő élek H-ban Hamilton-utat alkotnának.

## Multiplikatív hiba

Egy algoritmus multiplikatív hibával közelítve old meg egy minimalizálási problémát, ha minden *I* inputra polinom időben ad egy megoldást, amire:

## Minimális lefogó ponthalmaz

A lefogó pontok minimális számát () határozzuk meg úgy, hogy kiválasztjuk a független élek maximális rendszerét (), akkor mindkét végpontját tekintve -elemű ponthalmazt kapunk. Világos, hogy (legfeljebb annyi független él választható ki, mint a lefogó pontok minimális száma – definíció szerint), ezért multiplikatív hibáról van szó.

## Maximális páros részgráf keresése

Keressük *n* pontú gráf olyan kettéosztását, ahol a két pontosztály között a legtöbb él halad. Ez NP-nehéz.

Kiindulunk egy tetszőleges kettéosztásból, és ha egy *p* pont áthelyezése növeli a vágás élszámát, akkor áthelyezzük. Ez a maximum felét legalább kiadja.

Lépésszám becslése maxk*(k\*(n-k))*, azaz kb. *n*2/4.

16. tétel

A minimális lefogó ponthalmaz probléma visszavezetése a halmazfedési feladatra, a halmazfedési feladat közelítése. Közelítő algoritmus a Steiner-fa problémára, éles példa.

## Minimális lefogó ponthalmaz visszavezetése halmazfedésre

A lefedendő alaphalmaz a gráf élhalmaza (az elemek az élek), a lefedő halmazokat pedig a pontokhoz rendeljük úgy, hogy az elemei az adott pontból kiinduló élek. Minden lefedő halmaz költsége egységnyi. A halmazfedési feladat tehát NP-nehéz.

## A halmazfedési feladat közelítése

*A feladat:* Adott U alaphalmaz (n elemű), S1⊆U, ..., Sk⊆U részhalmazai és c:{Si}→R+ költségfüggvény. Keressük a minimális összköltségű fedést.

*Algoritmus:*

* C⊆U az U halmazból már lefedett elemek részhalmaza.
* *while (C!=U)* válasszuk azt az Si halmazt, amire minimális.

*Lemma:* legyen , ahol Si az a halmaz, ami az algoritmus során először fedte le e-t.   
e1, ..., em (m=|U|) az elemek az algoritmus során kapott lefedési sorrendben.

*Bizonyítás:* ∑p(e) = fedés összköltsége. Mivel az algoritmus mindig a minimális értékű halmazt választja, .

*Következmény:*

## Steiner-fa probléma

Adott egy összefüggő gráf, élein költségfüggvény. *V* két részre van osztva: *T*-beli terminálokra és *S*-beli Steiner pontokra. Cél minimális költségű fát keresni *G*-ben, ami az összes terminált tartalmazza (és esetleg pár Steiner-pontot is). Ha , akkor a feladat mohó algoritmussal hatékonyan megoldható, különben NP-nehéz.

A metrikus Steiner-fában a költségfüggvény kielégíti a háromszög-egyenlőtlenséget  
. Ez esetben létezik *k*-approximációs algoritmus.

## Tétel

Az általános Steiner-fa probléma visszavezethető a metrikus Steiner-fa problémára, úgy hogy a   
*k*-approximáció megmarad.

### Bizonyítás

A Steiner-fa feladat egy példányát (amely egy G = (V, E) gráfból, egy c költségfüggvényből és a   
V= T U S felosztásból áll) polinomidőben átalakítjuk a metrikus Steiner-fa feladat egy példányává. Legyen *G*’ a teljes gráf *V* ponthalmazon. Egy *G*’-beli {*u*, *v*} él költsége a *G*-beli legkisebb összköltségű *u* – *v* út hossza (Dijkstra-algoritmus – legrövidebb utak keresése). Mivel minden {*u*, *v*} él költsége *G*’-ben legfeljebb akkora, mint *G*-ben, ezért az optimális megoldás értéke is legfeljebb akkora, mint *G*-ben.

Legyen *F*’ egy *G*’-beli Steiner-fa. Helyettesítsük *F*’ éleit *G*-ben a legrövidebb utakkal, hogy megkapjuk *F* Steiner-fát. *G*’ és *G* csak élekben különbözik, így *F*’ után *F* is tartalmazza az összes terminált, összköltsége . Mivel *F* polinom időben számítható *F’*-ből, ezért megvagyunk.

## Tétel

Legyen *F* egy minimális költségű feszítőfa a terminálok halmazán. Ekkor .

### Bizonyítás

Vegyünk egy optimális Steiner-fát. Ennek éleit megduplázva olyan összefüggő gráfot kapunk, ami minden terminált összeköt. Ebben keressünk Euler-kört (van rá hatékony algoritmus). A költsége 2*OPT*.

Vegyük sorba az Euler-kör csúcsait, és dobjuk el azokat, amelyeket nem először érintettünk, vagy amelyek Steiner-pontok (mivel *G* teljes gráf, továbbra is létezik út, és a háromszög-egyenlőtlenség miatt az összköltség sem növekszik). A csúcsokat összekötve kaptunk egy Hamilton-utat a *T* halmazon, melynek költsége legfeljebb 2*OPT*. Ez az út feszítőfa is a terminálokon, tehát .

Éles példa a következő. Legyen az n+1 pontú teljes gráfon n terminál és egyetlen Steiner-pont. A Steiner-pontra illeszkedő élek költsége legyen 1, a többi él legyen 2 költségű (ez metrikus). Itt az optimum n, a terminálokon vett minimális költségű feszítőfa költsége viszont 2(n-1).

17. tétel

Az általános utazóügynök probléma közelíthetősége. Közelítő algoritmusok a metrikus utazóügynök problémára, Christofides algoritmusa.

## Utazó ügynök

Adott egy *G* teljes gráf, nemnegatív élsúly függvény. Keressük a minimális összsúlyú Hamilton-kört. A probléma NP-nehéz.

## Általános utazó ügynök közelítése

Általános utazó ügynök nem közelíthető *k*-approximációval. A probléma visszavezethető (egy nem teljes gráfban) Hamilton-kör keresésére.

Legyen *G* egy *n*-pontú (nem teljes) gráf, a bemenet, hozzá tartozó *G*’ pedig teljes gráf. Ha *G*-ben szomszédos két pont, akkor *G*-ben egységnyi az élsúly, egyébként *kn*. Ha *G*-ben van Hamilton-kör, akkor *G*’-ben a minimális összsúlyú Hamilton-kör összsúlya *n*, egyébként legalább . Mivel ez több, mint az optimum *k*-szorosa, ezért nincs ilyen közelítés. (Az algoritmus meg tudná különböztetni a Hamilton-körrel rendelkező és nem rendelkező gráfokat.)

Ugyanez *kn* helyett tetszőleges *f(n)* függvénnyel is eljátszható.

## Metrikus utazóügynök probléma

**Feladat**: adott G teljes gráf, és egy c:E→**R**+ élsúlyozás, amire teljesül a háromszög-egyenlőtlenség. Keressük a minimális összsúlyú Hamilton-kört.

**2-approximációs algoritmus**: vesszük a minimális feszítőfát, megduplázzuk az éleit, keresünk egy Euler-kört és leszűkítjük Hamilton-körré.

c(talált Hamilton-kör) ≤ c(Euler-kör) = 2\*c(minimális feszítőfa) ≤ 2\*c(minimális Hamilton-út) < 2\*c(minimális Hamilton-kör) = 2\*OPT

Minimális összsúlyú feszítőfa keresése: Kruskal-algoritmus

**Christofides algoritmus**: vesszük a minimális feszítőfát és egy minimális összsúlyú teljes párosítást a fa páratlan fokú csúcsai (páros darab van) által kifeszített G' részgráfban. A fa és a párosítás élei által meghatározott részgráfban (n-1+m él) keresünk egy Euler-kört, amit leszűkítünk Hamilton-körré.

**Tétel**: a Christofides algoritmus 3/2-approximációs.

**Biz.**: a minimális feszítőfa összsúlya < OPT, elég azt belátni, hogy a párosítás éleinek összsúlya ≤ OPT/2. Jelöljük a minimális Hamilton-kört G gráfban H-val, G' segédgráfban H'-vel. c(H')≤OPT, mert H-t levágásokkal át tudjuk alakítani úgy, hogy csak G'-beli csúcsokat tartalmazzon, miközben nem nő az összköltsége, és az így kapott G'-beli Hamilton-körnél H' nyilván nem nagyobb súlyú. A minimális összsúlyú teljes párosítás pedig legfeljebb c(H')/2 költségű, mert H' páros élei és páratlan élei is teljes párosítást alkotnak, a kettő költségének összege c(H'), tehát van legfeljebb c(H')/2 összköltségű párosítás, aminél nem nagyobb a minimális összsúlyú teljes párosítás.

18. tétel

Teljesen polinomiális approximációs séma a Részösszeg problémára

## Polinomiális approximációs séma

**Def.**: egy probléma polinomiális approximációs sémával közelíthető, ha ∀ε>0-ra van rá   
(1+ε)-approximáció. Ez nem mindig elég, mert ha a közelítő algoritmus lépésszáma pl. 21/εn4, még mindig exponenciálisan hosszú ideig fut. (Ilyenkor ugyanis rögzített ε-ra polinomiális az algoritmus, tehát a feltételnek megfelel.)

**Def.**: egy probléma teljesen polinomiális approximációs sémával közelíthető, ha ∀ε>0-ra van rá   
(1+ε)-approximáció, ami 1/ε-ban is polinomiális.

**Pl.**: a metrikus utazó ügynök problémára nincs P approximációs séma, de az euklideszi utazó ügynök problémára van teljesen P approximációs séma.

### Részösszeg probléma

Adott A = {*a*1, *a*2, ..., *an*} és t.

Kérdés: létezik-e *B*⊆*A* úgy, hogy Σ*bi* = t?

Speciális eset: partíció probléma, amikor t = Σ*ai*/2, még ez is NP-teljes.

### Részösszeg optimalizálási probléma

Adott A = {*a*1, *a*2, ..., *an*} és t.

Keressük *B*⊆*A*-t úgy, hogy Σ*bi* maximális és Σ*bi* ≤ t legyen.

A feladat NP nehéz, mert a részösszeg probléma visszavezethető rá.

Bizonyítás: ha találunk olyan B halmazt, amire Σ*bi* = t, akkor igen a válasz a részösszeg problémára, különben nem.

A feladat nem NP-beli, mert nem eldöntési probléma, tehát nem is NP-teljes.

### Közelítő algoritmus a részösszeg optimalizálási problémára

**Tétel**: a probléma teljesen polinomiális sémával közelíthető.

A bizonyítás egy konkrét algoritmus lesz:

1. Pontos megoldást adó algoritmus:  
   Ez az algoritmus gyakorlatilag nem más, mint egy „brute-force” megoldó, amely minden lehetséges részösszeget kiszámol.

Tegyük fel, hogy *a*1 ≤ *a*2 ≤ ... ≤ *an*. Definiálunk két halmazsorozatot:

* + *L*0 = {0}
  + *Li*' = {*l*+*ai* | *l*∈*Li*}
  + *Li*+1 = *Li* ∪ *Li*'

Röviden: *Li*+1 = *Li* ∪ (*Li*+*ai*+1)

Például:

* + *a*1=3, *a*2=5, *a*3=7
  + *L*0 = {0}
  + *L*0' = {3}, *L*1 = {0,3}
  + *L*1' = {5,8}, *L*2 = {0,3,5,8}
  + *L*2' = {7,10,12,15}, *L*3 = {0,3,5,7,8,10,12,15}

Az optimális részletösszeg max{*l*|*l*∈*Ln* ∧ *l*≤*t*} lesz.

1. Polinomiális közelítő algoritmus:

Egyrészt vegyük észre, hogy az *Li*' halmazokból nincs szükségünk a *t*-nél nagyobb elemekre. Képezzük a halmazokat a következő módon: *Li*' = (*Li*+*ai*) ∩ [0..*t*].

Def.: δ-val ritkítás

*L* növekvő sorrendbe rendezett halmaz.

foreach (*l*∈*L*) {

*m* az *l*-et megelőző, halmazban hagyott elem, vagy ha nincs ilyen, 0.

if (*l*<m(1+*δ*)) *l*-et kidobjuk.

}

A ritkítás után bármely két szomszédos elem hányadosa legalább 1+*δ*. A ritkított halmaz mérete felülről becsülhető: |*L*ritkított| ≤ log1+*δt*+2.

1. **Tétel**: ha az algoritmus során a halmazok *t*-nél nagyobb elemeit minden lépésben levágjuk és az eredményt *δ*=*ε*/2*n*-vel ritkítjuk, (1+*ε*)-approximációt kapunk a problémára, és a lépésszám *n*-ben, log *t*-ben és 1/*ε*-ban is polinomiális lesz.

**Biz.**: az algoritmus (1+*ε*)-approximációs.

**Lemma:**

**Biz.:**

X=0-ra mindkét oldal 0

X>0-ra:

**A derivált >0, így a függvény monoton nő, ezzel bebizonyítottuk a lemmát.**

A lemma alapján:

**Visszahelyettesítve:**

Az algoritmus lépésszáma mindhárom mennyiség szerint polinomiális.

Az algoritmus során n db lista összefésülést kell végezni, aminek a komplexitása O(∑|Li|) = O(n|Li|). A fenti becslések alapján látszik, hogy O(n|Li|) n-ben négyzetes, log(t)-ben lineáris, tehát mindkettőben polinomiális.

Olyan megoldást keresünk, amely

Ahhoz, hogy megmutassuk, hogy a kapott válasz relatív hibája kicsi, vegyük figyelembe, hogy amikor az Li listát ritkítjuk, egy legfeljebb nagyságú hibát okozunk a megmaradó képviselő értékek és a ritkítás előtti értékek között.

i-re vonatkozó teljes indukcióval be lehet látni, hogy :

**Ahhoz, hogy**  teljesüljön szükséges.

A bizonyítás során felhasználásra került a binomiális tétel és a mértani sor összegképlete.

19. tétel

Ütemezési feladatok típusai. Az 1|prec|Cmax és az 1||∑Cj feladat. Listás ütemezés a P|| Cmax feladatra tetszőleges, illetve LPT sorrendben (ez utóbbi biz. nélkül). Listás ütemezés a P|prec|Cmax feladatra (bizonyítás nélkül), az LPT sorrend szerepe. Példa arra, hogy a leghosszabb út szerinti sorrend nem 3/2-approximáció. Coffman-Graham (biz. nélkül), illetve Hu algoritmusai (biz. nélkül).

## Ütemezési feladatok alapjai

* Definíciók:
  + Ji munkák (job)
  + pi megmunkálási idő (processing time)
  + ωi súly (weight)
  + di határidő (due date)
  + ri rendelkezésre állási idő (release time)
  + Ci befejezési idő (completion time)
* egy adott időben egy gépen egy munka folyik
* célfüggvényre optimalizálunk (pl.: max(Ci): teljes átfutási idő)

## Ütemezési feladatok osztályozása

Az ütemezési feladatok egy α|β|γ hármassal írhatók le, ahol

* α a gépek száma
  + 1: 1 gép
  + Pm: m párhuzamosan futó gép
  + P: nem rögzített számú párhuzamosan futó gép. P|β|γ az {1|β|γ, P2|β|γ, P3|β|γ, ...} feladatokat tartalmazó osztály neve.
* β az infók halmaza az ütemezésről. Pl.:
  + prec: adott egy irányított gráf (DAG), ami megkötéseket tartalmaz arra nézve, hogy egy adott munka elkezdéséhez mely munkákat kell előbb befejezni
  + rj: adottak a release time-ok, azaz hogy melyik job mikortól áll rendelkezésre
  + pj: adottak a megmunkálási idők, mennyit vesz igénybe az adott munka elvégzése
* γ a függvény, ami szerint optimalizálunk
  + Cmax = max(Cj): az utolsó munka befejezési ideje
  + ∑Cj ~ ∑Cj/n: a munkák átlagos befejezési ideje

## 1| |Cmax

Optimális megoldást ad minden olyan megoldás, melynél a gép folyamatosan dolgozik. Ennek értéke a munkák elvégzési idők összege.

## 1|prec|Cmax

A feladatokat topologikus sorrendben adogatjuk a gépnek. Ez optimális, mert folyamatosan foglalt a gép.

## Algoritmus a topologikus sorrend meghatározására

Az algoritmus egy adott *s* pontból indul. Erről vagy előre ismert, hogy minden pont elérhető belőle, vagy az algoritmus adja hozzá (extra pont, amelyet a végén törlünk). A gráfba még beteszi az összes *sv* élt, ahol *v* eleme *V*.

Minden pontra két címkét tart számon. Az egyik címke azt mutatja, hogy a pont elért-e, és ha igen, akkor melyik csúcsból. A másik címke pedig azt, hogy a csúcs átvizsgált-e, vagy sem. Kezdetben *s* elért, nem átvizsgált.

Amíg van elért, de nem átvizsgált pont, addig ismétli a következőket:

Kiválasztja a legkésőbb elért, nem átvizsgált *u* pontot. Eldönti, hogy van-e egy *uv* él a gráfban, aminek *v* végpontja nem elért. Ha talál ilyet, akkor átállítja *v* címkéjét: *v* elért az *u* pontból. Ha nem talál, akkor átállítja *u* címkéjét: *u* átvizsgált, és feljegyzi a címkébe, hogy éppen hányadik lépésnél tart.

Az átvizsgálási sorrend topologikus sorrendet ad. Ha az *s* pont nem volt az eredeti gráfban, akkor törli.

Megfelelő adatstruktúrával mindez az élszámmal arányos időt vesz igénybe.

|  |  |
| --- | --- |
| [Directed acyclic graph.png](http://hu.wikipedia.org/w/index.php?title=F%C3%A1jl:Directed_acyclic_graph.png&filetimestamp=20071206165901) | A bal oldalon látható gráfnak több topologikus rendezése is létezik, köztük:   * 7,5,3,11,8,2,10,9 * 7,5,11,2,3,10,8,9 * 3,7,8,5,11,10,9,2 * 3,5,7,11,10,2,8,9 |

## 1||∑Cj

Munkaigény szerint növekvő sorrendben adogatjuk a feladatokat a gépnek. p1≤p2≤...≤pn. Az algoritmust *SPT*-nek (Shortest Processing Time) hívjuk.

Az SPT ütemezés optimális, mert véges sok sorrend létezik és egy nem SPT ütemezés javítható. Ha az ütemezés J1, ..., Jn sorrendben veszi a feladatokat, ahol valamelyik i-re pi>pi+1, akkor a két feladatot megcserélve ∑Cj nő.

## P||Cmax (Graham)

A 2||Cmax feladat NP nehéz, mert visszavezethető rá a partíciós probléma. P||Cmax is NP nehéz, mert speciális esetként tartalmazza 2||Cmax-ot.

**Def.**: listás ütemezés (Graham): A joboknak vesszük egy előre rögzített sorrendjét. Ha egy gép felszabadul, a listában következő munkát azonnal odaadjuk neki (ha van még).

**Tétel**: a listás ütemezés approximálja a Pm||Cmax feladatot. (m: gépek száma)

**Biz.**:

Könnyen látható, hogy a listás ütemezés polinomidőben elkészíthető. Az approximációs faktor igazolásához legyen:

* utoljára végződő munka (nem ugyanaz, mint az utolsóként elkezdett!)
* megkezdésének ideje

Így: (az LS segítségével kapott eredmény)

Mivel minden gép folyamatosan foglalt legalább a t időpillanatig (különben az LS algoritmus szabálya szerint a munkához valamely gép már előbb hozzákezdett volna), így:

Jelölje az optimális teljes átfutási időt. Ekkor:

Ezek alapján levezethető, hogy:

Tehát sikerült igazolni az approximációs faktor helyességét.

**Tétel**: Az LPT (Longest Processing Time) szerinti listás ütemezés 4/3-approximációs algoritmus a P||Cmax feladatra.

## Probléma a listás ütemezéssel

Példa arra, hogy a listás ütemezés nem mindig jó:

2 munka, 1-1 elvégzéshez szükséges időegység (1-es sebesség esetén), 2 gép (1db 1-es, 1db 10-es sebességű)

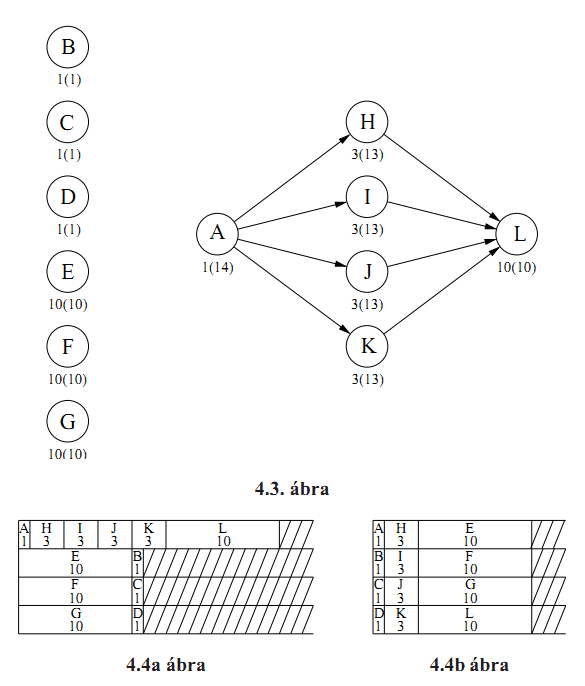
Listás ütemezéssel eredmény 1.1, viszont optimális 2/10 (mindkét munkát 10-es sebességű gép végzi el).

## P|prec|Cmax (Graham)

**Tétel**: a listás ütemezés (ha van szabad gép, a sorban az első olyan jobot teszem fel rá, ami már elkezdhető) approximációs a P|prec|Cmax feladatra is.

## Példa: a leghosszabb út szerinti sorrend nem mindig optimális

Baloldali rendezés leghosszabb út szerint történt (a precedenciagráf figyelembevétele miatt rendez így a gép), jobb oldali pedig az optimális.



## P2|prec, pj=1|Cmax

A feladatosztály bonyolultsága

* P|prec, pj=1|Cmax NP-nehéz
* P2|prec, pj=1|Cmax Coffman és Graham algoritmus ad optimális megoldást
* Pm|prec, pj=1|Cmax bonyolultsága ismeretlen
* P|prec, pj=1, in-tree|Cmax-re a Hu-algoritmus polinom időben optimális megoldást ad.

## Coffman és Graham algoritmusa

A P2|pi=1, prec|Cmax  feladatra ad megoldást.

1. A precedencia gráfon végezzünk tranzitív redukciót.
2. Osszuk csoportokra a redukált gráf pontjait „jobbról balra” haladva úgy, hogy az első csoportban legyenek azok a pontok, melyek ki-foka 0, a második csoportban azok, amelyek ki-foka 0-vá válik, ha elhagyjuk az első csoport pontjait, stb.. Ezután az első csoport pontjait számozzuk meg tetszőlegesen 1-től k1-ig. Ha már a p-edik csoport pontjainak számozását   
   (kp-1+1-től kp-ig) ismerjük, akkor a (p+1)-edik csoport pontjait az alábbi módon számozzuk. Először minden ilyen pont mellé odaírjuk csökkenő sorrendben azon pontok sorszámát, amelyekbe belőlük él vezet. Ezután ezeknek a számsorozatoknak a nemcsökkenő lexikografikus sorrendje jelöli ki kp+1, kp+2, …, kp+1 sorszámokat.
3. Ezután az algoritmus a fenti sorszámok szerint csökkenő sorrendbe rakja a munkákat, majd ezzel a rögzített sorrenddel listásan ütemez.

**Def.**: A tranzitív redukció során minden olyan esetben, ahol az A-ból C-be menő él benne van a gráfban, de érvényes az A->B és B->C feltétel is valamely B-re, az AC élt töröljük az irányított gráfból. Ez nem változtat a megelőzési feltételeken, de csökkenti a precedencia gráf éleinek számát.

## Hu algoritmusa

A P|pi=1, prec|Cmax  feladatra ad megoldást, ha a precendenciagráf *in-tree (be-fenyő)* tulajdonságú.

1. A precedencia gráf minden csúcsához határozzuk meg a gyökérbe vezető út hosszát.
2. Alkalmazzuk a listás ütemezést úthossz szerint csökkenő sorrendben.

**Def.**: Az *in-tree (be-fenyő)* egy olyan irányított gyökeres fa, aminek az élei a gyökér felé vannak irányítva.

20. tétel – Megbízható hálózatok terv.

Globális és lokális élösszefüggőség és élösszefüggőségi szám fogalma. λ(G) meghatározása folyamok segítségével (négyzetes es lineáris számú folyamkereséssel).

## Megbízható hálózat

?

Teendők:

* megbízhatóság kiszámítása
* megbízható hálózat tervezése minimális költséggel
* megbízhatóság növelése minimális költséggel

## Összefüggőségek

**Def.:** Egy gráfban pontok közötti lokális élösszefüggőség () az *u* és *v* pontok közötti éldiszjunkt utak száma.

**Def.:** Egy gráfban pontok közötti lokális összefüggőség () az *u* és *v* pontok közötti belsőleg pontdiszjunkt utak száma.

**Def.:** Egy G gráf k-összefüggő (pontösszefüggő), ha létezik k+1 csúcsa és bármely maximum k-1 darab csúcsát elhagyva összefüggő marad.

**Def.:** Egy G gráf k-élösszefüggő, ha bármely maximum k-1 darab élét elhagyva összefüggő marad.

A k-összefüggőségből következik a k-élösszefüggőség.

**Def.:** A élösszefüggőségi szám az a maximális k érték, hogy a G gráf még k-élösszefüggő.

**Def.:** A összefüggőségi szám az a maximális k érték, hogy a G gráf még k-összefüggő.

Általánosan igaz, hogy:

Példa arra, hogy a gráf 2-élösszefüggő, de nem 2-összefüggő:



## Menger-tétel

**Def.:** Egy F élhalmaz lefogó élhalmaz, ha a G (irányított vagy irányítatlan) gráf F élhalmaza lefog minden u-v utat, ha a G-F gráfban nem létezik u-ból v-be (irányított) út.

1. Ha G egy irányított gráf, , akkor az s-ből t-be vezető éldiszjunkt irányított utak maximális száma megegyezik az összes irányított s – t utat lefogó élek minimális számával.
2. Ha G egy irányított gráf, két nem szomszédos pont, akkor az s-ből t-be vezető, végpontoktól eltekintve pontidegen irányított utak maximális száma megegyezik az összes irányított s – t utat s és t felhasználása nélkül lefogó pontok minimális számával.
3. Ha G egy irányítatlan gráf, , akkor az s-ből t-be vezető élidegen irányítatlan utak maximális száma megegyezik az összes irányítatlan s – t utat lefogó élek minimális számával.
4. Ha G egy irányítatlan gráf, két nem szomszédos pont, akkor s-ből t-be vezető pontidegen irányítatlan utak maximális száma megegyezik az összes irányítatlan s – t utat s és t felhasználása nélkül lefogó pontok minimális számával.

A téma szempontjából lényeges állítás: Az u és v közötti éldiszjunkt utak maximális száma megegyezik az u és v közötti utakat lefogó élek minimális számával.

## Lokális összefüggőségek

**Áll.:** A élösszefüggőségi szám a lokális élösszefüggőségi számok közül a legkisebb.

**Biz.:**

u

v

Bármely két pont között van legalább k éldiszjunkt út, ekkor k-1 él elhagyásával nem eshet szét a gráf, tehát legalább k-élösszefüggő. (Menger-tétel)

Kiválasztjuk azt az (u,v) csúcspárt, amely csúcsok között a minimum felvétetik. A két csúcs között legfeljebb k darab éldiszjunkt út fut, ezeket le tudjuk fogni k éllel. Ezt a k élt törölve G szétesik, G tehát nem lehet (k+1)-élösszefüggő, legfeljebb k-élösszefüggő.

**Áll.:** A összefüggőségi szám a lokális összefüggőségi számok közül a legkisebb.

**Biz.:**

Előzőhöz hasonlóan.

## λ(G) meghatározása folyamok segítségével

Ha a gráfot egy hálózati folyamként értelmezzük, ahol minden él kapacitása egy, akkor *λ(u,v)* épp a maxfolyam probléma.

Ezt a problémát minden pontpárra megoldhatjuk, azaz összesen O(*n*2) db folyamot kell keresnünk. Megfigyelés, hogy általában *n*-1 folyamprobléma megoldása is elég.

Maximális folyam keresésére használhatjuk az Edmonds-Karp algoritmust. Ez az eredeti Ford-Fulkerson algoritmus továbbfejlesztése. A Ford-Fulkerson algoritmus irracionális kapacitásértékek előfordulása esetén nem állt le garantáltan, míg az Edmonds-Karp igen.

**Működése:**

1. Inicializálás: kezdeti folyam létrehozása (pl. ekvivalens 0 érték minden élre)
2. Javító folyam keresése. Ford-Fulkerson esetén ennek módszere mindegy, Edmonds-Karp esetén ennek a legrövidebb lehetséges javító folyamnak kell lennie, ami pl. szélességi kereséssel megtalálható.
3. Ha találtunk javító folyamot, akkor annak értékével megnöveljük az oda-élekre a folyamértéket, és lecsökkentjük a vissza-élekre a folyamértéket (ha van ilyen). Majd folytatjuk a 2. lépéstől.  
   Ha nincs megfelelő javító folyam, leállunk: találtunk egy maximális folyamot.

Lépésszáma O(|*V*|\*|*E*|2).

**A segédgráf létrehozása:**

Az irányított G gráfon adott f folyamhoz definiálunk egy Gf irányított gráfot:

Legyen V(Gf)=V(G) és Gf-ben fusson egy irányított él x-ből y-ba, ha vagy és f(x,y)<c(x,y) (f: folyam értéke, c: élek kapacitása) vagy és . Könnyen látható, hogy ha   
Gf-ben van egy irányított út s-ből t-be, akkor az ennek az útnak megfelelő élek G-ben épp egy javító utat adnak az f folyamra nézve. Ha pedig van javító út G-ben, akkor lesz irányított út s-ből t-be   
Gf-ben.

Így tehát elegendő megnézni, hogy az így kapott Gf gráfban van-e s-ből t-be út. Ezt viszont BFS algoritmussal könnyen eldönthetjük.

**Lineáris számú folyamkeresés:**

Ekkor n-1 folyamproblémát kell megoldani. Lépésszáma O(|*V*|\*|*V*|3)= O(|*V*|4).

**Biz.:**

A négyzetes számú folyamkereséshez képest szűkebb halmazon vizsgálódunk, ezért azt kell garantálnunk, hogy bármely a csúcsból kiindulva a értékek között lesz értékű. Legyen . Nyílván minden csúcsra igaz, hogy:

azaz .

Hagyjunk el G-ből k élt úgy, hogy szétessen, ekkor a kapott részgráfnak legalább két komponense lesz. Mivel a rögzített, válasszuk ki b-t úgy, hogy a és b külön komponensben legyenek, ekkor . miatt , azaz lesz olyan b csúcs, ahol a -nek megfelelő minimum felvétetik.

21. tétel – Megbízható hálózatok terv.

λ(G) meghatározása összehúzások segítségével, Mader tétele (biz. nélkül), Nagamochi és Ibaraki algoritmusa (biz. nélkül).

## Összehúzás fogalma



Egy G gráfban az u és v pont összehúzásával keletkező pontot nevezzük uv-nek. Ha az eredeti gráfban u és v között volt él, abból az új gráfban uv-hez tartozó hurokél lesz. Ha az eredeti gfráfban u-ból és p-ből is ment él egy közös p pontba, akkor abból az új gráfban uv és p közti párhuzamos élek lesznek.

## Összefüggőség kiszámítása összehúzásokkal

G-ből készítünk egy hálózatot egységnyi élsúlyozással, és λ(G) meghatározásához keressük a maximális folyamot, vagyis a minimális vágást. Alkalmazzunk most olyan módszert, amihez nem kell folyamot számolni.

**Tétel:** Két pont összehúzásával a vágás nem csökkenhet, és csak akkor nőhet, ha az összes minimális vágás a két pont között halad át. Vegyük tehát , ahol *Gu,v* a *G*-ből az *u* és *v* pontok összehúzásával kapott gráf. Ezt *n*–1-szer kiszámolva megkapjuk *λ(G)*-t.

**Biz.:** Belátjuk, hogy legfeljebb , legfeljebb , és az egyikkel egyenlő is.

* . Triviális a definíció miatt.
* Ha nem összefüggő, akkor az összehúzás előtt G sem volt az (az állítás megfordítva nem igaz, ha G nem összefüggő, akkor lehet az, pl.: egy C2-ből és egy izolált pontból álló gráf esetén a C2 egyik pontját az izolált ponttal összehúzva összefüggő gráfot kapunk).  
   esetén -nek van k olyan éle, amit elhagyva szétesik. A -beli vágás G-re transzformálva a megfelelő k élt elhagyva G is szétesik, mert ha nem így lenne, akkor az összehúzás utáni is összefüggő lenne. Ezért biztosan teljesül, miatt .
* vagy . Vegyünk egy k élű vágást G-ben.
  + Ha a k élt elhagyva u és v különböző komponensben vannak, akkor , azaz . Korábban beláttuk, hogy , tehát .
  + Ha ugyanabban a komponensben vannak, akkor is szétesik ugyanannak a k élnek az elhagyásával (mert u és v összehúzása nem teheti összefüggővé), tehát , azaz . Korábban beláttuk, hogy , tehát .

## Max-vissza sorrend

**Def.:** A gráf pontjainak egy sorrendje max-vissza sorrend, ha esetén , ahol a *vk*-ból -ba menő élek száma.

Max-vissza sorrend készítése esetén az elsőként választott csúcs tetszőleges lehet.

## Mader tétel

Minden G gráfban létezik olyan csúcspár, hogy , ahol d(v) a v pont fokszáma.

Ha *u* és *v* a max-vissza sorrend szerinti utolsó két pont, akkor . Ha tehát minden iterációban az összehúzandó párt a max-vissza sorrend utolsó két pontjaként választjuk, akkor a egyszerűen d(*v)* fokszáma lesz. Így eltekinthetünk a folyam számolásától.

## Nagamochi és Ibaraki algoritmusa

Legalább kétpontú gráfra.

1. készítsük el a gráf max-vissza sorrendjét. Ha , akkor legyen . Ha még legalább három pontú a gráf, akkor húzzuk össze a max-vissza sorrend szerinti utolsó két pontot, GOTO 2. Egyébként STOP.

Nem ismert hasonló eljárás a pontösszefüggőség kiszámítására és irányított gráfokra sem.

22. tétel – Megbízható hálózatok terv.

Minimális méretű 2-élösszefüggő, illetve 2-összefüggő részgráfok keresése. A problémák NP-nehézsége, Khuller-Vishkin (bizonyítás nélkül, de éles példával) és Cheryan-Thurimella algoritmusok (biz. nélkül).

## Minimális költségű többszörösen összefüggő részgráfok

Az általános optimalizálási feladat a következő: legyen adott egy G=(V,E) (irányított) gráf, élein egy  
 költségfüggvénnyel, valamint egy követelményfüggvény a pontpárokon. Keressük G-nek olyan feszítőgráfját, melyben minden pontpárra   
 (illetve) teljesül, és erre nézve minimális.

Ez a feladat NP-nehéz. Például az és eset általánosítása a Hamilton-kör problémának. Ha a pontok egy részhalmazának pontpárjain a követelmény azonosan egy, egyébként pedig nulla, a Steiner-fa problémához jutunk, amely szintén NP-nehéz. Vannak jól közelíthető esetek is. Ha minden párra, akkor a feladat egy minimális költségű feszítőfa keresése, melyre ismert hatékony (mohó) algoritmus (Kruskal-algoritmus).

## Minimális méretű 2-élösszefüggő részgráf keresése

Adott egy *G* 2-élösszefüggő irányítatlan gráf, minden él költsége 1, és (élösszefüggési követelmény; ez a Hamilton-kör általánosítása: „dupla Hamilton-kör”). Feladat egy minimális élszámú 2-összefüggő feszítőgráf keresése *G*-ben. Látható, hogy az optimum értéke |*V*|, ha G-ben van Hamilton-kör. A feladat NP-nehéz, ezért nincs hatékony algoritmus, de közelítés van.

## Khuller-Vishkin

Az algoritmus 2-élösszefüggő irányítatlan gráfban 2-élösszefüggő feszítőgráfot keres.

Hajtsunk végre egy mélységi keresést (DFS) a gráfban egy tetszőleges pontból kiindulva, és közben jelöljük ki azt az E’ élhalmazt, amelyet a megoldásunk tartalmazni fog. A keresés során minden olyan él, mely a felépítendő T mélységi fa éle lesz, bekerül E’-be. Ezenkívül minden olyan pillanatban, amikor a keresés egy v pontból visszalép a T valamely uv éle mentén (ekkor a T-nek a v pontnál gyökerező T(v) részgráfját már teljesen bejártuk), ellenőrizzük, hogy az uv él elvágó él-e az eddig kijelölt E’ élhalmaz által feszített gráfban. Ha igen, adjunk egy olyan T(v)-ből – tehát nem csak v-ből, hanem a v-ből kiinduló részgráfjából – kilépő élt az E’-höz, mely nincs T-ben, és a T(v)-n kívüli végpontját a keresés legelőször érte le (azaz a mélységi száma a legkisebb). A mélységi keresés végén kapott E’ élhalmaz által feszített részgráf legyen az output.

Könnyű igazolni, hogy az algoritmus által adott E’ élhalmaz 2-élösszefüggő gráfot feszít a V ponthalmazon. Valamivel bonyolultabb annak igazolása, hogy az optimum (jelölje ω) legalább kétszer akkora, mint a végső E’ élhalmaz T-hez nem tartozó éleinek száma. Ezt felhasználva, és megfigyelve, hogy az optimum legalább |V|, hiszen minden megengedett megoldásban minden pont foka legalább kettő, a következő becsléshez jutunk:

Az algoritmus tehát -approximációs.

### Példa 1.

Az éles példa azt jelenti, hogy olyan példa, amikor az approximációs algoritmus a lehető legrosszabb becslést adja (itt pl. az optimum 3/2-szeresét), ami mutatja, hogy a becslés "éles", tehát nem javítható.

Pl. egy olyan gráf, amikor egy teljes páros gráfot pakolunk össze úgy, hogy az egyik komponens *n* darab izolált pont, a másik pedig egy *n* csúcsú teljes gráf. Mondjuk egy *K*4,4 és ennek az egyik osztályába felveszünk még 6 élet.

kapott megoldás

optimális megoldás

vizsgálandó gráf

Az optimum ilyenkor a Hamilton-kör hossza, tehát 2\**n* (*K*4,4-nál 8 él). Az algoritmus pedig úgy lesz "rossz", hogy a mélységi kereséssel először bejárjuk az egyik oldalon lévő teljes részgráfot, ez *n*-1 él, és utána minden, a másik oldalon lévő ponthoz fel kell venni 2 élet, tehát 2*n*-et. Ez összesen 2*n*+*n*-1 él (itt: 11).

Összesítve, az approximációs faktor: , így nem lehet jobb, mint 3/2.

### Példa 2.



A csúcsok sorszámai a mélységi bejárás során kapott mélységi számok. A vastag fekete élek jelölik a mélységi bejárás útját, a vékony piros élek a visszalépéskor hozzáadott élek (a feliratuk a behúzás sorszáma, szögletes zárójelben a csúcs, amelyből történő visszalépéskor behúztuk), a szaggatott élek az -beli élek.

## Minimális méretű 2-összefüggő gráf

Egy 2-összefüggő gráf egy él vagy egy pont törlése esetén összefüggő marad.

## Cheryan-Thurimella

Az algoritmus 2-összefüggő irányítatlan gráfban 2-összefüggő feszítőgráfot keres.

1. Minimális *F* lefogó élhalmazt keresünk (például úgy, hogy maximális párosítást keresünk, utána hozzáveszünk olyan éleket, hogy a párosításból kimaradt pontok is bekerüljenek)
2. Hagyjuk el a gráfból az *F*-hez nem tartozó éleket addig, amíg a gráf 2-összefüggő marad. (A kérdéses él akkor hagyható el, ha végpontjai között van 3 pontdiszjunkt út. Ezt egy maximális folyam algoritmussal dönthetjük el.) A megmaradt élek a kimenet (-approximáció mellett)

### Példa 1.



A vastag fekete élek a minimális lefogó élhalmaz élei, a szaggatott vonallal jelzetteket törültük, a vékony piros élek maradtak meg az halmazból. A zárójelben lévő számok az egyes élek vizsgálatának sorrendjét jelölik.

Az ábrán , -ben 13 él marad, a kapott feszítő részgráf összesen 23 élből áll.