1. tétel

Az optimális hozzárendelés problémája, Egerváry algoritmusa.

## Magyar módszer

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

Helyesség bizonyítása: legyen a két pontosztály A és B, az algoritmus leállásakor talált párosítás M. Legyen a M által nem lefedett A-beli pontok halmaza U. Az U-ból alternáló úton elérhető B-beli pontok halmaza legyen T’. T’ A-beli szomszédai (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: .

## Címkézés definíció

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

## Lemma

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 *U* az *M’* által le nem fedett *A*-beli pontok halmaza, *T’* a *B* azon része, melyek az *U*-ból alternáló úton elérhetők és pirosak (teljesül rá a ). *T* az eddigi párosítás szerint *T’* párja.

Ekkor , majd az új címkézés:

Folytatjuk 2-től (*M*’ és *c*’-vel).



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.

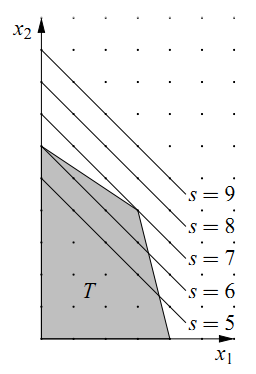
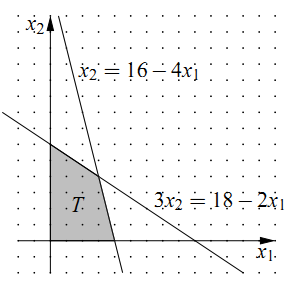
*A* *m*×*n*-es mátrix, *x* egy *n* dimenziós, *b* egy *m* dimenziós, *c* egy *n* dimenziós vektor. A LP alapfeladata ekkor:

Megjegyzések:

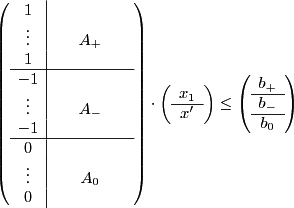
* helyett
* helyett és
* helyett (aztán ellentett-vétel a végleges megoldáshoz)

## 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

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 (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

Kettőnek egyszerre nincs megoldása:

Ha (2) nem megoldható, akkor (1) igen. (1)-et átírjuk egyenlőtlenségrendszerre, amire alkalmazható a Farkas lemma 1.

## Célfüggvény korlátosság

megoldható, ekkor az állítások ekvivalensek:

1. Az megoldáshalmazás *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).

## 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 korlátosságnál megvolt, (2) és (3) következik ebből az állításból:

*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) primálra:** indirekt módszer. 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) duálisra:** egyszerűen átírjuk: alakra.

**(3)** fennáll. Indirekt: 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

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.

## 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 2. (Bound lépés.)
4. lépés: IP*(i)* helyett LP*(i)* relaxált feladat megoldása. Ha nincs megoldás, akkor GOTO 2. Egyébként maximum *z(i)*, maximumhely *x(i)*.
5. lépés
   1. Ha , akkor GOTO 2
   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 2. (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.
* +lágazásnál 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.

## 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

### Lemma bizonyítása

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

1. (1): a determináns (–1)-szeres lesz
2. (2): a determináns megegyezik vagy (–1)-szeres lesz az egység pozíciójától függően.
3. (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

## 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 (azaz egységvektor, vagy annak ellentettje):  
    Ha ezt az oszlopot nem vesszük figyelembe, akkor az indukciós lépés miatt a maradék mátrix TU (mert részmátrixai TU-k, hiszen k-nál kisebbek). Ha ehhez figyelembevesszük ezt az egy egységvektort (lemma (2)) vagy ellentettjét (lemma (1)), akkor továbbra is TU marad, vagyis a teljes mátrix is TU.
  + 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. A *B*-hez tartozó sorokat negáljuk, de ez nem változtat TU tulajdonságon (lemma (1)).

## 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 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 IP feladatként (vázlat)

* Intervallumgráf és az ebből képzett mátrix definíciója
* Állítás: az ilyen mátrix TU
* 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
* 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.

## A tétel szerkezete

* Hálózat és folyam alapfogalmak
* Ford-Fulkerson említése
* Segítő lemma
* Felírás LP feladatként
* Felírás duálisként => új Ford-Fulkerson bizonyítás, ami nem-racionális élkapacitásokra is jó!
* Alkalmazás min. ktg-ű folyamokra
* Alkalmazás többtermékes folyamokra

## Jelölések

irányított gráf

* két kitüntetett csúcs
* nemnegatív *kapacitásfüggvény*. tetszőleges függvény.
* *ρx(v)*: *v*-be belépő élek összege *x* szerint
* *δx(v)*: *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 .

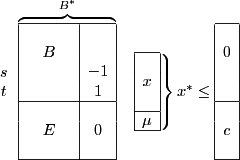
## Lemma

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

### Bizonyítás

Minden élből 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 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

A . Írjuk fel másképp *y*-t: . Ekkor:

1. ,
2. minden él esetén

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, mert *w’* definíciója miatt .

Optimális, mert , mert .

Legyen és a fentiek szerinti *S* és *T*. Minden élre (2) miatt . Minden más élen *w’(e)* csak akkor lehet 1, ha , mert -ra változtatása után a feltételek továbbra is fennállnának, 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

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)*.

## 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.

## Rangfüggvény szubmodularitás

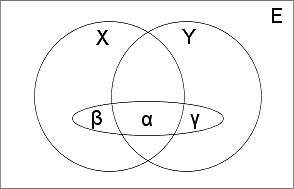
Legyen *r* egy matroid rangfüggvénye. Ekkor minden halmazpárra:

1. .
2. , ha

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

### 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.

## 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: , ahol .

## Összehúzás

matroid, összehúzása: , 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.

## T test felett reprezentálható

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

### Definíció2

Az matroid reprezentálható *T* test felett, ha létezik olyan mátrix, amelynek oszlopai *T* feletti vektorok, és ezek által meghatározott lineáris matroid izomorf *M*-mel.

## Duális reprezentációja T test felett

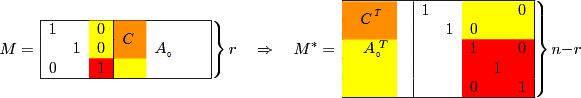
Ha matroid reprezentálható *T* test felett, akkor *M\** is.

### Bizonyítás

Induljunk ki a reprezentálás definíciójából: az alaphalmaz elemszámát jelöljük , *M* rangját *r*-rel, *M*-mel izomorf lineáris matroidot 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.



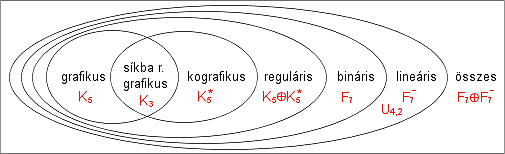
11. tétel

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).

## 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.



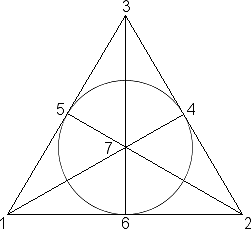
K3,3\*

## 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.

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?

2-matroid metszet probléma: két matroid esetén,

## 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.

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|.

### 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 jelképezi, 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 a 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.

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

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

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

## k-polimatroid matching probléma

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

**Def.**: k-polimatroid matching probléma: adott f 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: f(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

**Lovász-tétel**: a 2-polimatroid matching probléma polimatroid rangfüggvény orákulummal általános esetben exponenciális.

**Biz.**: |E|=2n, X⊆E. Tekintsük az alábbi polimatroid rangfüggvényt:

D:\Letöltések\latex20fb65f47925d6834ef0f77f15347faa.pngHa egy n elemű részhalmazra kérdeznek rá, 2n-1-et mondok. Polinom sok kérdéssel nem ellenőrizhető, hogy fogok-e 2n-et mondani.

**Lovász-tétel**: Tegyük fel, hogy E (ai,bi) párokból áll és az f függvényt a következőképpen definiáljuk az {1,2,...,n} indexhalmazon: ∀J⊆I-re f(J)=r(∪i∈J{ai,bi})   
Ha ∀i r({ai,bi})=2 (nincs hurokél és ai,bi párhuzamos élek), f egy 2-polimatroid rangfüggvény. Ha emellett M koordinátázva is van **R** felett, a 2-polimatroid matching probléma polinom időben megoldható.

15. tétel

NP-nehéz feladatok polinomiális speciális esetei: példa. Additív hibával közelítő algoritmus fogalma, példák. Példa additívan nem közelíthető problémára. k-approximációs algoritmusok, példák: minimális lefogó ponthalmaz, maximális páros részgráf keresése.

## NP-nehéz feladatok

|  |  |
| --- | --- |
| **NP-nehéz általában** | **de P-beli, ha** |
| a maximális élszámú (s,t)-vágás keresése | a hálózat síkba rajzolható |
| meghatározása | a G gráf páros |
| a leghosszabb irányított út keresése | a gráfban nincs irányított kör |

Konkrét példa: leghosszabb irányított út keresése NP-nehéz, de ha nincs irányított kör (azaz gráfunk egy DAG), akkor szélességi kereséssel meghatározható.

## 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 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.

Ugyanakkor a leghosszabb kör keresése egy gráfban nem közelíthető additívan (kivéve, ha P = NP). Ugyanis ha algoritmusunk C additív hibával megmondja a leghosszabb kör hosszát, akkor egy G gráf minden élére C új pontot felvéve egy olyan G’ gráfot kapunk, melyben a leghosszabb kör hossza C+1-szerese az eredeti G gráf leghosszabb körének hosszánál. Erre lefuttatva algoritmusunkat és kerekítve a legközelebbi C+1-gyel osztható számra, pontosan meg tudnánk mondani G’-ben (és így G-ben) a leghosszabb Hamilton-kör hosszát. Ez pedig NP teljes probléma.

## 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.

## 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 közelítése

*A feladat:* Adott U alaphalmaz, S1⊆U, ..., Sn⊆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 c(Si)/|Si-C| minimális.

*Lemma:* legyen p(e)=c(Si)/|Si-C|, 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. p(ek)≤OPT/(m-k+1).

*Bizonyítás:* ∑p(e) = fedés összköltsége. Mivel az algoritmus mindig a minimális c(Si)/|Si-C| értékű halmazt választja,

p(ek) = c(Si)/|Si-C| ≤ OPT/|C| = OPT/(m-k+1)

*Következmény:* a fedés költsége ≤ OPT/1+OPT/2+...+OPT/m < OPT\*(ln(m) + 1).

## 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

Legyen *G*’ a teljes gráf *V* ponthalmazon. Egy *G*’-beli {*u*, *v*} él költsége a *G*-beli legkisebb összköltségű *u* – *v* út hossza. 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 .

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.

## 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 él, 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.

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

**Christofides algoritmus**: vesszük a minimális feszítőfát és egy minimális összsúlyú teljes párosítást a fa páratlan fokú csúcsai által kifeszített G' részgráfban. A fa és a párosítás élei által meghatározott részgráfban 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

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 a 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 algoritmuste:

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. (Kép hiányzik. ☹)

**Biz.**: az algoritmus lépésszáma mindhárom mennyiség szerint polinomiális.

D:\Letöltések\latex31c610a20d43e3134b429d830ee6143b.png

Másrészt az x≥-1-re érvényes x/(1+x) ≤ ln(1+x) összefüggést kihasználva x=ε/2n helyettesítéssel:

D:\Letöltések\latex83f6e5bad01cb4b94ec30f80de0bc0e3.png

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. Az 1/ε szerinti becslést órán lenyeltük.

19. tétel

Ütemezési feladatok típusai. Az 1|prec|Cmax és az 1||∑Cj feladat. Graham közelítő algoritmusai a P||Cmax és a P|prec|Cmax feladatokra (utóbbi biz. nélkül). A P|prec, pj = 1|Cmax feladat. Példa arra, hogy a leghosszabb út szerinti sorrend nem feltétlenül optimális, Hu algoritmusa (biz. nélkül).

## 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, 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|prec|Cmax

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

## 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 nem dolgozik, a listában következő munkát azonnal odaadjuk neki.

**Tétel**: a listás ütemezés (2-1/m)-approximálja a Pm||Cmax feladatot.

**Biz.**:

* OPT ≥ max(pi)
* OPT ≥ ∑pi/m
* Az utolsóként befejeződő k munka t-kor kezdődött, és t-ig egy gép sem állhatott ⇒ ∑i≠kpi ≥ mt
* Cmax = t+pk ≤ ∑i≠kpi/m+pk = ∑pi/m+pk(1-1/m) ≤ OPT\*(2-1/m)

**Tétel**: Az LPT (Longest Processing Time) szerinti listás ütemezés 4/3-approximációs minden m-re.

## P|prec|Cmax (Graham)

**Tétel**: a listás ütemezés (ha van szabad gép, a sorban az első olyan jobot teszem fel rá, ami már elkezdhető) (2-1/m)-approximációs a P|prec|Cmax feladatra is.

## P|prec, pj=1|Cmax

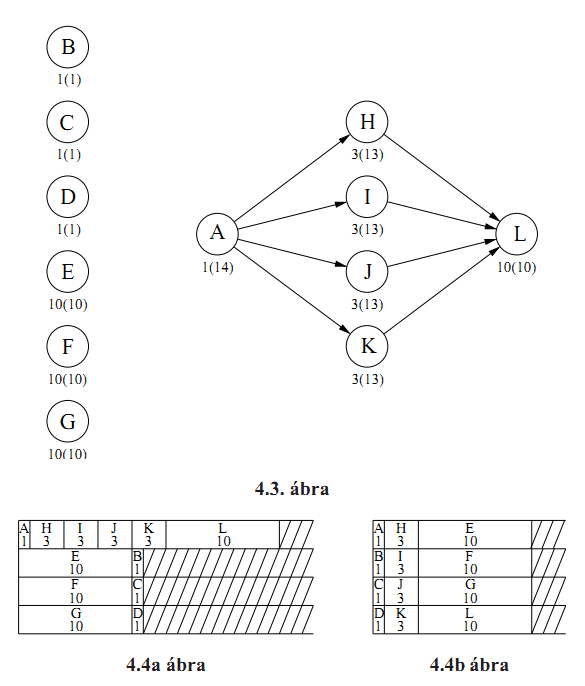
A feladatosztály bonyolultsága

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

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

## 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.



## 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.

20. tétel

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

## Lokális élösszefüggőség

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

Menger-tétel:

Vagyis az *u*, *v* pontokat elválasztó ponthalmaz minimális fokszáma.

## Gráf élösszefüggőségi száma

Órai definíció szerint: *λ*(*G*)-1 tetszőleges él törlésekor még biztosan összefüggő marad a gráf, de *λ*(*G*) él már törölhető úgy, hogy a gráf szétessen.

A lokális élösszefüggőségi számok közül a legkisebb.

## λ(G) meghatározása folyamok segítségével

Ha a gráfot egy 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 keresésésre 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).

21. tétel

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

## Összefüggőség kiszámítása összehúzásokkal, Mader tétele

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.

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.

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

**Mader tétele** szerint: 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 *v* fokszáma lesz. Így eltekinthetünk a folyam számolásától.

## Nagamochi és Ibaraki algoritmusa

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.

22. tétel

Minimális méretű 2-élösszefüggő, illetve 2-összefüggő részgráfok keresése: Khuller-Vishkin (éles példával) és Cheryan-Thurimella algoritmusok (biz. nélkül).

## 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

Mélységi bejárás (*T*), közben építjük a megoldást: *E*’. *T* éleit bevesszük *E*’-be. Továbbá ha az algoritmus *v* pontból visszalép egy teljesen bejárt részgráfból, és a visszalépésnél használt *uv* él elvágó, akkor a *T*(*v*)-ből azt a kilépő élet hozzáadjuk *E*’-höz, ami nincs *T*-ben és a *T*(*v*)-n kívüli pontját először érte el a keresés.

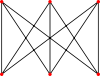
-approximációs.

### Példa

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 egy *K*3,3 és ennek az egyik osztályába felveszünk még 3 élet.

*K*3,3 kiegészítés nélkül:

[](http://hu.wikipedia.org/w/index.php?title=F%C3%A1jl:Complete_bipartite_graph_K3,3.svg&filetimestamp=20060307174215)

Az optimum ilyenkor a Hamilton-kör hossza, tehát 2\**n* (*K*3,3-nál 6 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.

Ez ugyan 1-gyel kevesebb, mint az optimum 3/2-e, de ez volt az órai példa, így valószínűleg elfogadható.

## Minimális méretű 2-összefüggő gráf

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

## Cheryan-Thurimella

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.) A megmaradt élek a kimenet (-approximáció mellett)