Tételek vázlatai

[I. Az optimális hozzárendelés problémája, Egerváry algoritmusa. 2](#_Toc483661981)

[Magyar módszer 2](#_Toc483661982)

[Algoritmus 2](#_Toc483661983)

[Alternáló út 2](#_Toc483661984)

[Javító út 2](#_Toc483661985)

[Javító út keresése 2](#_Toc483661986)

[Helyesség bizonyítás 2](#_Toc483661987)

[Optimális hozzárendelés 2](#_Toc483661988)

[Feladat 2](#_Toc483661989)

[Maximális összsúlyú teljes párosítás vagy maximális összsúlyú párosítás 2](#_Toc483661990)

[Maximális összsúlyú párosítás visszavezetése a maximális összsúlyú teljes párosításra 2](#_Toc483661991)

[Címkézés 2](#_Toc483661992)

[Definíció 2](#_Toc483661993)

[Címkézés és teljes párosítás súlyának kapcsolata 2](#_Toc483661994)

[Lemma 2](#_Toc483661995)

[Bizonyítás 2](#_Toc483661996)

[Éles címkézés és teljes párosítás súlyának kapcsolata 2](#_Toc483661997)

[Lemma 2](#_Toc483661998)

[Bizonyítás 2](#_Toc483661999)

[Egerváry algoritmus 2](#_Toc483662000)

[Előfeltétel 2](#_Toc483662001)

[Algoritmus 2](#_Toc483662002)

[Bizonyítás 2](#_Toc483662003)

[II. A lineáris programozás alapfeladata, kétváltozós feladat grafikus megoldása. Lineáris egyenlőtlenségrendszer megoldása Fourier-Motzkin eliminációval. 2](#_Toc483662004)

[Lineáris programozás 2](#_Toc483662005)

[Alapfeladat 2](#_Toc483662006)

[Formális definició 2](#_Toc483662007)

[Átalakítások 2](#_Toc483662008)

[Kérdések 2](#_Toc483662009)

[A kétváltozós feladat grafikus megoldása 2](#_Toc483662010)

[Fourier-Motzkin elimináció 2](#_Toc483662011)

[Algoritmus 2](#_Toc483662012)

[III. Farkas-lemma (két alakban). A lineáris program célfüggvénye felülről korlátosságának feltételei. 2](#_Toc483662013)

[Farkas lemma 1. 2](#_Toc483662014)

[Tétel 2](#_Toc483662015)

[Bizonyítás 2](#_Toc483662016)

[Farkas lemma 2. 2](#_Toc483662017)

[Tétel 2](#_Toc483662018)

[Bizonyítás 2](#_Toc483662019)

[Célfüggvény korlátosság 2](#_Toc483662020)

[3 kalitkás tétel 2](#_Toc483662021)

[Bizonyítás 2](#_Toc483662022)

[IV. A lineáris programozás dualitástétele (két alakban). A lineáris programozás alapfeladatának bonyolultsága (biz. nélkül). 2](#_Toc483662023)

[Dualitás 1 2](#_Toc483662024)

[Tétel 2](#_Toc483662025)

[Bizonyítás 2](#_Toc483662026)

[Dualitás 2 2](#_Toc483662027)

[Tétel 2](#_Toc483662028)

[LP bonyolultsága 2](#_Toc483662029)

[V. Egészértékű programozás: a feladat bonyolultsága, korlátozás és szétválasztás (Branch and Bound). Totálisan unimoduláris mátrix fogalma, példák. Egészértékű programozás totálisan unimoduláris együtthatómátrixszal (biz. nélkül). 2](#_Toc483662030)

[Egészértékű programozás 2](#_Toc483662031)

[Definiciók 2](#_Toc483662032)

[A feladat bonyolultsága 2](#_Toc483662033)

[Eldöntési probléma 2](#_Toc483662034)

[NP-beliség 2](#_Toc483662035)

[coNP-beliség 2](#_Toc483662036)

[NP-teljesség 2](#_Toc483662037)

[A Branch and Bound algoritmus 2](#_Toc483662038)

[Program 2](#_Toc483662039)

[Jelölések 2](#_Toc483662040)

[Az algoritmus lépései 2](#_Toc483662041)

[Az algoritmus hatékonysága 2](#_Toc483662042)

[Gyakorlati tapasztalatok az algoritmus használatára 2](#_Toc483662043)

[Totális unimoduláris (TU) mátrix 2](#_Toc483662044)

[Definició 2](#_Toc483662045)

[TU tartó műveletek 2](#_Toc483662046)

[TU tartás bizonyítása 2](#_Toc483662047)

[Egészértékű programozás totálisan unimoduláris együtthatómátrixszal 2](#_Toc483662048)

[Illeszkedési mátrix 2](#_Toc483662049)

[Irányított gráf illeszkedési mátrixa TU 2](#_Toc483662050)

[Páros gráf illeszkedési mátrixa TU 2](#_Toc483662051)

[VI. A lineáris és egészértékű programozás alkalmazása páros gráfokra és intervallumrendszerekre: Egerváry tétele, intervallumrendszerek egyenletes színezése. 2](#_Toc483662052)

[Maximális összsúlyú párosítás IP feladatként 2](#_Toc483662053)

[Program 2](#_Toc483662054)

[Duális (Egerváry Jenő tétele) 2](#_Toc483662055)

[Intervallumgráf 2](#_Toc483662056)

[Intervallumok reprezentációja mátixxal 2](#_Toc483662057)

[Intervallum metszési mátrixának TU tulajdonsága 2](#_Toc483662058)

[Intervallumgráfok egyenletes színezése *k* színnel 2](#_Toc483662059)

[Intervallumgráfok egyenletes színezése *k* színnel IP segítségével 2](#_Toc483662060)

[VII. A lineáris és egészértékű programozás alkalmazása hálózati folyamproblémákra: a maximális folyam, a minimális költségű folyam és a többtermékes folyam feladatai, ezek hatékony megoldhatósága a tört-, illetve egészértékű esetben 2](#_Toc483662061)

[Jelölések 2](#_Toc483662062)

[Maximális folyam, mint LP 2](#_Toc483662063)

[Program 2](#_Toc483662064)

[TU tulajdonság 2](#_Toc483662065)

[Minimális költségű folyam keresésére 2](#_Toc483662066)

[Feladat 2](#_Toc483662067)

[Program 2](#_Toc483662068)

[TU tulajdonság 2](#_Toc483662069)

[Alkalmazás többtermékes folyamra 2](#_Toc483662070)

[Feladat 2](#_Toc483662071)

[Program 2](#_Toc483662072)

[TU tulajdonság 2](#_Toc483662073)

[VIII. Matroid definíciója, alapfogalmak (bázis, rang, kör). Példák: lineáris matroid (mátrixmatroid), grafikus matroid, uniform matroid. A rangfüggvény szubmodularitása. 2](#_Toc483662074)

[Matroid, mint függetlenségi axiómák 2](#_Toc483662075)

[Axiómák 2](#_Toc483662076)

[Alapfogalmak 2](#_Toc483662077)

[Matroid típusai 2](#_Toc483662078)

[Definicók 2](#_Toc483662079)

[Uniform matroidok grafikussága 2](#_Toc483662080)

[Matroid, mint rangfüggvény 2](#_Toc483662081)

[Axiómák 2](#_Toc483662082)

[Kapcsolat a függetlenségi axiómákkal 2](#_Toc483662083)

[IX. Mohó algoritmus matroidon. Matroid megadása rangfüggvényével, bázisaival (biz. nélkül). Matroid duálisa, a duális matroid rangfüggvénye. 2](#_Toc483662084)

[Mohó algoritmus matroidon 2](#_Toc483662085)

[Feladat 2](#_Toc483662086)

[Mohó algoritmus optimális matroidon 2](#_Toc483662087)

[Matroid, ha mohó algoritmus optimális 2](#_Toc483662088)

[Matroid, mint bázisok 2](#_Toc483662089)

[Axiómák 2](#_Toc483662090)

[Kapcsolat a függetlenségi axiómákkal 2](#_Toc483662091)

[Rangból matroid 2](#_Toc483662092)

[Matroid duálisa 2](#_Toc483662093)

[Definició 2](#_Toc483662094)

[Duális matroid tétel 2](#_Toc483662095)

[Síkbarajzolható gráfok duálisa 2](#_Toc483662096)

[Duális rangfüggvény 2](#_Toc483662097)

[X. Elhagyás és összehúzás. Matroidok direkt összege, összefüggősége. T test felett reprezentálható matroid duálisának T feletti reprezentálhatósága. 2](#_Toc483662098)

[Matroid minorok 2](#_Toc483662099)

[Definició 2](#_Toc483662100)

[Elhagyás 2](#_Toc483662101)

[Összehúzás 2](#_Toc483662102)

[Elhagyás és ősszehúzás kommutativítása 2](#_Toc483662103)

[Minorképzés és duális 2](#_Toc483662104)

[Minorképzés és matroidképzés 2](#_Toc483662105)

[Uniform mátrixok minorja 2](#_Toc483662106)

[Wagner tétel 2](#_Toc483662107)

[Matroidok direkt összege 2](#_Toc483662108)

[Definició 2](#_Toc483662109)

[Matroidok összefüggősége 2](#_Toc483662110)

[Direkt összeg koordinázhatósága 2](#_Toc483662111)

[Matroid koordniázhatósága 2](#_Toc483662112)

[Definició 2](#_Toc483662113)

[Bázis koordinázhatósága egységmátrixként 2](#_Toc483662114)

[Duális reprezentációja T test felett 2](#_Toc483662115)

[Bizonyítás 2](#_Toc483662116)

[XI. 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) 2](#_Toc483662117)

[Matroid osztályok definició 2](#_Toc483662118)

[Grafikus matroid 2](#_Toc483662119)

[Kografikus matroid 2](#_Toc483662120)

[Lineráis matroid 2](#_Toc483662121)

[Bináris matroid 2](#_Toc483662122)

[Regurális matroid 2](#_Toc483662123)

[Matroid osztályok kapcsolata 2](#_Toc483662124)

[Venn diagram 2](#_Toc483662125)

[Grafikus gráf reguláris 2](#_Toc483662126)

[Fano matroid 2](#_Toc483662127)

[Definició 2](#_Toc483662128)

[Anti-Fano 2](#_Toc483662129)

[Fano és Anti-Fano koordinázhatósága 2](#_Toc483662130)

[Tutte tételei 2](#_Toc483662131)

[Seymour tétel 2](#_Toc483662132)

[Test karakterisztákáka 2](#_Toc483662133)

[Fano és Anti-Fano koordinázhatóságának bizonyítása 2](#_Toc483662134)

[XII. Matroidok összege. k-matroid-metszet probléma, ennek bonyolultsága k ≥ 3 esetén. 2](#_Toc483662135)

[Matroid összege 2](#_Toc483662136)

[Definició 2](#_Toc483662137)

[Matroidok összege is matroid 2](#_Toc483662138)

[k-matroid metszet probléma (k-MMP) 2](#_Toc483662139)

[Feladat 2](#_Toc483662140)

[2-MMP bonyolultsága 2](#_Toc483662141)

[3-MMPk bonyolultsága 2](#_Toc483662142)

[k-MMP bonyolultsága 2](#_Toc483662143)

[XIII. A k-matroid partíciós probléma, ennek algoritmikus megoldása. A 2-matroid-metszet feladat visszavezetése matroid partíciós problémára. 2](#_Toc483662144)

[k-matroid partíciós probléma (k-MPP) 2](#_Toc483662145)

[Feladat 2](#_Toc483662146)

[Bonyolultság 2](#_Toc483662147)

[2-MMP visszavezetése 2-MPP-re 2](#_Toc483662148)

[2-MMP 2](#_Toc483662149)

[2-MMP 2](#_Toc483662150)

[Redukció 2](#_Toc483662151)

[XIV. k-polimatroid rangfüggvény fogalma. A 2-polimatroid-matching probléma, ennek bonyolultsága, Lovász tétele (biz. nélkül). 2](#_Toc483662152)

[k-polimatroid rangfüggvény 2](#_Toc483662153)

[Definició 2](#_Toc483662154)

[k-polimatroid matching probléma (k-PMMP) 2](#_Toc483662155)

[k-matching 2](#_Toc483662156)

[Feladat 2](#_Toc483662157)

[Speciális esetek 2](#_Toc483662158)

[Bonyolultság 2](#_Toc483662159)

[Matroidpárosítási probléma (2-PMMP) 2](#_Toc483662160)

[Bonyolultság 2](#_Toc483662161)

[Bizonyítás 2](#_Toc483662162)

[Lovász-tétel 2](#_Toc483662163)

[XV. Polinomiális időben megoldható feladat fogalma, példák. Az NP, co-NP, NP-nehéz és NP-teljes problémaosztályok definíciója, viszonyaik, példák problémákra valamennyi osztályból. NP-nehéz feladatok polinomiális speciális esetei: intervallumgráfok színezése, algoritmus a maximális független ponthalmaz problémára és az élszínezési problémára páros gráfokon. 2](#_Toc483662164)

[Eldöntési probléma osztályok 2](#_Toc483662165)

[Definiciók 2](#_Toc483662166)

[Viszonyaik 2](#_Toc483662167)

[Híres eldöntési problémák 2](#_Toc483662168)

[Független és lefogó halmazok 2](#_Toc483662169)

[Jelölés 2](#_Toc483662170)

[Viszony 2](#_Toc483662171)

[NP-nehéz feladatok polinomiális speciális esetei 2](#_Toc483662172)

[Intervallumgráfok színezése 2](#_Toc483662173)

[Maximális független ponthalmaz páros gráfokon 2](#_Toc483662174)

[Élszínezés páros gráfokon 2](#_Toc483662175)

[XVI. Additív hibával közelítő algoritmus fogalma, példa pont-, illetve élszínezési problémákra .A Hamilton-kör probléma visszavezetése a leghosszabb kör probléma additív közelítésére. k-approximációs algoritmus fogalma, példák: két-két algoritmus a minimális lefogó ponthal-maz keresésére és a maximális páros részgráf keresésére. 2](#_Toc483662176)

[Additív hiba 2](#_Toc483662177)

[Definició 2](#_Toc483662178)

[C-Additív közelítő algoritmus 2](#_Toc483662179)

[Élszínezés 1-additív hibával 2](#_Toc483662180)

[Csúcsszínezés 1-additív hibával síkba rajzolható gráfokon 2](#_Toc483662181)

[Hamilton-kör visszavezetése leghosszabb kör probléma additív közelítésére 2](#_Toc483662182)

[Multiplikatív hiba 2](#_Toc483662183)

[Definició 2](#_Toc483662184)

[k-approximációs algoritmus 2](#_Toc483662185)

[Maximális páros részgráf 2-approximációja 2](#_Toc483662186)

[Maximális páros részgráf online 2-approximációja 2](#_Toc483662187)

[Minimális lefogó ponthalmaz 2-approximációja 2](#_Toc483662188)

[Minimális lefogó ponthalmaz 2-approximációja mohó algoritmussal 2](#_Toc483662189)

[XVII. A minimális lefogó ponthalmaz probléma visszavezetése a halmazfedési feladatra, a halmazfedési feladat közelítése, éles példa. Közelítő algoritmus a Steiner-fa problémára, éles példa. 2](#_Toc483662190)

[Súlyozott halmazfedés 2](#_Toc483662191)

[Feladat 2](#_Toc483662192)

[Minimális lefogó ponthalmaz probléma visszavezetése a halmazfedési feladatra 2](#_Toc483662193)

[A halmazfedés közelítése 2](#_Toc483662194)

[Közelítésének faktora 2](#_Toc483662195)

[Közelítés éles példája 2](#_Toc483662196)

[Steiner-fa 2](#_Toc483662197)

[Feladat 2](#_Toc483662198)

[Bonyolultság 2](#_Toc483662199)

[Metrikus gráf 2](#_Toc483662200)

[2-approximáció metrikus gráfon 2](#_Toc483662201)

[Éles példa 2-approximációra metrikus gráfon 2](#_Toc483662202)

[Általános Steiner-fa visszavezetése metrikus Steiner-fára 2](#_Toc483662203)

[XVIII. A Hamilton-kör probléma visszavezetése az általános utazóügynök probléma k-approximációs megoldására. Közelítő algoritmusok a metrikus utazóügynök problémára,Christofides algoritmusa. 2](#_Toc483662204)

[Általános utazó ügynök feladat 2](#_Toc483662205)

[Feladat 2](#_Toc483662206)

[Utazó ügynök approximáxiója 2](#_Toc483662207)

[Hamilton-kör visszavezetése utazó ügynök k-approximációjára 2](#_Toc483662208)

[Metrikus utazóügynök probléma 2](#_Toc483662209)

[Feladat 2](#_Toc483662210)

[2-approximációs algoritmus 2](#_Toc483662211)

[XIX. Teljesen polinomiális approximációs séma fogalma. A részösszeg probléma, bonyolultsága. Teljesen polinomiális approximációs séma a részösszeg problémára. 2](#_Toc483662212)

[Polinomiális approximációs séma 2](#_Toc483662213)

[Definició 2](#_Toc483662214)

[Teljesen polinomiális approximációs 2](#_Toc483662215)

[Részösszeg probléma 2](#_Toc483662216)

[Feladat 2](#_Toc483662217)

[Optimalizálási feladat 2](#_Toc483662218)

[Megoldása 2](#_Toc483662219)

[Közelítő algoritmusa 2](#_Toc483662220)

[Teljesen polinomiális approximációs sémája 2](#_Toc483662221)

[XX. Ütemezési feladatok típusai. Az 1|prec|Cmax és az 1||ΣCj feladat. Approximációs algoritmusok a P|| Cmax feladatra: listás ütemezés tetszőleges sorrendben, éles példa tetszőleges számú gép esetére; listás ütemezés LPT sorrendben (biz. nélkül), éles példa tetszőleges számú gép esetére. Approximációs algoritmus a P|prec| Cmax feladatra (biz. nélkül), példák: az LPT sorrend, illetve a leghosszabb út szerinti ütemezés sem jobb, mint (2−1/m)-approximáció. A P|prec,pi = 1| Cmax feladat, Hu algoritmusa (biz. nélkül). 2](#_Toc483662222)

[Ütemezési feladatok alapjai 2](#_Toc483662223)

[Ütemezési feladatok osztályozása 2](#_Toc483662224)

[1| |Cmax 2](#_Toc483662225)

[1|prec|Cmax 2](#_Toc483662226)

[1||∑Cj 2](#_Toc483662227)

[SPT (Shortest Processing Time) 2](#_Toc483662228)

[P||Cmax 2](#_Toc483662229)

[Bonyolultság 2](#_Toc483662230)

[LS listás ütemezés (2-1/m) 2](#_Toc483662231)

[LS-LPT (Longest Processing Time) 2](#_Toc483662232)

[Probléma a listás ütemezéssel 2](#_Toc483662233)

[P|prec|Cmax 2](#_Toc483662234)

[2-1/m approximáció 2](#_Toc483662235)

[LPT éles példa 2](#_Toc483662236)

[Leghosszabb út szerinti sorrend éles példa 2](#_Toc483662237)

[P|prec, pj=1|Cmax 2](#_Toc483662238)

[Bonyolultság 2](#_Toc483662239)

[Hu algoritmusa 2](#_Toc483662240)

# I. Az optimális hozzárendelés problémája, Egerváry algoritmusa.

## Magyar módszer

### Algoritmus

1. Adott páros gráf és egy párosítás (üres is jó)
2. Keres javító utat
3. Ha nincs, a jelenlegi párosítás maximális
4. Javító út mentén az élek szerepét felcseréjük (párosított <> párosítatlan)
5. Go to 2

### Alternáló út

Párosítatlan csúcsból induló út amiben párosított és párosítatlan élek felváltva szerepelnek.

### Javító út

Párosítatlan csúcsba vezető alternáló út.

### Javító út keresése

Módosított BFS algoritmus:

* Csak párosítatlan csúcsból indítható.
* Párosított és párosítatlan éleket felváltva választ.

### Helyesség bizonyítás

Ha leállt az algoritmus az alábbi helyzet alakul ki:

|  |  |
| --- | --- |
| A, B | Páros gráf csúcsosztályai |
| M | A jelenlegi párosítás |
| U | A\M (A csúcsok közül az M által fedetlenek) |
| T’ | U-ból alternáló úton elérhető B csúcsok |
| T | T’ párosításai |



T’ u (A \ (T u U)) lefogó csúcshalmaz, hisz:

* (T u U)-ból induló élek T’-be kell, hogy menjenek (különben lenne javító út).
* A csúcshalmaz többi csúcsa pedig a lefogó csúcshalmaz része.

|T’ u (A \ (T u U))| = |M|, hisz:

Minden párosításbeli élnek pontosan egy csúcsát tartalmazza.

König tétele szerint páros gráfban a maximális párosítás mérete megegyezik a minimális lefogó csúcshalmazéval. Mivel találtunk |M| méretű lefogó csúcshalmazt, ezért M maximális.

## Optimális hozzárendelés

### Feladat

Adott páros gráf és élsúly-függvény.

Keressük maximális összsúlyú *M* teljes párosítást: .

### Maximális összsúlyú teljes párosítás vagy maximális összsúlyú párosítás

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.

### Maximális összsúlyú párosítás visszavezetése a maximális összsúlyú teljes párosításra

* Az egyik csúcshalmazt kibővítjük, hogy a kettő elemszáma megegyezen
* A hiányzó éleket 0 súllyal felvesszük
* Negatív súlyú éleket 0 súlyúvá írjuk át

**Visszavezetés helyességének bizonyítása**

Legyen M1’ a maximális teljes párosítás a transzformált gráfon.

Legyen M1 az eredeti gráfon egy párosítás, amit a 0 súlyú élek lehagyásával kaptunk. (Ez nyilván súlyban megegyezik M1’-vel és csak az eredeti gráf éleit tartalmazza, hisz minden felvett vagy módosított él 0 súlyú, a párosítás tulajdonságát se veszti el, hisz csak éleket hagyunk el.)

Indirekt: tfh létezik M2, amely súlyban nagyobb M1.nél.

Ekkor viszont a transzformált gráfon létezik egy M2’ teljes párosítás, amely nagyobb egyenlő súlyú, mint M2 (negatív élek helyett 0 súlyú élt választunk és a fedetlen csúcsokat 0 élekkel veszük be a párosításba).

w(M2’) >= w(M2) > w(M1) = w(M1’), de M1’ maximális volt, tehát ellentmondásba ütköztunk, nem létezhet M2 párosítás.

## Címkézés

### Definíció

páros gráf.

címkézés (csúcs súlyfüggény), ha minden élre .

## Címkézés és teljes párosítás súlyának kapcsolata

### Lemma

Ha w súly függvényhez képest c címkézés és M egy párosítás.

### Bizonyítás

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

\* A címkézési tulajdonság miatt.

\*\* Mivel M teljes, ezért minden csúcsot tartalmaz pontosan egyszer.

## Éles címkézés és teljes párosítás súlyának kapcsolata

### Lemma

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

### Bizonyítás

\* Az éles címkézési tulajdonság miatt.

\*\* Mivel M teljes, ezért minden csúcsot tartalmaz pontosan egyszer.

## Egerváry algoritmus

### Előfeltétel

A gráfokat kiegészítjük úgy, hogy létezzen bennük teljes párosítás.

(Nem feltétlenül kell, hogy teljes gráf legyen.)

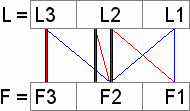
### Algoritmus

Egy (F,L;E) práos gráfon számon tart egy c címkézést és egy M párosítást, a kettőt addig javítja míg a párosításon éles lesz a címkézés, ezzel garantálva, hogy maximális élsúlyú.

1. ;

1. M = Magyar módszer M-ből kiinudlva a PIROS részgráfon
2. M teljes párosítás eredeti gráfon => M maximális
3. Ha M még nem TP, akkor élesítjük c címkézést

|  |  |
| --- | --- |
| F1 | M álal nem fedett |
| L2 | F1-ből piros alternáló úton elérhető |
| F2 | L2 párja M-ben |
| L1 | M általá nem fedett |
| L3 | A maradék |
| F3 | A maradék |



(Minden piros él F1 U F2 és L2 között megy, szeretnénk új éleket bevenni, hogy bővülhessen a párosítás, ezért az F1 U F2 és L1 U L3 közöttiek közül a legpirosközelebbit bevesszük az F1 U F2 címkézés csökkentésével, de L2-ben pedig növeüljük, hogy ott se romoljon el a piros tulajodnság.)

1. Go to 1.

### Bizonyítás

Létezik-e delta:

Azaz létezik-e L1 u L3 és F1 u F2 között él, a válasz igenlő, hisz a Hall-feltétel miatt egyébbként TP se létezhetne.

Kell még, hogy delta értéke nem 0, hisz különben nem javulna a címkézés, de ez is igaz, hisz csak akkor lehetne 0, ha piros él mentén találtuk meg a minimumot, de F1 u F2 összes piros szomszédja L2-ben kell legyen, különben létezne javító út.

c címkézés marad:

változása:

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

A címke növekedése a címkézési tulajdonságot nem rontja el (nagyobb egyenlőség).

A L1 u L3 és F1 u F2 között pedig úgy választottuk meg -t, hogy minimális lesz, ezért sehol nem fog elromlani a címkézési tulajdonság, de legalább egy helyen élesülni fog.

Eltűnő piros élek:

Csak a L2 és F3 közötti élek esetén nő a címkézés, itt elromolhat a piros tulajdonság, ám ezek nem lehetnek részei az M párosításnak, illetve az L2-beli csúcsok továbbra is elérhetőek maradnak F1-ből piros alternáló úton.

Keletkezik egy új piros él F1 u F2 és L1 u L3 között:

F1-ből elérhető alternáló úton F2, ezért bármelyikből is indul az újonnan piros él a másik vége alternáló úton elérhető lesz, így L2 mérete nőni fog.

O(n) iteráció után (ha még nem nőtt) már minden L-beli csúcs elérhető lesz alternáló úton, azaz ha még M nem teljes és létezik teljes párosítás, akkor nőni fog a párosítás mérete.

O(n2) iteráció után n-szer nőtt a párosítás azaz teljesnek kell lennie.

Egy iteráció kereséséből áll, ami O(e).

Tehát az algoritmus O(n2e) lépés után előállít egy teljes párosítást, amin éles a címkézés, azaz maximális összsúlyú.

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

## Lineáris programozás

### Alapfeladat

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

### Formális definició

### Átalakítások

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

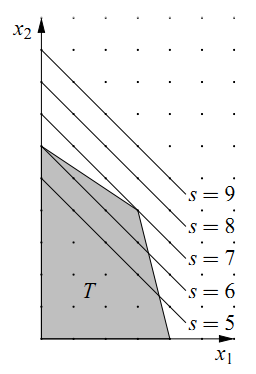
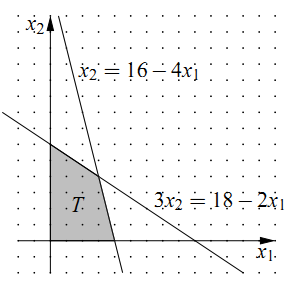
### Kérdések

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.
* A felsíkok metszetében léteznek a 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).

A célfüggvény meghatározza a meredekséget, egy egyenesen az azonos értékű megoldások lesznek. Azaz addig próbáljuk „felfelé” csúsztatni a célfüggvény egyenest, amíg már nem metszi a megoldások halmazát, itt találjuk meg az optimális megoldásokat.



## Fourier-Motzkin elimináció

### Algoritmus

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.

1. **Pozitív számmal való szorzással** olyan alakra hozzuk a mátrixot, hogy 1, 0 vagy -1 legyen az első oszlopban. (Oszlopokat lehet átrendezni, csak kövessük nyomon melyik változó oszlopáról volt szó.)
2. Legyen *A*+,-,0 azon sorok halmaza, ahol az első oszlop pozitív, negatív vagy nulla.
3. Ha A+,- üres, akkor a teljes sora elhagyható, az éppen aktuális változóra pedig az alábbi feltételek fognak vonatkozni (csak egy oldalról lesz korlátozva):

A+:

A-:

1. Ha |A+|>0 és |A-|>0, |A+|+|A-| sorból -> |A+||A-| sor lesz (minden +,- párra egy új sor)
2. Az első olszop csupa 0 lesz, ezért elhagyható
3. Ha még több, mint egy olszop (változó) van, Go to 2.

Úgy kell xn-t választani, hogy: esetén: és , azaz megoldható, ha (ha egyik nem létezik, akkor végtelen/-végtelen kerül helyére azaz megoldható)

A0 esetén ha van létezik negatív , akkor a rendszer nem megoldható, egyébbként minden x értékére teljesül az A0 sorai, ezért elhagyhatóak.

Megjegyzés: A Fourier-Motzkin elimináció futási ideje exponenciális.

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

## Farkas lemma 1.

### Tétel

Pontosan az egyiknek van megoldása:

### Bizonyítás

Kettőnek egyszerre nincs megoldása, mert: .

(1) nem megoldható => (2) igen

Vizsgáljuk a .

(A cél ekkor belátni, hogy )

C tulajdonságai, ha , azaz

Ez pont annak a definiciója, hogy , „tanuval”

Ez pont annak a definiciója, hogy „tanuval”



i-dik sor tanuja

* F-M minden lépése C-ben hagyja a mátrix sorait (természetesen az csupa 0 oszlopokat most nem hagyhatjuk el).

Ha nem megoldható, akkor (A|b)-n futtatva F-M algoritmust két lehetőség van:

Olyan sort kapunk, amelyre 0\*xn <= negatív => !

Kaptunk két sort: , tehát   
!

## Farkas lemma 2.

### Tétel

Pontosan az egyiknek van megoldása:

### Bizonyítás

Kettőnek egyszerre nincs megoldása:

(1) nem megoldható => (2) igen

(1)-et átírjuk egyenlőtlenségrendszerre:

Farkas lemma 1 szerint ennek megfelelő másik egyenlőtlenségek:

ahol

Legyen:

Tehát a Farkas lemma 1 értelmében, ha (1) nem megoldható, akkor létezik egy y’-t, ami megoldja (2)-t.

## Célfüggvény korlátosság

### 3 kalitkás tétel

megoldható, ekkor az állítások ekvivalensek:

1. Az megoldáshalmazon *cx* felülről korlátos. [És yb alulról korlátos]
2. Nincs megoldása az rendszernek
3. Van megoldása az rendszernek.

### Bizonyítás

Elég belátni, hogy (1)🡪(2)🡪(3)🡪(1)

(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 => Farkas lemma 2 transzponált:

xT=y,BT=A, ,bT=c és vT=-z behelyettesítéssel alkalmazható.

(3)🡪(1).

, minden *x, y*-ra *yb* felső korlát *cx* értékére.

\*(3) miatt

\*\*Eredeti feltevés miatt

\*\*\*Vegyük észre, hogy cx korlátossága egyben yb korlátosságát is jelenti.

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

## Dualitás 1

### Tétel

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:

*Segéd állítás:*

Tegyük fel, hogy megoldható és *t* tetszőleges.

Ha -nek nincs olyan megoldása, amelyre teljesülne, akkor -nak van megoldása, amelyre teljesül.

*Bizonyítás:*

Átírva:

Farkas lemma 1 szerint ha ez nem megoldható, akkor az alábbi egyenlőség rendszer viszont igen:

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.

## Dualitás 2

### Tétel

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

## 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

# V. Egészértékű programozás: a feladat bonyolultsága, korlátozás és szétválasztás (Branch and Bound). Totálisan unimoduláris mátrix fogalma, példák. Egészértékű programozás totálisan unimoduláris együtthatómátrixszal (biz. nélkül).

## Egészértékű programozás

### Definiciók

* IP alapfeladat:
* IP duálisa:

## A feladat bonyolultsága

### Eldöntési probléma

-t kiegyenlítő *x* egész megoldások közül van-e, amire ?

### NP-beliség

NP-beli: tanú

### coNP-beliség

nincs dualitástétel: nem adódik a co-NP-beli.

### NP-teljesség

Az NP-beliség triviális, tehét még azt kell belátni, hogy NP-nehéz, erre meg kell mutatni, hogy rajta keresztül megoldható egy NP-teljes probléma, például a k-Klikk (azaz Karp-redukálható az IP-re).

k-Klikk redukálható IP-re:

)

(Ha nincs él köztük, akkor csak egyik csúcs lehet benne)

## A Branch and Bound algoritmus

### Program

, *f* és *g* tetszőleges egész korlát vektorok.

### Jelölések

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

### Az algoritmus lépései

1. eredeti feladat, , nem definiált.
2. Ha üres, akkor STOP (megoldás *z\**, *x\** helyen). Egyébként vegyünk egy feladatot -ből (és töröljük onnan).
3. Ha : IP*(i)* nem lehet megoldás, GOTO 1. (Bound lépés.)
4. IP*(i)* helyett LP*(i)* relaxált feladat megoldása. Ha nincs megoldás, akkor GOTO 1. Egyébként maximum *z(i)*, maximumhely *x(i)*.
   1. Ha , skip

(A relaxált feladatnál nem lehet jobb, tehát már ismerünk az itt található lehetséges megoldásoknál jobbat.)

* 1. Ha , *x(i)* egész vektor. Ekkor z\* és x\* fölülírjuk ezeket az értékekkel.

(Ez esetben találtunk egy jobb megoldást.)

* 1. Ha , *x(i)* nem egész vektor.

(A relaxált feladat megoldása jobb, de nem egész, ezért ketté bontjuk és úgy próbálunk tovább vizsgálódni.)

Ekkor válasszunk *x(i)*-ben egy elágazási változót: xj, és egy közbülső értéket:

.

Defináljunk két új korlátot f’ és g’, ahol , másikban

(Azaz xj értékét egyik részfeladatban t-re alulról, másikban t+1-re felülről korlátozzuk)

Ez a Branching lépés, az eredeti feladat helyett két új feladatot defináltunk, egyiket ahol xj kisebb egyenlő t, és egyet ahol nagyobb egyenlő t+1-nél.

GOTO 1.

### Az algoritmus hatékonysága

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 . (csak f és g közti t értékek mentén tudunk elágazni.)

Optimális: indirekt

Legyen *z0* az optimum, de az eljárás csak -t találta.

-ben mindig van feladat, aminek az optimuma *z0*.

Ez a legelején (0. lépésnél) fönnáll.

Ezen optimális megoldás tartalmazó feladat esetén 4b vagy 4c lépéshez jut az algoritmus.

4b-nél meg is találja 🡪 ellentmondás.

4c esetén az egyik alfeladatban lesz benne az optimumhely (hisz egész értékű megoldás nincs t és t+1 között) 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.

## Totális unimoduláris (TU) mátrix

### Definició

Minden négyzetes részmátrixának determinánsa 0, 1 vagy –1.

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

### TU tartó műveletek

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

### TU tartás bizonyítása

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

1. A determináns (–1)-szeres lesz (pl adott sor szerint kifejtve könnyű belátni).
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.

### Egészértékű programozás totálisan unimoduláris együtthatómátrixszal

Legyen *A* totálisan unimoduláris mátrix, *b* egész koordinátájú vektor.

A (LP) feladat megoldható és korlátos.

Ekkor a (IP) feladat is megoldható, és maximuma megegyezik (LP) feladat maximumával, azaz az (IP) feladatnak létezik egész maximum helye.

, akkor ha c is egész vektor.

(A DIP-ben a „c” a b megfelelője.)

### Illeszkedési mátrix

Legyen az n-pontú G irányított 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.

### Irányított gráf illeszkedési mátrixa TU

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. (Ha létezik hurok él a gráfban.)
  + Egyébként minden oszlopban egy +1 és egy –1 elem van, ekkor *M* sorainak összege nullvektor, a determináns 0.

(Hiszen minden oszlop egy él, amihez egy kezdő és egy végpont tartozik pontosan, kivéve ha hurok él.)

### Páros gráf illeszkedési mátrixa TU

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 viszont visszakapjuk az eredeti irányítatlan gráf illeszkedési mátrixát.

# VI. A lineáris és egészértékű programozás alkalmazása páros gráfokra és intervallumrendszerekre: Egerváry tétele, intervallumrendszerek egyenletes színezése.

## 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).

### Program

.

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.

### Duális (Egerváry Jenő tétele)

.

Ekkor *y* megoldás minden *v* csúcshoz egy címkét rendel.

A azt jelenti, hogy minden élre.

(Egerváry tétele lényegében a max=min dualitását mondja ki csak LP nélkül megfogalmazva.)

## Intervallumgráf

### Intervallumok reprezentációja mátixxal

Adott intervallumhalmaza, aminek eleme [1,n] intervallum és ennek véges sok rész intervalluma.

Defináljunk

(Tehát a mátrix sorai az egész időpillanatok, míg oszlopai az intervallumok és 1 érték áll, ha az adott időpillanatban aktív az intervallum.)

(Tetszőleges intervallumokra is definálható, ez esetben a legbővebb intervallumot feleltetjük meg [1,n]-nek, a többit pedig a metszések szerint definálhatjuk.)

### Intervallum metszési mátrixának TU tulajdonsága

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, +1 vagy 0.\*\*

\*Egy mátrixban, ha egy oszlopot x+y alakban fel tudjuk írni, akkor a determinánsát fel lehet írni két mátrix determinánsának az összegeként ahol a kiválaszott sor helyett szerepel x és y.

Itt a két mátrix az eredeti mátrix és annak egy változata, ahol az i-dik oszlop helyett j-dik oszlop (-1-szerese van).

De a második mátrixban az i-dik oszlop kétszer szerepel, ezért 0 a determinánsa.

\*\*Bázis elhelyezéskor csak az átlókba elhelyezett permutációban nem lesz 0 szorzó, ezért csak ennek a produktuma határozza meg a determinánst, de ez tudjuk, hogy csak 0 és 1 értékűek. (A -1 az olszopok cserélgetése miatt lehet.)

### Intervallumgráfok egyenletes színezése *k* színnel

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.

Egyenletes egy színezés, ha az i-t tartalmazó intervallumok száma di, akkor ezek közül minden felhasznált szín esetén az ilyen színű intervallumok száma vagy .

### Intervallumgráfok egyenletes színezése *k* színnel IP segítségével

Az eredeti feladat helyett azt oldjuk meg, hogy keressünk az intervallumoknak egy olyan részhalmazát, amire igaz, hogy minden i időpillanatban vagy darab van a kiválasztottak közül.

Ekkor biztos színezhetjük egy színűre, majd elhagyva k-1-vel folytathatjuk a színezést míg minden intervallumnak nem lesz színe.

(Vegyük észre, hogy mindkét egyenlet rendszer egymást követő egész számokat ad meg alsó és felső határnak, tehát lényegében az egyik határ értékét kell felvennie.)

Az A mátrixról ismert, hogy TU továbbá TU-tartó műveletekkel előállítható a teljes IP mátrix:

1. Felvesszük A minden sorát még egyszer
2. Az újonnan felvett sorokat -1-vel szorozzuk
3. Felvesszük az egység mátrix sorait kétszer
4. Szorozzuk egyik példányát minden egységvektornak -1-vel

Az IP feladat mátrixa tehát TU, a hasonlítási vektor valamilyen számok alsó és egész részei, illetve 1 és 0 értékeket tartalmaz, tehát egész.

Vegyük észre, hogy a vektor triviális megoldása (és így optimuma) az LP-nek, de a fentiek értelmében („A” TU és „b” egész) léteznie kell egész helyű optimumnak is, azaz az intervallum gráfok mindig színezhetőek egyenletesen.

# VII. A lineáris és egészértékű programozás alkalmazása hálózati folyamproblémákra: a maximális folyam, a minimális költségű folyam és a többtermékes folyam feladatai, ezek hatékony megoldhatósága a tört-, illetve egészértékű esetben

## Jelölések

irányított gráf

* két kitüntetett csúcs: forrás ás nyelő
* nemnegatív *kapacitásfüggvény*.
* tetszőleges függvény.
* : *v*-be belépő élek összege *x* szerint
* : *v*-ből kilépő élek összege *x* szerint
* *x* *folyam*, ha
* *x* megengedett, ha
* A folyam értéke
* B a G illeszkedési mátrixa

## Maximális folyam, mint LP

### Program

### TU tulajdonság

A B mátrixról ismert, hogy TU továbbá TU-tartó műveletekkel előállítható a teljes IP mátrix:

1. Felvesszük B minden sorát még egyszer
2. Az újonnan felvett sorokat -1-vel szorozzuk
3. Felvesszük az egység mátrix sorait kétszer
4. Szorozzuk egyik példányát minden egységvektornak -1-vel

Az IP feladat mátrixa tehát TU, a hasonlítási vektor 0 és c(e) értékeket tatalmaz, tehát ha a kapacités egész, akkor van egész értékű maximális folyam is.

(Ha egyáltalán létezik folyam rajta és korlátos.)

## Minimális költségű folyam keresésére

### Feladat

Cél:

Tehát a legalább M költségű folyamok közül keressük a legolcsóbbat.

### Program

B\* legyen a folyam és kapacitás feltételeket leíró TU mátrix amit már ismertettünk.

### TU tulajdonság

Az még egy sor, de ettől még TU marad, hisz lényegében az eredeti B illeszkedési mátrix t-nek megfelelő sorát kell csak visszavenni hozzá.

## Alkalmazás többtermékes folyamra

### Feladat

(s,t) helyett k db (si,ti) forrás és nyelő páros

x(i,j) := i-dik élen, a j-dik termélből mennyi folyik át ~ k\*e változónk van

(Tehát k termékünk van, az i-dik termék csak si-ben termelődik és ti-ben nyelődik el, a kapacitás viszont az összes termékre egységesen vonatkozik.)

### Program

A Kirchoff feltételek ugyanúgy kifejezhetőek (i-dik termék esetén csak az (si,ti) sorokat hagyjuk el).

Arra kell figyelni, hogy most k\*e változónk van míg az illeszkedési mátrix jobb dimenziója e, azaz ahelyett hogy egymás allá raknák az adott mátrixokat egymás mellé tudjuk ragasztani és a sor elhagyása helyett az i-dik termék mátrixában az (si,ti) sorait nullvektorral helyettesítjük.

A kapacitás feltételek viszont újjelegűek:

### TU tulajdonság

Mivel mindig más sort nullázunk ki, így már nem beszélhetünk arról, hogy már létező olszopot veszünk hozzá plusszba és így a TU tulajdonság sem garantált.

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

## Matroid, mint függetlenségi axiómák

### Axiómák

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

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

### Alapfogalmak

|  |  |  |  |
| --- | --- | --- | --- |
| Fogalom | Matroid | Mátrixmatroid | Körmatroid |
| X⊆E és X∈F | független | független | körmentes/ erdő |
| X⊆E és X∉F | összefüggő | összefüggő | nem körmentes |
| max X⊆E és X∈F | bázis | bázis | feszítő erdő |
| min X⊆E és X∉F | kör | - | kör |
| |X|: max X⊆E és X∈F | dimenzió/rang | dimenzió | n - |komponens| |
| egy elemű összefüggő | hurok | nullvektor | hurok |
| mindentől független | híd | - | híd |

## Matroid típusai

### Definicók

* *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.

### Uniform matroidok grafikussága

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

*U4,2* nem grafikus (3 kört alkot, a negyediket úgy kéne beúzni, hogy minden élpárral kört alkosson).

Ha 2 <= k <= n-2, akkor k-2 összehúzás és n-k-2 elhagyás után *U4,2*-t kapnánk.

## Matroid, mint rangfüggvény

### Axiómák

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

1. .
3. (szubmodularitás)

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

### Kapcsolat a függetlenségi axiómákkal

Függetlenségból rang

(Tehát egy (E,F) matroid egyik részhalmazának a rangja az abban található maximális részhalmaz számossága.)  
Az így definált r rang függvény teljesíti a fenti axiomákat.

Bizonyítás:

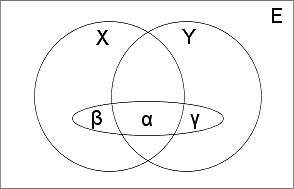
(R1) – (R3) a rangfüggvény definíciójából adódik.

(R4):

Keressünk -ban maximális függetlent, legyen .

Terjesszük ki A-t -ben és -ban maximális független B és C halmazokká.

*X*-ből *β*, *Y*-ból *γ* új elem kerül *B*-be.

A rangfüggvények a következőképp alakulnak:

Összesítve: .

Rangból függetlenségi:  
Ha egy r függvény teljesíti a rang axiómákat, akkor egyértelműen létezik egy (E,F) matroid, aminek r a rangfüggvényve, még pedig az alábbi módon adható meg:

}

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

## Mohó algoritmus matroidon

### Feladat

matroid

nemnegatív súlyfüggvény.

.

(w = 1 esetén, ez pont a rang keresése)

### Mohó algoritmus optimális matroidon

Tétel:

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, ha mohó algoritmus optimális

Tétel:

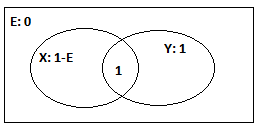
(F1) és (F2) kell, hogy egyáltalán értelme legyen a mohó algoritmusnak.

(Igazából egy kicsivel kevesebb is elég, leszálló halmaz rendszer helyett elég hogy egy elemet ki lehet venni a függetlenség elrontása nélkül => greedoid.)

Ekkor ha a mohó algoritmus optimális megoldást ad, akkor (F3) is igaz lesz!

Bizonyítás: Indirekt

Mohó algoritmus OK, de (F3) nem teljesül.

Ilyenkor , amelyeknél fennáll, de nem létezik olyan , amelyre teljesülne. Legyen a súly függvény a következő:

Az algoritmus először kiválasztja *Y* elemeit. Ezt követően az indirekt feltevés miatt már nem választhat *X – Y*-beli elemeket, csak 0 súlyúakat.

Legyen |X-Y|=a, |Y-X|=b, |X metszett Y| = c

Ekkor w(Y) = b + c, w(X)=a(1-E)+c.

Tehát ha E-t úgy választjuk meg, hogy a(1-E)>b (E < 1 – b/a), akkor elérhetjük, hogy a mohó algoritmus rossz eredményt adott.

## Matroid, mint bázisok

### Axiómák

bázishalmaza, ha:

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

### Kapcsolat a függetlenségi axiómákkal

Függetlneségből bázisok:

A maximális független halmazok eleget tesznek a bázis axiómáknak.

Bizonyítás:

B1:

F1 miatt legalább {0} eleme.

B2: Indirekt

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

B3:

Hisz különböző halmazokról van szó, legalább egy különböző elemük van, vegyünk ki X2-ből.

Ekkor |X1| > |X2| tehát (F3) miatt X1-ből átrakható egy X2-ba, hogy az maximális független legyen megint (azaz bázis), a kivett elem nincs X1-ben, ezért az átrakott elem tényleg egy másik elem.

Bázisokból matroid:

Ha egy halmazrendszer, amelyre teljesül (B1) – (B3), akkor matroid, ahol és ez a matroid egyértelmű.

### Rangból matroid

Ha egy r függvény teljesíti a rang axiómákat, akkor egyértelműen létezik egy (E,F) matroid, aminek r a rangfüggvényve, még pedig az alábbi módon adható meg:

}

## Matroid duálisa

### Definició

matroid bázisai ,

akkor a duális matroid bázisai .

Ebből már adódik

### Duális matroid tétel

Az matroid.

### Síkbarajzolható gráfok duálisa

Síkgráfokon a duális képzés és matroid képzés felcserélhető, az eredmény ugyanaz lesz.

(Körből vágás, vágásból kör lesz.)

(Fából pedig nem fa és nem fából fa.)

### Duális rangfüggvény

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

## Matroid minorok

### Definició

Matroid minorja olyan M matroid, amelyet elhagyás vagy összehúzással képeztünk.

### Elhagyás

matroid, elhagyása: matroid, ahol  
.

(Tehát elhagyunk elemeket, és a függetlenek ugyanazok maradnak, csak a kérdéses elemeket elhagyjuk belőlük is, ha szerepeltek.)

(Gráfoknál pl. élek elhagyása.)

### Összehúzás

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

(Tehát elhagyunk elemeket és azok maradnak függetlenek, akik az elhagyottakkal együtt is azok voltak.)

(Miért nem függetleneket adjuk meg? Problémát azt az esetet kezelni, amikor X magában is függő.)

(Gráfoknál pl. egy él összehúzása egy csúcsba.)

### Elhagyás és ősszehúzás kommutativítása

Az elhagyások és összehúzások fölcserélhetők.

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

### Minorképzés és duális

.

### Minorképzés és matroidképzés

G gráf esetén

M(G)\e =M(G\e)

M(G)/e=M(G/e)

(Tehát a minorképzés és a matroidképzés felcserélhető gráfok esetén.)

### Uniform mátrixok minorja

Un,k \ 1 = Un-1,k

Un,k / 1 = Un-1,k-1

(1 alatt bármelyik egy elem értendő.)

* Minorképzés nem vezet ki az uniform matroid halmazából

### Wagner tétel

G síkbarajzolható ⬄ M(G)-nek nem létezik M(K5) vagy M(M3,3)

Nem ugyanaz, mint Kuratowszky tétele!

Kuratowszky csak soros pontok összevonását engedi.

(Pl. Peterson gráfnak van K5 minorja, de soros izomorj részgráfja nincs, K3,3.val van csak.)

## Matroidok direkt összege

### Definició

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.

### Direkt összeg koordinázhatósága

Koordinátázza Ai Mi-t, ekkor M1++M2-t koordinátázza az alábbi mátrix:

|  |  |
| --- | --- |
| A1 | 0 |
| 0 | A2 |
| A1’ | A2’ |

(Ezt nevezzik blokkdiagonálisnak, Ai’ csak a két résznek adott név, nem a mátrix része.)

Ai’ oszlopai nyilván pontosan ugyanakkor függetlenek amikor Ai oszlopai, hisz csak nullákkal tolldottuk meg.

Egymás függetlenségeit pedig nem tudják elrontani, hisz ahol az egyiknek értékei vannak, ott a másikban nullák szerepelnek.

## Matroid koordniázhatósága

### Definició

M koordinázható T test felett, ha létezik olyan T test feletti A mátrix, aminek a matroidja megfeleltethető M-vel.   
(Azaz M(A) elemei és A soraii között létezik olyan bijektív leképezés, hogy egy részhalmaz vagy mindkettőben független vagy egyikben sem.)

(Pl. U4,2 nem koordinázható a bináris test felett, hisz (1,0), (0,1) és (1,1) után olyan vektornak kéne jönnie, amelyik mind a hárommal párban független, de csak (0,0) vagy az egyik ismétlése létezik a bináris testben.)

### Bázis koordinázhatósága egységmátrixként

Koordinázásnál feltehetjük, hogy r(E) sorból áll és az első r(E)xr(E) résznégyzetmárix egységmátrix.

Sorok eldobása:

Ha a rang r(E), akkor max r(E) lineárisan független sor van, ezeket tartjuk meg.

Ennél kevesebb nyilván nem elég, ettől a rang nem csökkenhet még.

Egységmátrix kialakítása:

Ha a rang r(E), akkor létezik egy nem szinguláris (azaz det != 0) r(E)xr(E) részmátrixa. Ez egy bázistranszformáció által egységmátrixba vihető át, ami viszont a függetlenek nem változtatja meg.

(Konkrétan a talált bázis inverzével kell balról megszorozni, ami létezik, hisz det != 0.)

### Duális reprezentációja T test felett

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

### Bizonyítás

Feltehető, hogy M-t koordinázó A r(M)=r sorú és r(M) első oszlopában az egységmátrix van, legyen ez M1=(Erxr|A), ez esetben azt állítjuk, hogy M2=(AT|En-rxn-r) koordinátázza a M\*-t.

Ehhez azt fogjuk belátni, hogy ha M1-ben B oszlopok bázist alkotnak, akkor E\B oszlopok M2-ben bázisok.

Egy tetszőleges B bázist vizsgáljunk amelynek első valahány oszlopa E-beli, a többi A-beli.

(Természetesen semmi nem garantálja, hogy egymás mellett oszlopokról van szó, de az oszlopok sorrendje nem jelentős, hiába mátrixról beszélünk, igazából vektorhalmazról van szó, nyugodtan rendezzük nekünk kényelmes sorrendre az olszopokat bár igazából a kimondott következtetések enélkül is igazak lennének.)

Ekkor négy részre osztható a mátrix, a bal felső sorokban nullmátrix van, bal alsóban egy egységmátrix, jobb felül valami A0 mátrix és alatta valami ismeretlen.

Átrendezve megtalálhatóak ezek a duális mátrixában is, csak A0 helyett transzponáltja áll.

|  |  |  |  |
| --- | --- | --- | --- |
|  | 0 | A0 |  |
|  | E | ? |  |
|  | B | |  |
| A0T |  |  | 0 |
| ? |  |  | E |
| E-B |  |  | E-B |

Mivel B bázis, ezért determinánsa nem nulla, ám det(B)=|E||A0| kifejtve E szerint, azaz det(B)!=0 ⬄ det(A0) != 0.

Viszont E\B determinánsa hasonló módon kifejthető A0T-re, a transzponálás pedig nem változtat a determinánson, tehát valóban B és E\B ugyanakkor független, és ha B bázis, akkor E\B is az (a méreteik miatt maximálisak).

Ez pont a duális definiciója, tehát akkor tényleg koordinátázza a duálist az adott mátrix.

# XI. 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 definició

### Grafikus matroid

, {*G*-beli erdők}

### Kografikus matroid

Grafikus matrtoid duálisa

### Lineráis matroid

Matroid ami koordinázható valamelyen test felett

### Bináris matroid

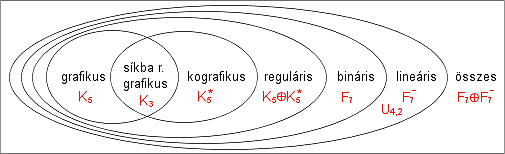
Matroid, ami koordinázható a bináris test felett

### Regurális matroid

Matroid, ami koordinázható minden test felett.

## Matroid osztályok kapcsolata

### Venn diagram



A minorkézpés egyik halmazból se vezet ki, a duális képzés a grafikus és kografikus között oda-vissza kapcsolat, de a többiből viszont nem vezet ki.

### Grafikus gráf reguláris

Koordinizása:

Alábbi módon konstruáljuk a gráf mátrixát:

1. Irányítsuk meg az éleit tetszőleges módon
2. Az Rn egységvektorait és nullvektort osszuk ki a csúcsokra

(0 és 1 értéket használunk csak, ez minden testben léteznie kell.)

(Egy fa mérete max. n-1 lehet, ezért az egységvektorok lehetnek bázisok és r(M)=n-1.)

(Ettől lehetne kisebb is, ha nem összefüggő.)

1. (x->y) élt reprezentálja az x-y vektor
2. Ezekből építünk egy mátrixot

Ahhoz hogy belássuk, hogy a kreált mátrix tényleg és a gráf tényleg ugyanazt a matroidot fejezik ki azt kell belátnunk, hogy a függetleneik halmaza megegyezik, ehhez pedig azt látjuk be, hogy kölcsönösen tartalmazzák egymást.

Körmentes => lineárisan független: Indirekt

Tfh: , de megfelelő élek viszont nem kört határoznak meg

A vektorok egységvektor (és nullvektor) különbségeként jöttek létre, azaz csak 0 és legfeljebb 2 db +-1 értékből állhatnak.

Ahhoz hogy ezek lineárisan összefüggőek legyenek az összegüknek 0-nak kell lennie nem nulla együtthatókkal, azaz ha egyik sorban van nem nulla érték, akkor legalább még egynek kell lennie (több is lehet.)

De egy sorban csak úgy lehet nem nulla érték, ha az adott él csatlakozik a sort meghatározó csúcshoz, azaz egy sorban lévő nem nulla értékek alsó becslés a csúcs fokára.

Azaz két nem nulla érték minden oszlopban => minden scúcs foka legalább 2 => létezik kör.

Lineárisan független => körmentes: Indirekt

Tfh:

Irányítsuk meg a kört az indexek sorrendjében majd képezzük a lineáris kombinációjákat, úgy, hogy az együttható +1, ha a kör irányába mutat, és -1 ha ellentétes a kör bejárási irányával.

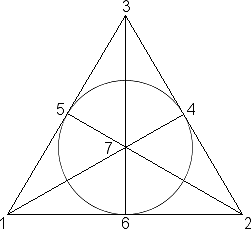
Tehát j és j+1 csúcs közötti él értéke xj – xj+1 a kombinációban.

(Tehát az él forráscsúcs vektora +, az él nyelőcsúcs vektora -1 együtthatóval lesz a kombinációban.)  
(x és y csúcsvektorok esetén, x->y élre x-y élvektor lesz és együttható 1 lesz, míg y->x él esetén az élvektor y-x, de együtthatója -1, így -1(y-x) = x-y.)  
Ekkor mivel körről van szó minden csúcs egyszer lesz bemenő és kimenő része is egy élnek, azaz egyszer szerepel + és – előjellel, azaz az összeg 0 lesz.

Találtunk egy nem triviális lineáris kombinációt, ami 0 értéket ad, tehát összefüggőknek kell lennie az élvektoroknak.

## Fano matroid

### Definició

Alaphalmaz: .

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

### Anti-Fano

A legfeljebb kételemű részhalmazok függetlenek, továbbá azok a háromeleműek, amik nincsenek egy egyenesen, ez az anti-Fano matroid ().

Tehát Fano és Anti-Fano-ban is összefüggőek az alábbiak {1,3,5}, {1,2,6}, {1,4,7}, {2,3,4}, {2,5,7},{3,6,7}, és Fano-ban még a {4,5,6} is.

### Fano és Anti-Fano koordinázhatósága

Fano koordinázható 2 karakterisztikájú test felett (pl. Bináris) míg Anti-Fano minden olyan test felett, amely nem 2 karakterisztikájú.

## 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.

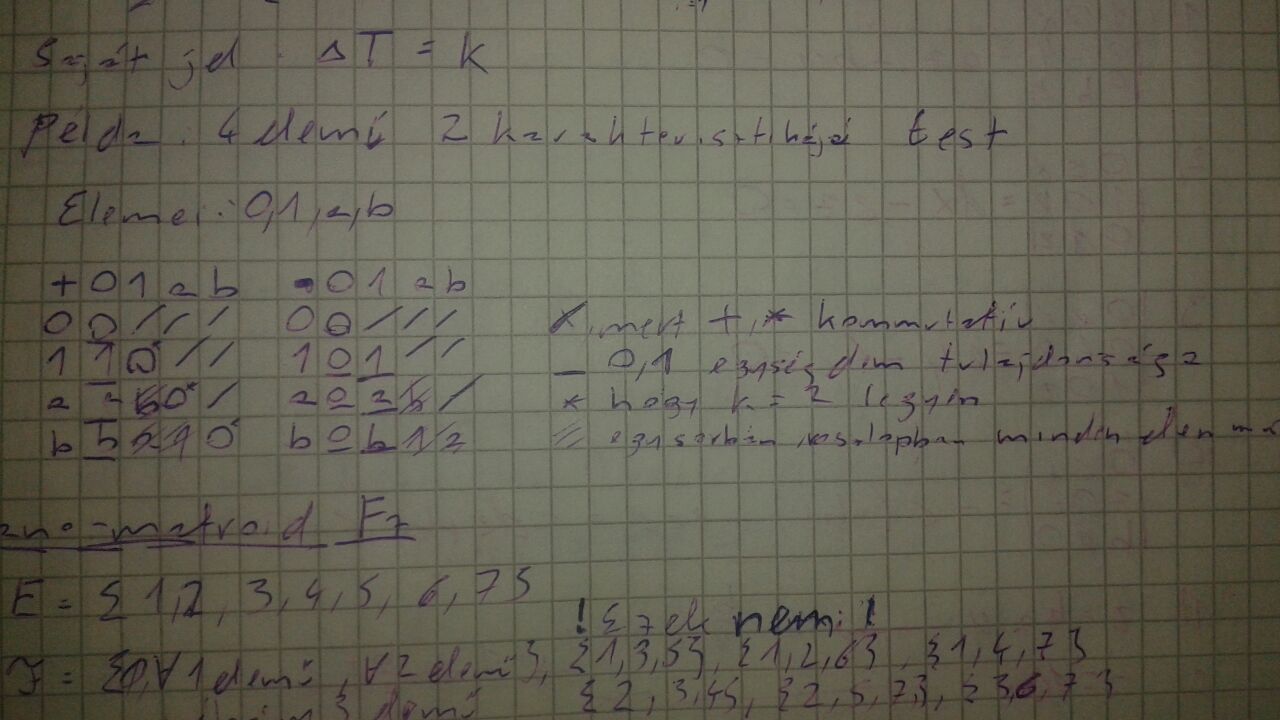
## Seymour tétel

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.

### Test karakterisztákáka

T test karakterisztikája

(Tehát olyan számok minimumát keressük, amennyiszer egy T-beli testet önmagával összeadva nullát kapunk. A szokásos számhalmazokon ilyen szám nincs (0-nak szoktuk definiálni ezért), a maradékhalmazokon a moduláló szám ilyen, az adott műveleti táblák meletti {0,1,a,b} karakterisztikája pedig 2.)

### https://cdn.discordapp.com/attachments/243868652762300416/318007334020448257/13c14c0b-3bbf-44c6-8f1d-7de854c86da7.jpgFano és Anti-Fano koordinázhatóságának bizonyítása

Kezdjük el koordinátázni a Fano matroidokat.

Tudjuk, hogy rangja 3 (hisz annál nagyobb független nincs), ezért a koordinátázó mátrixról feltehetjük, hogy 3 sora van és első három oszlopában egységmátrix szerepel.

Pl az {1,2,3} bázist feleltessük meg az egységmátrixnak. (Ezzel még semmilyen koordinázhatóságot nem rontottunk el, hisz 0 és 1 eleme minden testnek van.)

Ezek után az ábra alapján paraméterezzük a többi elemhez tartozó vektorokat.

Miért (c,0,d)? Tudjuk, hogy (1,0,0) és (0,0,1)-vel együtt összefüggő, ezért a középső koordinátában neki is 0-nak kell lennie, különben nem létezhetne nullvektort adó lineáris kombinációjuk.

(A többi analóg módon, a középső elem pedig teljesen ismeretlen egyelőre.)

Így paraméterezve az oldal egyeneseken lévő összefüggőségeket garantáltuk, már csak 3 belő egyenes (Fano esetén +1 a kör miatt) összefüggőségét kell garantálni, ehhez vizsgáljuk a determinánsokat, ezeknek kell 0-nak lenni.

Ezen három egyenletből kapjuk azt, hogy ade=bcf.

(Ez nyilván minden test felett garantálható.)

Az utolsó hármas Fano esetén összefüggő (=0) ( Anti-Fano esetén független (!=0)).

bcf helyére behelyettesítve ade-t (ahogy tudjuk az első három egyenlőtlenségből) azt kapjuk, hogy:  
Fano esetén: ade + ade = 0 ⬄ Test karakterisztikája 2

Anti-Fano esetén: ade + ade != 0 ⬄ Test karakterisztikája nem 2

Ezzel megis adtuk egy megfelelő karakterisztikájú test feletti koordinázást mindkét matroidnak, csak arra kell figyelnünk, hogy ade=bcf tetszőleges számokat válasszunk.

(Meglepően x,y,z-nek nincs is jelentősége.)

Ebből az is következik, hogy a direkt összegük semmilyen test felett sem lehet koordinázható, hisz egyszerre kéne 2 és nem 2 karakterisztikájunak lennie.

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

## Matroid összege

### Definició

és matroidok összege

, ahol , hogy és , illetve .

(*X* előáll egy -beli és egy -beli elem uniójaként)

(Ha előáll uniójukként, akkor diszjunk unióként is előáll, hisz pl. az egyikből a metszett elemeit el lehet dobni, ettől a függetlenség nem romolhat el, és az unió változatlan.)

### Matroidok összege is matroid

Lássuk be, hogy a függetlenségi axiómák továbbra is igazak maradnak.

(F1): 0 U 0 = 0

(F2):

(F3) indirekt:

Definíció szerint *X*, *Y* halmazoknak létezik (esetleg többféle) és felbontása, ahol , illetve .

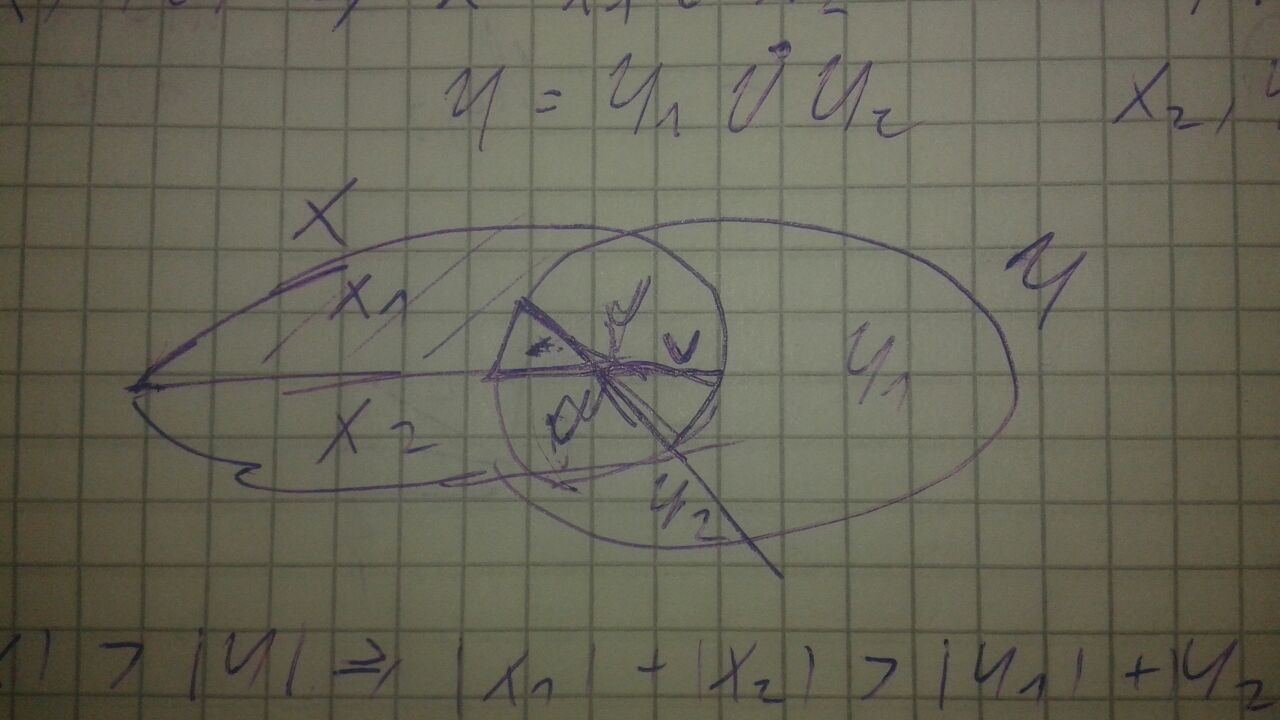
Válasszunk olyat, hogy a részhalmazok diszjunktak legyenek és érték legyen minimális, továbbá legyen . (|X| > |Y| tehát egyik párra igaznak kell lennie.)

Ekkor *M*1 matroidra alkalmazva (F3)-mat, létezik , amelyre .

Ha , akkor (hisz x se Y1 se Y2-nek nem része), de ez ellentmondás.

Tehát . Ekkor egy másik felbontás, ráadásul

összeg eggyel kevesebb, mint az mint érték, ami ellentmondás (minimálisra választottuk).



## k-matroid metszet probléma (k-MMP)

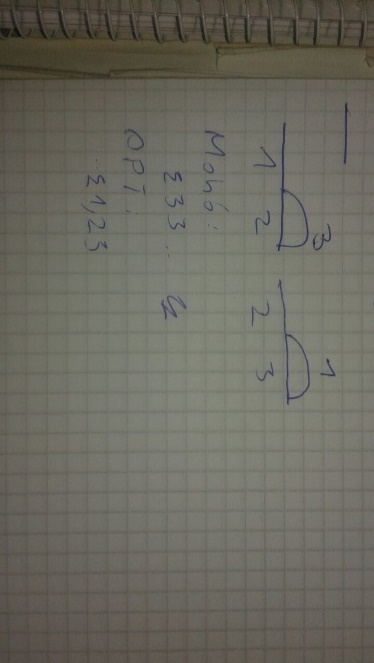
### Feladat

?

(Azaz minden matroidnak egy-egy függetlenét vehetjük és a kérdés, hogy tudunk-e úgy választani, hogy a metszettük megfelelően nagy-e, avagy keressük azon elemeket, amelyek minden matroidban függetlenek és ezek közül a legnagyobbat. Ezek nem feltétlenül határoznak meg egy matroidot.)

### 2-MMP bonyolultsága

Mohó algoritmus jó 1-MMP esetén (bázis keresés), de már 2-MMP-re se jó:



De, létezik rá polinomiális algoritmus, a 2-MPP-re visszavezethető.

### 3-MMPk bonyolultsága

NP-teljes, hisz visszavezethető rá az irányított u->v Hamilton-út keresése (NP-sége meg triviális).

* 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 u->v Hamilton-útjai, hiszen:

M1 és M2 metszett bázisai olyanok, hogy minden csúcs ki-be foka 1, azaz utak uniói és körök uniója.

(Illetve u be-foka 0 => kezdőpont, v ki-foka 0 => végpont.)

M3 miatt viszont nem lehet kör, ezért pont az utai közül keressük a legnagyobbat.

(Megjegyzendő, hogy semmi nem garantálja azt, hogy a metszett bázisok összefüggők, de egy

|V|-1 körmentes út n különböző csúsból áll, ami az össes, és ezért összefüggő is lesz, kisebb utak keresésére ugyan ez a módszer nem lenne alkalmas, hisz találhatnánk pl. n-3 út helyett egy n-5 és 2 hosszú utat is.)

### k-MMP bonyolultsága

NP-teljes, hisz 3-MMP visszavezethető rá, ha kiegészítjük az három matroidot k-3 szabad matroiddal.

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

## k-matroid partíciós probléma (k-MPP)

### Feladat

?

(Azaz particionálhatóak úgy, hogy kiadják a teljes alaphamazt avagy az összes matroid összege a szabad matroid-e.)  
(Természetes most is igaz, hogy feltehető, hogy Fi-k diszjunktak.)

### Bonyolultság

Mohó:

A mohó algoritmus jó 1-MPP-re, de 2 esetén már nem.

NP: Tanu a particionálás.

coNP: Tanu egy X⊆E halmaz, ami biztosan összefüggő az összegben, azaz ∑ri(X)<|X|.

**P:**Ezek után sejthető, hogy létezik rá egy polinomiális algoritmus is:

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.   
  Ilyen él azt jelenti, hogy x felvehető i-dik particióba.
* (x→y) ∈ E(G), ha ∃i x∉Ei, y∈Ei, Ei∪{x}∉Fi, de Ei∪{x}-{y}∈Fi.   
  Ilyen él azt jelenti, hogy x bevehető egy particióba, ha helyette y-t kivesszük.

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.

1. Különben STOP, nemleges a válasz, és a tanú az (E-∪Ei)-ből irányított úton elérhető pontok halmaza.

∑ri(X)<|X| lesz igaz, hisz különben x még bővíthető lenne rajta kívüli elemmel és nem állt volna le az algoritmus.

(Fontos, hogy a legrövidebb javító utat használjuk. Páros gráfban TP kereséskor mindegy, folyam maximalizálás esetén kell, hogy polinomiális legyen, k-MPP esetén viszont akár hibás eredményt is adhat.)

## 2-MMP visszavezetése 2-MPP-re

### 2-MMP

?

### 2-MMP

?

### Redukció

M1=(E,F1), M2=(E,F2), p egész.

Ha p>min(r1,r2), nem a válasz (hisz nem lehet p metszett független, ha egyikben eleve nincs ekkore)

Csonkoljuk a matroidokat addig, amíg a rangjuk p-re nem csökken.

Ezzel a problémát redukáltuk közös bázis keresésére. r1=r2=p-re.

(Hisz >p függetlenek nincsenek, =p pedig pont a bázisok, ha pedig van >p metszett annak bármelyik p méretű része is metszett, ezért ekvivalens a két feladat.)

Ezek után kimondható, hogy:

Létezik közös bázis ⬄ 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 jó particionálása a szabadmatroidnak.

⇐: legyen E=C u D (~ E-D=C )egy olyan diszjunkt felbontása a szabadmatroidnak, ahol C∈F1 és D∈F2\* és C,D bázis, ekkor:

|C| <= p (hisz C független M1-ben, ahol a bázisok p méretűek)

|D| <= n-p (hisz D független M2\*-ben, ahol a bázisok n-p méretűek)

|C|+|D| = |E| = n (hisz diszjunkt felbontásai a teljes alaphalmaznak)  
|E| = |C|+|D| <= p + n - p = n = |E|.

Tehát az egyenlőség is teljesül, és a egyenkénti <= miatt egyenként is:

|C| = p, azaz |C| független és p méretű => bázis M1-ben

|D| = n-p, azaz |D| független és n-p méretű => bázis M2\*-ben

Ha M2\*-ben D bázis, akkor M1-ben E-D=C bázis, tehát beláttuk, hogy C M1 és M2-ben is bázis.

**Def.:** M = (E,F) csonkoltja M′ = (E,F′) ahol F′ az F elemeit tartalmazza a bázisokon kívül.

A matroid rangja ettől eggyel csökken.

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

## k-polimatroid rangfüggvény

### Definició

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

* r(∅)=0
* r(x)≤k|X| (k=1 esetén matroid) ~ szubmodularitás
* X⊆Y ⇒r(X)≤r(Y)
* r(X)+r(Y)≥r(X∪Y)+r(X∩Y)

## k-polimatroid matching probléma (k-PMMP)

### k-matching

X⊆E k-matching, ha r(X)=k|X|

### Feladat

**Ad**ott r és t∈**N**.

Van-e legalább t elemű k-polimatroid matching?

### ****Speciális esetek****

Párosítás:

r(X)=|X által lefedett pontok halmaza|<=2|X|

2-matching: párosítás (minden él két különböző csúcshoz kötödik)

k-MMP:

Adott k db matroid r(X) = **Σri(x) <= k\*|X| (a rangfüggvények összege)**

k-matching: X független mindegyik matroidban (r(X)=|X| <=> X független)

Páros gráfban párosítás

Ez közös speciális eset a fenti kettő problémának, a párosítás triviális.

2-MMP-re visszavezethető, legyen G=(A,B;E).

M1:= két él összefüggő, ha A-ban egy csúcsba mennek

M2:= két él összefüggő, ha B-ben egy csúcsba mennek

Nyilván M1 és M2 bázisai olyanok, hogy se A se B-ben nincs közös csúcsok.

### Bonyolultság

**Természetesen NP-nehéz, hisz visszavezethető rá a k-MMP, de több is állítható róla: nem polinomiális (és még nem is NP), már a 2-PMMP sem P-beli általánosságban.**

## **Matroidpárosítási probléma (2-PMMP)**

### Bonyolultság

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.

### Bizonyítás

Legyen E páros, |E|=n.

.

Tehát |X|>n/2-re k-matching lesz, |X|<n/2-re nem, =n/2 esetén pedig majd eldöntjük.

(Azt nem bizonyítjuk, hogy ez miért szubmoduláris.)

P=n/2-re ezt nem lehet polinomiális időben eldönteni.

A biztos k-matching-k <n/2 ezért nem lesznek megfeleők, a >n/2 nem k-matching-k.

Tehát a feladat a |X|=n/2 halmazon dőlik el, ezek számossága viszont , ami , azaz exponenciális.

Egy polinomiális algoritmus csak polinomiálisan sokat tud megvizsgálni belőle, ezért lesz olyan, amit nem vizsgál meg.

Így mindenképpen el tudom rontani a válaszát, hisz ha rákérdez egyikre, akkor |X|-1-nek állítom az értékét.

Ha ezek után IGEN választ ad, akkor az összes többit is |X|-1-nek választom és hibás lesz a válasza.

HA NEM választ ad, akkor az egyik nem vizsgáltat 2|X|-nek választom és így lesz hibás a válasza.

(Megjegyzés: a „közben döntöm el” lényegében annyit jelent, hogy tetszöleges minden elemre hogy melyik értéket választom, így minden algoritmusra ami egyet esetleg jól oldana meg, tudok muttatni egy másikat, amin szükségszerűen rossz választ fod adni.)

## ****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.

(Lényegében az a trükk, hogy a halmaz elemei az indexek, amikhez két vektor tartozik, így ha két vektor független, akkor „egy indexhez” két független is lesz.)

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

# XV. Polinomiális időben megoldható feladat fogalma, példák. Az NP, co-NP, NP-nehéz és NP-teljes problémaosztályok definíciója, viszonyaik, példák problémákra valamennyi osztályból. NP-nehéz feladatok polinomiális speciális esetei: intervallumgráfok színezése, algoritmus a maximális független ponthalmaz problémára és az élszínezési problémára páros gráfokon.

## Eldöntési probléma osztályok

### Definiciók

Csak eldöntési problémákra van definálva (megadható pl. az IGEN-t adó bemenetek halmazával és az összes bemenetet leírni képes abécé-vel.)

A probléma osztályok:

* P: polinomiális időben megoldható
* NP: ha a válasz IGEN, akkor létezik egy tanuja, ami segítségével polinomiálisan belátható, hogy tényleg ez a válasz.
* co-NP: ha a feladat negáltja NP (azaz ua, mint NP, csak a NEM válaszra kell tanu létezzen).
* NP-nehéz: minden NP-beli probléma Karp-redukálható rá

(azaz ha adott egy NP-beli probléma, annak minden bemenetét át tudjuk polinomiális időben alakítani egy saját bemenetünkre, hogy az adott válasz nem változik)

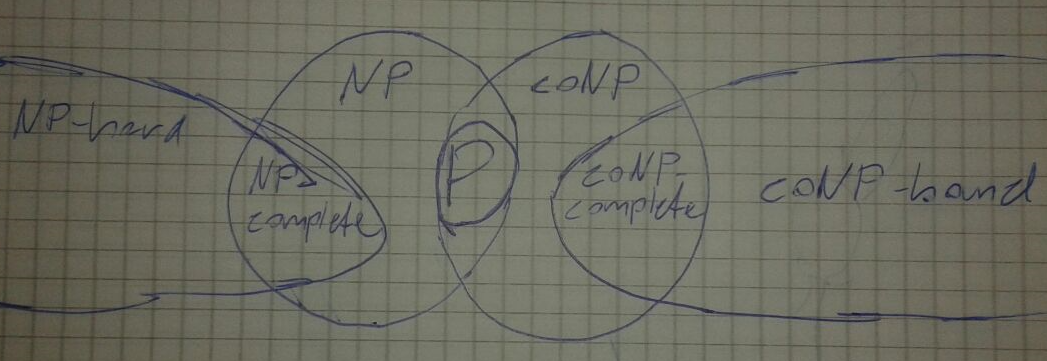
(Ezzel azt látjuk be, hogy az adott feladat nem lehet a sajátunknál lényegesen nehezebb, ha ezek közül bármelyiket meg tudnánk oldani P-ben, akkor az összes többi is P-beli lenne.)

* NP-teljes: NP-beli és NP-nehéz

### Viszonyaik

Nyitott kérdések:

* (sejtés: nem, sok NP-beli problémára nem ismert P-beli megoldás)
* (sejtés: igen, legtöbb problémára ismert P-beli megoldás)



### Híres eldöntési problémák

|  |  |  |
| --- | --- | --- |
| **feladat** | **probléma osztály** | **algoritmus** |
| összefüggőség | P | mélységi keresés (DFS), szélességi keresés (BFS) |
| TP páros gráfban | P | javító utas algoritmus |
| 2-Szín | P | BFS |
| 3-Szín | NP-teljes | - |
| Síkbarajzolhatóság | P | Kuratowski-tétel |
| k-Klikk | NP-teljes | - |
| k-Ftln | NP-teljes | - |
| k-Ftln páros gráfon | P | magyar módszer |
| Hamilton-kör | NP-teljes | - |
| utazóügynök probléma | NP-teljes | - |
| k-Hosszú kör | NP-teljes |  |
| leghosszabb irányított út keresése | NP-nehéz  (nem eldöntési) | - |
| leghosszabb irányított út keresése DAG-ban | P | szélességi keresés (BFS) |
| Gráf izomorfizmus | NP | *(Nem ismert, hogy P vagy NP-teljes)* |
| Megállási probléma | NP-nehéz  (nem NP!) | Véges időben eldönthetetlen |

## Független és lefogó halmazok

### Jelölés

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

### Viszony

## NP-nehéz feladatok polinomiális speciális esetei

### Intervallumgráfok színezése

Az intervallum elejétől indulva mohón színezzük a csúcsokat, ez maximum w(G) színt használ, aminél kevesebbel nem is lehetne kiszínezni.

(Indirekten tfh egy újonnan érkező intervallumot w+1. színnel akarjuk színezni, ekkor azt jelenti, hogy balról már létezik w+ intervallum, ami az előtt lévő színeket elhasználja és az új intervallum kezdeténél is aktívak, tehát akkor ezen kezdési pillanatban w+1 intervallum van fedésben, ami egy w+1 klikk lenne, ami ellentmondásos)

### Maximális független ponthalmaz páros gráfokon

Futtassuk a magyar módszert, ha már nincs javító út, akkor az alábbi helyzet alakul ki:

|  |  |
| --- | --- |
| A, B | Páros gráf csúcsosztályai |
| M | A jelenlegi párosítás |
| U | A\M (A csúcsok közül az M által fedetlenek) |
| T’ | U-ból alternáló úton elérhető B csúcsok |
| T | T’ párosításai |



(B \ T’) u (T u U) független csúcshalmaz, hisz

* (T u U)-ból induló élek T’-be kell, hogy menjenek (különben lenne javító út).

T’ u (A \ (T u U)) lefogó csúcshalmaz, hisz:

* (T u U)-ból induló élek T’-be kell, hogy menjenek (különben lenne javító út).
* A csúcshalmaz többi csúcsa pedig a lefogó csúcshalmaz része.

|T’ u (A \ (T u U))| = |M|, hisz:

Minden párosításbeli élnek pontosan egy csúcsát tartalmazza.

T’ u (A \ (T u U))-ról tudjuk, hogy minimális lefogó, hisz egy párosításnak minden éléhez legalább egy csúcsot tartalmaznia kell (azaz >= |M|).

Gallai tétel értelmében viszont |max független csúcshalmaz| + |min lefogó csúcshalmaz| = |V(G)|, de [(B \ T’) u (T u U)] u [T’ u (A \ (T u U))] = A u B, azaz a talált független ponthalmaz teljesíti ezt a feltételt, ezért maximális.

### Élszínezés páros gráfokon

König tétele: χe(G) = ∆(G).

Ennek bizonyítása egyben algoritmus arra, hogy ∆(G) hogy lehet kiszínezni páros gráfot P-ben:

G(A,B,E) gráf éleit sorban színezzük, mindig helyesen, ezalatt esetleg átszínezve a már színezetteket (de színtelenné nem alakítjuk).

e1: teljesen mindegy mire színezzük

: tfh. ei-1-ig már helyesen színezett

Definició szerint mind u és v csúcsra is max ∆(G) él illeszkedhet.

De maximum csak ∆(G)-1 szín lehet használatban mindkét csúcsban jelenleg, hisz legalább az él még színtelen, azaz mindkét csúcsban még van szabad szín(ek).

Ha ezek között van egyezés, azt a színt megkaphatja ei.

Ellenkező esetben legyen u egyik szabad színe piros, v-é meg kék.

Konstruáljuk G azon rész gráfát C-t, amiben csak azon élek és azok csúcsai vannak amellyek kék vagy piros színűek. I

C minden csúcs foka maximum 2 (egy piros és egy kék él), azaz diszjunk utak és körökre bontható.

u és v fokai viszont maximum egy (különben lett volna egyező szín), azaz egy-egy út végpontjai lehetnek csak.

Legyen az u-t tartalmazó út U, ezen nem lehet rajta v, hisz csak másik végpontja lehet, de az adott út páratlan hosszú (egyik színosztályból a másikba vezet) és váltakozó színú (hisz a színezes helyes), azaz ha az első él kék (u-ban piros volt szabadon), akkor az utolsó is, azaz v-ben is piros szín szabad lenne.

Tehát ekkor az U mentén felcserélhetjük a színeket, ezzel nem elrontva a színezést, de u-ban piros helyett a kék szín lesz szabad így, amivel színezhető.

# XVI. Additív hibával közelítő algoritmus fogalma, példa pont-, illetve élszínezési problémákra .A Hamilton-kör probléma visszavezetése a leghosszabb kör probléma additív közelítésére. k-approximációs algoritmus fogalma, példák: két-két algoritmus a minimális lefogó ponthal-maz keresésére és a maximális páros részgráf keresésére.

## Additív hiba

### Definició

C-Additív hibától eltekintve helyes egy megoldás, ha

Maximalizálásnál:

Minimalizálásnál:

### C-Additív közelítő algoritmus

Olyan A algoritmus, amely képes C-Additív hibától eltekintve helyes megoldást adni POLINOMIÁLIS lépésszámban.

(A közelítő megoldás értéke nem elég, a konkrét x’ megoldás kell.)

### Élszínezés 1-additív hibával

Ismert Vizing tétele szerint, hogy ∆(G) ≤ ∆χe(G) ≤ ∆(G) + 1 (egyszerű gráfon).

Létezik algoritmus, ami ∆(G) + 1 színnel polinomiális időben kiszínezi egy egyszerű gráf éleit.

### Csúcsszínezés 1-additív hibával síkba rajzolható gráfokon

Négyszíntétel értelmében 4 színnel biztos színezhetőek és erre létezik is polinomiális algoritmus.

Az additív hiba 3-ről 1-re csökkenthető, ha előtte egy DPS futtatásával meggyőződünk arra, hogy páros-e.

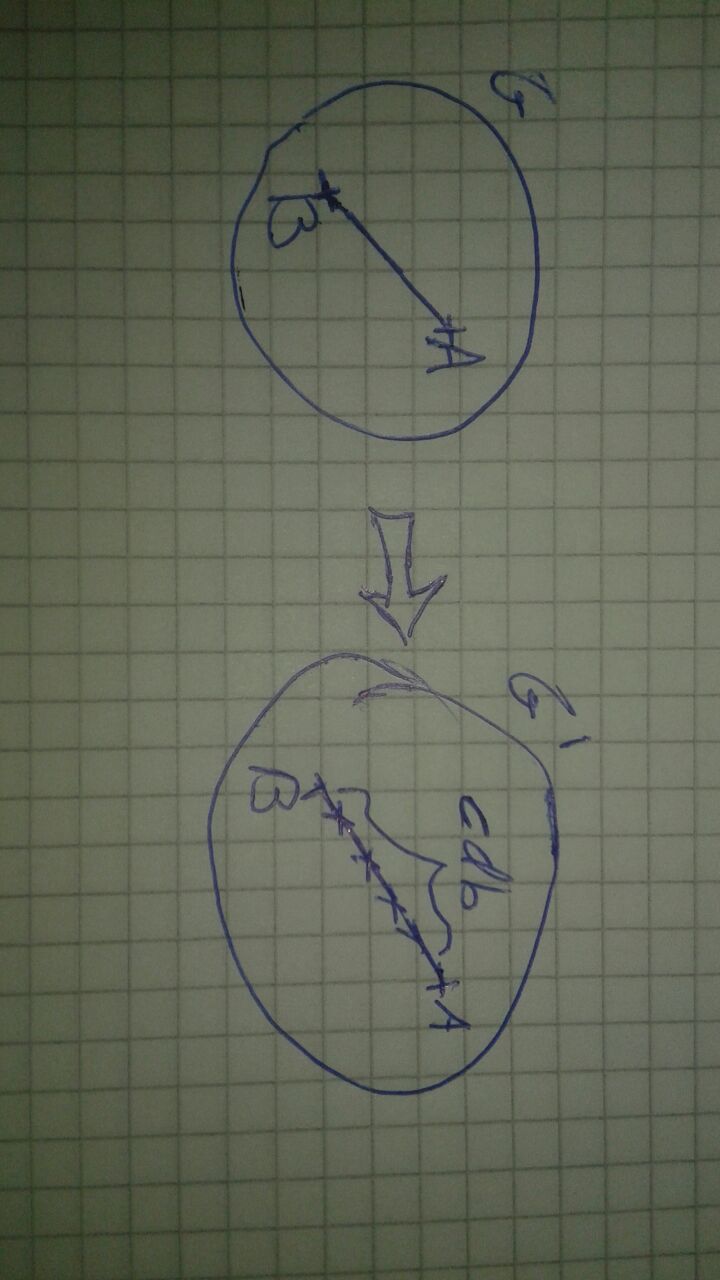
Ha páros, akkor a DPS mentén egy színezést is találtunk és így nincs hiba.

Ha nem páros, akkor 4 színnel színezzük, de eze setben a kromatikus szám legalább 3, azaz biztosak lehetünk benne, hogy max 1 a hiba.

### Hamilton-kör visszavezetése leghosszabb kör probléma additív közelítésére

A Hamilton-kör eldöntési problémát visszavezetjük a leghosszabb kör c-additív közelítésére.

G -> G’: az eredeti gráf minden élén felveszünk c db új csúcsot:



Vegyük észre, hogy G’ és G körei egymásnak bijektíven megfeleltethetőek.

Ráadásul G’ egy köre az eredeti kör hosszának c+1-szerese.

Ennek következménye, hogy G köreinek a hosszai „c+1”-es léptékekben nőnek, ami c additív hiba megengedése mellett is azt jelenti, hogy a mindig a leghosszabbat kell megtalálnunk.

Tehát ha találtunk egy k\*(c+1) hosszú kört G-ben az megfelel egy Hamilton-körnek G-ben.

Ha létezik Hamilton kör G-ben, akkor G’-ben létezik k\*(c+1) hosszú kör, a nála rövidebb kör már csak (k-1)\*(c+1) hosszú lehet csak, ami nem fér bele a c-additív hibába.

Tehát ha az additív problémát meg tudjuk oldani polinomiális időben, akkor a Hamiltön-kör problémát is meg tudnánk oldani polinomiális időben. (Ehhez még fontos azt is konstatálni, hogy az átalakítás maga megy polinomiáls időben.)

## Multiplikatív hiba

### Definició

k-multiplikatív hibától eltekintve helyes egy megoldás, ha

Maximalizálásnál:

Minimalizálásnál:

### k-approximációs algoritmus

A egy feladat k-approximációs algoritmusa, ha polinomiális időben képes egy feladatot k-multiplikatív hibától eltekintve helyesen megoldani.

### Maximális páros részgráf 2-approximációja

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

Kiindulunk egy tetszőleges kettéosztásból, és ha egy *p* pont áthelyezése növeli a vágás élszámát, akkor áthelyezzük.

Nem biztos, hogy minden csúcsot csak egyszer kell áthelyeznünk (egy későbbi áthelyezés miatt lehet mégis vissza akarnánk tenni), de azt tudhatjuk, hogy minden áthelyezésnél legalább egy éllel több fog menni a két színosztály között, azaz O(E) lépés után biztos véget ér.

Jelölje inner(v) = a v csúcsból saját osztályán belüli csúcsba vezető élek számát,

outer(v) = a v csúcsból a másik osztályba vezető élek számát.

Az algoritmus akkor áll le, ha minden csúcsra igaz, hogy: outer(v) >= inner(v).

Ekkor viszont

**Σouter(v) >= Σinner(v) és**

**Σouter(v) + Σinner(v) = |V|,**

**tehát**

**Σouter(v) >= |V|/2 >= |OPT|/2.**

### Maximális páros részgráf online 2-approximációja

Online algoritmus, létrehozunk két üresosztály és sorban beosszunk az összes csúcsot valamelyikbe.

Mohón osztunk be a két osztály közül abba helyezzük, amelyikkel végül több keresztélet kapunk.

Itt nem igaz, hogy outer(v) >= inner(v) viszont **Σouter(v) >= Σinner(v) továbbra is igaz, hisz minden elhelyezésnél úgy döntünk, hogy a Σouter(v)** legyen többségben.

### Minimális lefogó ponthalmaz 2-approximációja

Keressünk egy optimális párosítást, ezen csúcsait választjuk.

Lefogó:

Ha nem lenne, akkor létezne egy él amely egyik csúcsában sem csatlakozik bármelyik párosított élhez, azaz hozzá tudnánk venni a párosításhoz, de ez ellentmondás, hisz maximális a párosításunk.

2-approximáció:

jelölje a minimális lefogó csúcshalmaz méretét és a maximális független élek számát.

A mi párosításunk maximális, azaz és méretű, azaz a válaszott lefogó csúcshalmazunk elemű.

Ismert, hogy , ebből következik, hogy , ami a 2-approximáció definiciója.

### Minimális lefogó ponthalmaz 2-approximációja mohó algoritmussal

Vegyük észre, hogy a lefogó csúcshalmaz bizonyításakor csak annyit használtunk, hogy a párosítás nem bővíthető tovább és a 2-approximációnál felhasznált egyenlőtlenség pedig minden párosítás élszáma és lefogó csúcshalmaz csúcsszáma között fennáll nem csak a maximum és minimum között.  
Tehát nem kell optimális párosítás, bőven elég nem bővíthető, ezt pedig egy mohó algoritmussal is meg tudjuk keresni.

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

## Súlyozott halmazfedés

### Feladat

### Minimális lefogó ponthalmaz probléma visszavezetése a halmazfedési feladatra

R egy-egy eleme pont egy-egy csúcsnak felel meg, hogy melyikeket válasszunk be a lefogó csúcshalmazba és azon élek halmazából áll, amiket az adott csúcs lefog.

Ebből következik az is, hogy a halmaz fedés NP-nehéz.

### A halmazfedés közelítése

Legyen C ⊆ U a már eddig fedett értékei az U alaphalmaznak.

Ekkor mohó algoritmust alkalmazunk a legkisebb érték szerint.

(A a halmaz költsége, a részhalmaz által fedhető elemek száma, azaz a lefedett elem egységköltségét próbáljuk minimalizálni mohó módon. A \C jelentősége, hogy amit már egy másik halmazzal lefedtünk, azt nem akarjuk még egyszer lefedni, csak az újonnan lefedett elemek száma érdekel minket.)

### Közelítésének faktora

Bizonyítás:

Az előbb definált újonnan behozott elem egységárt az elemekre is defináljuk.

Ekkor egy f fedés öszköltsége az elemek f fedés szerinti költségüknek az összege.

Vizsgáljuk meg egy halmazát egy optimális fedésnek.

Indexeljük S elemeit amilyen sorrendben a f mohó algoritmus választotta őket és vizsgáljunk meg egy tetszőleges i-dik elemet (i <= r).

Ekkor , hiszen mikor -t fedni készül a mohó algoritmus, akkor S még i új elemet tudna lefedni, tehát a költsége , ha S’ költsége ennél nagyobb lenne, akkor S’ helyett S-t válaszotta volna ezen a ponton.

Az i és S választása tetszőleges, tehát egy S-beli elemeket a mohó algoritmus legfeljebb: költséggel fedi.

Azaz az egész halmazt pedig:

Ez pont annak a deficiója, hogy az közelítési faktor .

### Közelítés éles példája

értékét úgy választjuk meg, hogy elég kicsi legyen ahhoz, hogy az optimális megoldás a teljes U-t válassza, de elég nagy legyen ahhoz, hogy a mohó algoritmus ne válassza a teljes halmazt.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Válaszott részhalmaz |  |  |  |  |
| Költsége\* |  |  |  |  |
| u költsége |  |  |  |  |

A mohó algoritmus költsége épp Hr lesz, míg az optimum a 1+ lenne.

=> minden Hr-nél nagyobb számra találunk olyan kicsit -t, amire már elrontjunk a példát, tehát ennél kisebb faktorú közelítésre ez a mohó algoritmus nem képes.

\*Vegyük észre, hogy itt igazából 1/i osztva 1 (mert egy új elemet hoz be) szerepel!

## Steiner-fa

### Feladat

(Ilyen természetesen létezik, hisz összefüggő gráfnak létezik feszítőfája, ami Steiner-fa is.)

### Bonyolultság

esetén közismerten P-beli, mohó algoritmussal megoldható.

esetén a lehetséges feszítőfákat besorolhatjuk aszerint, hogy a Steiner pontok közül melyeket használja így kapunk részfeladatot (ennyir részhalmaza létezik S-nek), ezeken már elég egy egyszerű minimális feszítőfát keresni majd a részfeladatok optimumai közül kiválasztani a legjobbat. (Ez megy polinomiális időben, hisz konstans).

esetén a részfeladatok száma így összeségében még mindig polinomiáls marad a lépésszám.

Általánosságban viszont NP-teljes problémáról van szó.

### Metrikus gráf

### 2-approximáció metrikus gráfon

Algoritmus:

T-ben keresünk minimális feszítőfát (tudunk, mert G teljes).

Bizonyítás:

* + - 1. T0\* := optimális Steiner-fa
      2. T1\* := T0\* minden élet megduplázzuk ~ c(T1\*) = 2\*c(T0\*)
      3. Keres T1\* Euler-sétáját
      4. T2\* := T1\* részleges Euler bejárása, csak a T-beli csúcsokat érintve ~ c(T2\*) <= c(T1\*)

(Meg tudjuk tenni, hisz teljes a gráf, de persze itt T1\* kívűli G-beli élt is használunk, de ettől a súly csak csökkeni fog mivel metrikus a gráf!)

* + - 1. T3\* := T2\*-ból elhagyva egy élet ~ SOL <= c(T3\*) <= c(T2\*)

(Így csökken a súlya és T-beli feszítőfát kapunk, amitől a minimális feszítő fa nyilván kisebb egyenlő.)

Tehát: SOL <= c(T3\*) <= c(T2\*) <= c(T1\*) = 2\*c(T0\*) = 2\*OPT

### Éles példa 2-approximációra metrikus gráfon

Teljes, metrikus gráf, egyetlen Steiner ponttal.

T-n belül 2, T és T között 1 az élek költsége.

OPT = n-1 (Csak az S és T közötti éleket vesszük be, |T|=n-1, élek súlya 1).

SOL = 2(n-2) (Csak T közötti éleket vesszük be, ebből n-2 hosszú út elég pl, de súlyaik 2).

~ Azaz tudunk találni minden x < 2-re olyan |T|=c számot, amire a fenti példa elrontaná a x-approximációt.

### Általános Steiner-fa visszavezetése metrikus Steiner-fára

(G továbbra is egyszerű, összefüggő.)

1. Metrizál G->G’: minden él az összekötött két csúcs távolsága lesz az eredeti gráfban ~ OPT’<=OPT

Pl. Floyd algoritmusa O(n3) elvégzi a metrizálást.

(Mivel összefüggő, ezért teljes gráfot kaptunk, a háromszög egyenlőtlenség azért igaz, hisz egy-egy él egy-egy útnak felelt meg az eredeti gráfban, ha létezne a metrizált olyan út, ami egy él helyett gyorsabb lenne, akkor az eredetiben is létezne a kiválaszott legrövidebb út helyett egy rövidebb út, ami ellentmondás.)

(G’-ben minden él szerepel, ami G-ben, illetve az élek súlya csak csökkenhettek, hisz maga az él is út a két csúcs között, tehát G-ben a minimális Steiner-fa, továbbra is Steiner-fa G’-ben, de előfordulhat, hogy már nem minimális, ezért OPT’<=OPT.)

1. Keres 2-approx F1’ Steiner-fa-t G’-ben. ~ c(F1’)<=2\*OPT’
2. F1 legyen F1’ miután lecseréltünk minden élet a metrizálásnál neki értéket adó legrövidebb út éleire ~ c(F1) = c(F1’)
3. F2 legyen F1-en keresett minimális feszítő fa c(F2) <= c(F1)

(A demetrizálás során egy él többször is bekerülhet vagy feleslegessé válhat.)

Tehát:

1. SOL = c(F2) <= c(F1) <= c(F1’) <= 2\*OPT’ <= 2\*OPT

# XVIII. A Hamilton-kör probléma visszavezetése az általános utazóügynök probléma k-approximációs megoldására. Közelítő algoritmusok a metrikus utazóügynök problémára,Christofides algoritmusa.

## Általános utazó ügynök feladat

### Feladat

Adott egy *G* teljes gráf, nemnegatív élsúly függvény.

Keressük a minimális összsúlyú Hamilton-kört. A probléma NP-nehéz.

### Utazó ügynök approximáxiója

Nem ismert semmilyen k-appoximációja és nem is létezhet, hacsak nem P=NP.

### Hamilton-kör visszavezetése utazó ügynök k-approximációjára

G helyett G’ teljes gráf (|G|=n)

G’-ben a lehetséges Hamilton-körök súlyai minimálistól kezdve:

n\*1 (n db eredeti élből ⬄ létezik H-kör G-ben)

kn + (n-1)\*1 (n-1 db eredeti él, és egy darab kn)

Ha létezik H-kör G-ben, akkor létezik n\*1 súlyú H-kör G’-ben, tehát egy k-approximáció algoritmus egy maximum k\*n\*1 méretű H-kört kell megtalálnia, de k\*n\*1 <= kn + (n-1)\*1, tehát csak a n\*1 súlyú H-körök tesznek ennek eleget, ezzel viszont egy H-kört is megtalálnánk.

(Ez igazából k helyett tetszőleges f(n)-re igaz, tehát semmilyen approximációja nem létezhet ennek a problémának.)

## 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.

(Ez NP-nehéz, létező él -> 1, nem létező él -> 2 súlyozással visszavezethető Hamilton-körre.)

### ****2-approximációs algoritmus****

Algoritmus:

1. F := G minimális feszítő fája
2. E := F éleit duplázva
3. H := E-n keres Euler sétát és a mentén bejár összes csúcsot egyszer (amiben már voltunk azt átugorjuk, mivel teljes a gráf ezért mindig lesz él a következő nem bejárt csúcsba) így kapva egy H-kört

2-approximáció bizonyítása:

Minimális Hamilton-körből elhagyva egy élet egy olcsóbb Hamilton-utat kapunk, viszont a H-út egyben feszítőfa, tehát ennél olcsóbb F, ami egy minimális feszítőfa.

Az él duplázás kétszerezi a költséget, de a levágások már csak csökkentik, tehát:

c(talált H-kör) ≤ c(dupla él m feszítőfa) = 2\*c(min feszítőfa) ≤ 2\*c(H-út) < 2\*c(min H-kör) = 2\*OPT

**Christofides algoritmus**

Algoritmus:

1. F := G minimális feszítő fája
2. minTP : F páratlan fokú csúcsain egy minimális teljes párosítás (G-beli éleket használva)

(Ez azért jó, mert így a F u TP minden csúcsa páros fokú lesz és így garantáltan lesz Euler-séta benne a drága élkétszerezés nélkül is.)

1. H := F u minTP-n keres Euler sétát és a mentén bejár összes csúcsot egyszer (amiben már voltunk azt átugorjuk, mivel teljes a gráf ezért mindig lesz él a következő nem bejárt csúcsba) így kapva egy H-kört

3/2-approximáció:

A 2-approximációnál ismertett indoklás alapján analóg módon felírható:

c(talált H-kör) ≤ c(F u minTP) <= 3/2\*c(min feszítőfa) ≤ 3/2\*c(H-út) < 3/2\*c(min H-kör) = 3/2\*OPT

Csak ezt kéne belátni, hogy c(F u minTP) <= 3/2\*c(min feszítőfa), amihez elég, hogy

c(TP) <= ½\*c(min feszítőfa):

Egyrészt tudjuk: c(min feszítőfa) <= c(H-út) <= c(min H-kör)

De egy H-kört fel tudunk osztani két diszjunk TP-re úgy, hogy c(TP1) + c(TP2) <= c(min H-kör), azaz

c(TP1) <= ½\*c(min H-kör) (feltéve, hogy TP1 a kisebb költségű)

**Ezekből már összerakható, hogy**

**c(minTP) <= c(TP1) <= ½\*c(min-Hkör)**

(Azért vegyük észre, hogy itt nem a teljes gráf H-köréről van szó, hanem csak a F-ben páratlan fokú csúcsok alkotta részgráf H-köréről, ám ez a metrikusság miatt csak tovább csökkentheti a költséget, növelni nem tudja.)

# XIX. Teljesen polinomiális approximációs séma fogalma. A részösszeg probléma, bonyolultsága. Teljesen polinomiális approximációs séma a részösszeg problémára.

## Polinomiális approximációs séma

### Definició

∀ ε > 0 : ∃ (1+ ε)-approxmáció egy feladatra

Miben legyen polinomiális?

Bemenet mérete, de akkor lehet pl. O(21/ε\*n), ami ε csökkenésére (hiba csökkentésére) exponenciálisan nő, ezért bevezetendő:

### Teljesen polinomiális approximációs

Ha 1/ε-ben is polinomiáls.

n ε viszont teljesen megfelel, hiszen ε, a hiba mértéke, ha a hiba csökkentésével tudjuk exponenciális mérétkben csökkenti a lépésszámot is, akkor az csak jó.

(Pl. Eukilideszi utazó ügynökre létezik ilyen.)

## Részösszeg probléma

### Feladat

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

∃ *B*⊆*A* : Σ*bi* = t?

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

### Optimalizálási feladat

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

Find *B*⊆*A* : max { Σ*bi* } && Σ*bi* ≤ t

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

(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.

### Megoldása

Brute force: megviszgálunk minden lehetséges részösszeget

L0 = {0}

Li' = {l+ai+1 | l ∈ Li}

Li+1 = Li ∪ Li'

Kezdjük az első 0 elemből álló részösszegekből.

Ha már ismerjük a első i elemből álló résszösszegeket, akkor ezekhez hozzáadhatjuk az i-dik elemet.  
Ekkor az i elemű részösszegek és azokhoz i-dik elemet hozzávett részösszegek kiadják az első i+1 elemből képezhető részösszegeket.

(Nyesési lehetőség: ami túllépte a t-t vagy ismétlödő, azt eldobjuk.)

Ln az összes réssz összeg elő fog állni, a legnagyobb <t-t válasszuk.

(Ha az elemeket növő sorrendben rendezzük és a listákat összefésülésses rendezéssel uniozzuk, akkor Li rendezett marad így elég a végéről az elsőt kiválasztani, ami már t-től kisebb.)

A listák mérete 2|L|-re duzzadhat, ezért exponenciális lesz a lépésszám.

### Közelítő algoritmusa

A pontos megoldásban lineáris számú iteráció volt, ahol az egyes iterációk lineárisak a lista méretével és elemszámmal is.

A problémát az okozta, hogy a listák hossza exponenciális lehetett, ezért az approximációnk alapötlete, hogy ritkítsuk a lista elemeit minden iteráció után, hogy ne tudjanak nagyon elszaporodni a lehetséges részösszegek, de mindig maradjanak olyan elemek benn, akik jó közelítéssel „képviselni” tudják, ha esetleg épp az optimumot dobtuk el.

Ehhez fontos lesz, hogy a lista végig növekvő sorrendben maradjon, egy δ érték szerint ritkítunk.

Adott l elem, elhagyható minden olyan elem utána, ami kisebb egyenlő, mint (1+ δ)l.

(Balról kezdünk el elhagyni elemeket, hogy ha elhagyunk valamit, akkor az már nem tud egy másik elemet képviseiplni.)

(Természetesen az Ln listát nincs értelme ritkítani.)

Az algoritmus akkor adhat nem pontos eredményt, ha azt valamikor kitöröltük a listából ám ekkor léteznie kellett annak egy képviselőjének, ami max (1+ δ) hibát jelent, ám a képviselőt képviseltethettük egy másik elemmel, ez maximum n-szer történhetett meg, azaz a hiba maximum (1+ δ)n lehet.

### Teljesen polinomiális approximációs sémája

Azt fogjuk belátni, hogy δ=ε/2n választással a hiba faktor (1+ ε) a lépésszám pedig polinomiális n, logt (bemenet) és 1/*ε*-ben is (teljesen polinomiális).

(1+ ε) multiplikatív hiba:

Láttuk, hogy a hiba max (1+ δ)n = (1+ ε/2n)n, lássuk be, hogy ez <= 1+ ε.

Binomiális tétel

(n alatt i) felülről becsülhető ni-vel

Egyszerűsítünk ni-vel

Első tagot kifejtjük szumma elé

ε<=1, tehát ε>=εi, ε szorzót ekkor ki tudjuk hozni szumma elé

Polinomiális lista méret:

A lista első eleme 0, a következő legalább 1, ez után viszont két elem közti arány min (1+ ε/2n) (következő/előző), hisz különben ritkítottuk volna az adott elemet., így az i-dik elem legalább

(1+ ε/2n)i-2 értékű (-2: első és második elem miatt).

A t-nél nagyobbakat viszont eldobjuk, tehát ennél kisebb minden elem értéke:

Tehát ezek szerint a listában szereplő i-dik elemnek a sorszáma maximum az adott kifejezés lehet, ami viszont n-ben, lnt-ben (bemenet) és 1/*ε* polinomiális.

Lemma:

Tehát a kettő különbségének deriváltja pozitív azaz a különbségük monoton növő, ami már bizonyítja az egyenlőtlenséget.

# XX. Ütemezési feladatok típusai. Az 1|prec|Cmax és az 1||**Σ**Cj feladat. Approximációs algoritmusok a P|| Cmax feladatra: listás ütemezés tetszőleges sorrendben, éles példa tetszőleges számú gép esetére; listás ütemezés LPT sorrendben (biz. nélkül), éles példa tetszőleges számú gép esetére. Approximációs algoritmus a P|prec| Cmax feladatra (biz. nélkül), példák: az LPT sorrend, illetve a leghosszabb út szerinti ütemezés sem jobb, mint (2−1/m)-approximáció. A P|prec,pi = 1| Cmax feladat, Hu algoritmusa (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) ~ mikortól kezdve lehet
  + 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
  + precedencia: 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
  + minCmax = max(Cj): az utolsó munka befejezési ideje
  + min∑Cj ~ ∑Cj/n: a munkák átlagos befejezési ideje
  + min∑Cj-dj: a késések összege

## 1| |Cmax

Optimális megoldást ad minden olyan megoldás, melynél a gép folyamatosan dolgozik.

Ennek értéke a munka elvégzési idők összege.

## 1|prec|Cmax

A feladatokat topologikus sorrendben adogatjuk a gépnek.

Ez optimális, mert folyamatosan foglalt a gép és értéke továbbra is a munka elvégzési idők összege.

## 1||∑Cj

### SPT (Shortest Processing Time)

Munkaigény szerint növekvő sorrendben adogatjuk a feladatokat a gépnek. p1≤p2≤...≤pn.

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’i+1 befejezési ideje ugyanaz, mint pi+1, de p’i-jé csökken pi-pi+1 értékkel.)

## P||Cmax

### Bonyolultság

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

(Legyenenek a munkaidők a particiós probléma bemenetei, ha ezek optimális megoldása

Cmax <= **Σpi/2, akkor particionálható ~ lényegében itt csak egyenlőség lehet és úgy, hogy mindkét gép folyamatosan dolgozik és egyszerre végez.)**

### LS listás ütemezés (2-1/m)

Algoritmus:

A munkáknak adott egy sorrendje, ha felszabadul egy gép akkor azonnal kezdi a listában a legelső még nem elkezdett munkát.

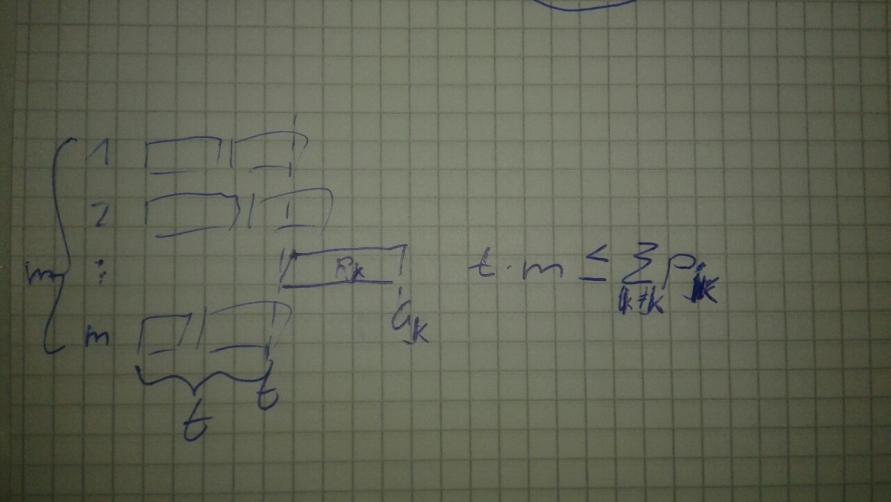
(2-1/m) approximáció:

Könnyen látható, hogy a listás ütemezés polinomidőben elkészíthető.

Az approximációs faktor igazolásához legyen:

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

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

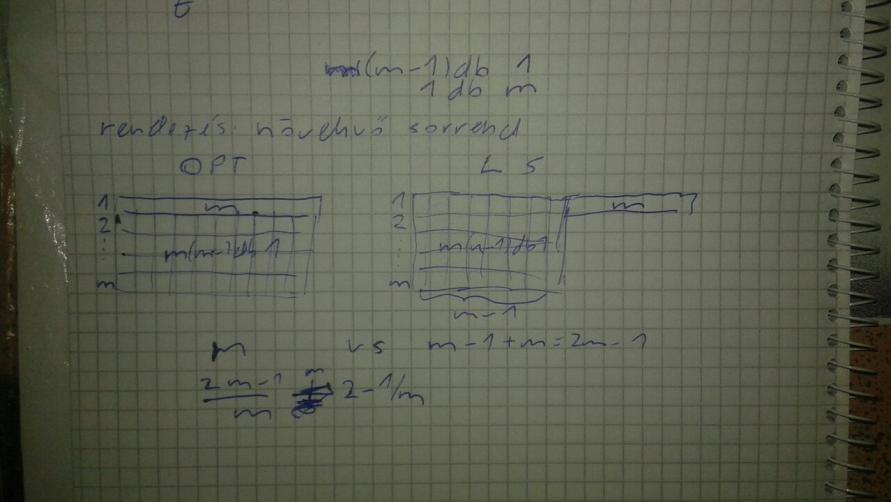


Jelölje az optimális teljes átfutási időt. Ekkor:

Ezek alapján levezethető, hogy:

Tehát sikerült igazolni az approximációs faktor helyességét.

Éles példa:



### LS-LPT (Longest Processing Time)

Algoritmus:

LS ütemezés megmunkálási idő szerint csökkenő sorrendbe rendezve.   
(Ezzel a fenti éles példa kiküszöbölése a cél: ne a végére maradjanak a nagy munkák, hisz őket nehezebb egyenletesen beosztani, előbb kezdjük el őket és a szabad helyerek „betuskoljuk” a végén a kicsi munkákat.)

Éles példa (:

2db:

2m-1, 2m-2, ... m+1-ig

3db:

m

OPT = 3m:

A 2db-sokból (m-1) db 3m párt csinál, a 3db m-t egy 3m-mé teszi össze

Így végül m db 3m-t kap, amit egy-egy gépre téve 3m alatt végez

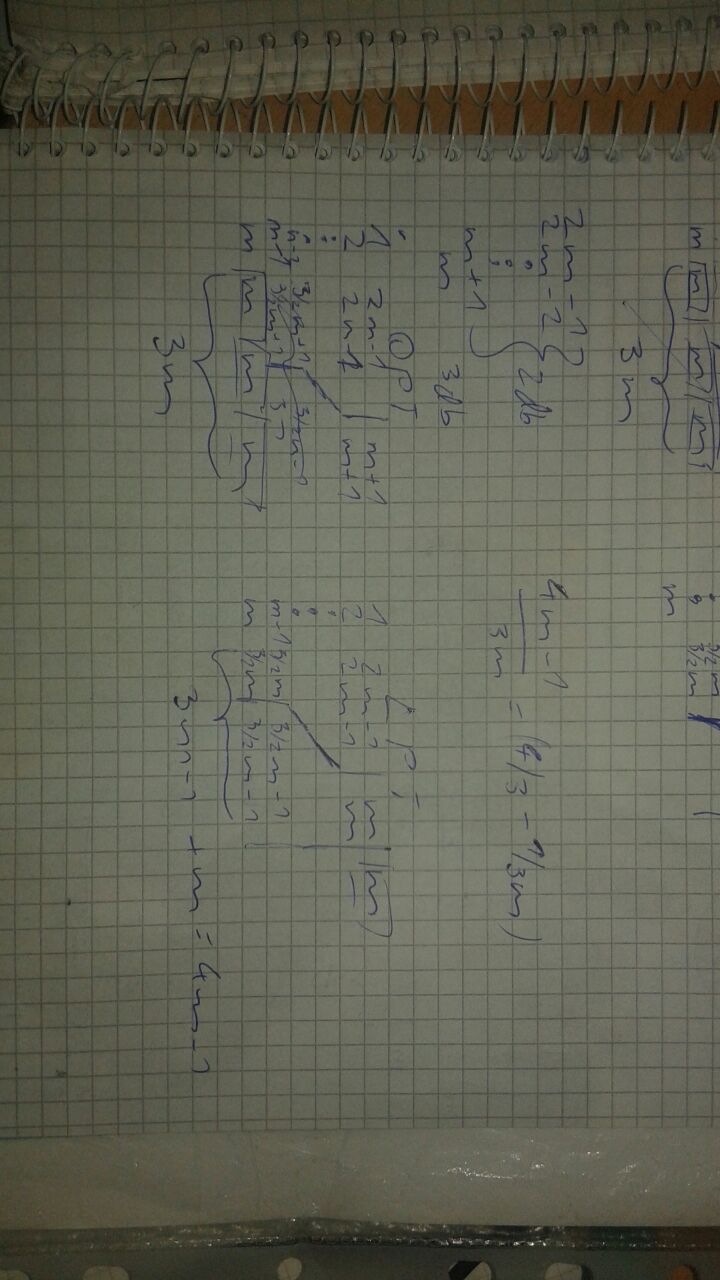
LPT = 4m-1:

Előszőr elkezdi feltölteni a gépeket fenntről lefelé a leghosszabbakkal: 2m-1,...,3/2m

(Hisz m gép van, de mindegyikből 2db van, ezért csak „-1/2m”-vel csökken a számaik)

Majd a lenntről felfelé érnek véget és tölti fel a gépeket újra a 3/2m-1,...,m-ig, de egy db m még hátra marad

Egy-egy gépen (2m-1)+(m), (2m-2)+(m+1),...,(3/2m)+(3/2m-1) feladatpárok futnak, ezek elvégézse egységesen 3m-1, de még hátra van egy m hosszú így összeségében 4m-1 lesz.



### Probléma a listás ütemezéssel

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

### 2-1/m approximáció

LS ütemezést kell alkalmazni.

### LPT éles példa

m(m-1) db 1

1 db ½

1 db m-½

Precedencia: ½ -> m-½

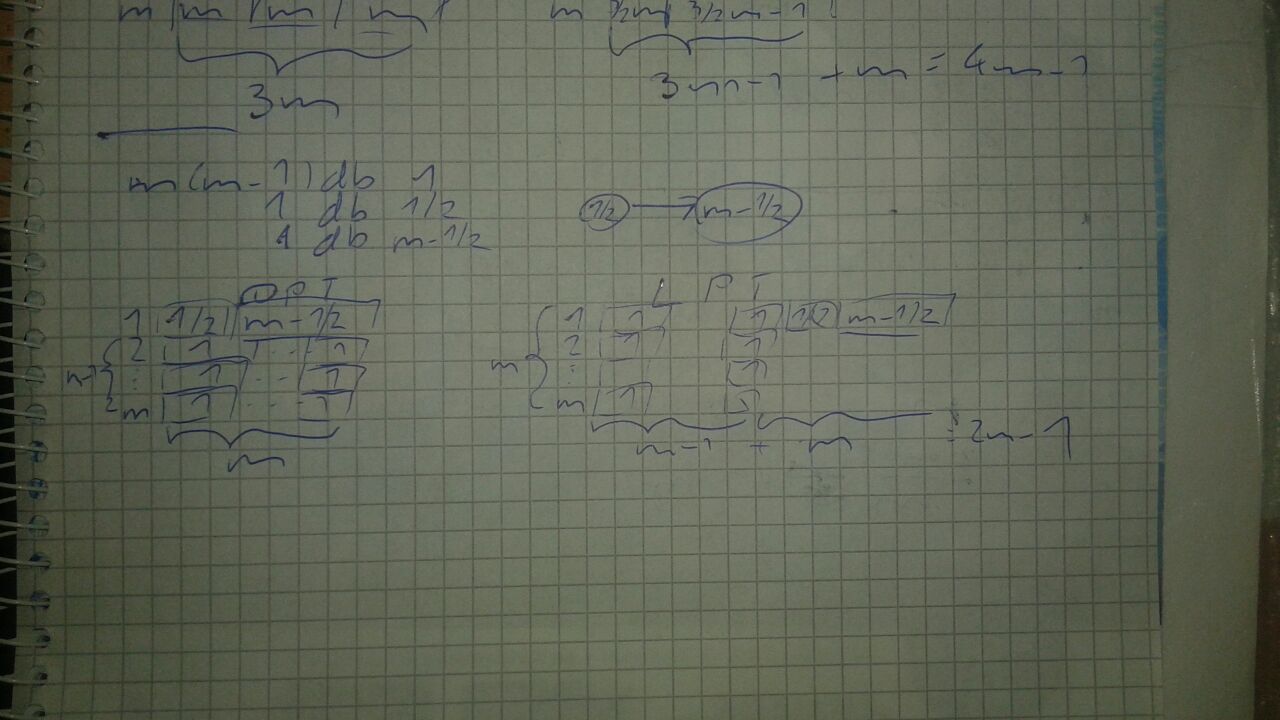
Lényegében azzal romlik el az LPT itt, hogy egy hosszabb munkát beelőzhet egy rövidebb, ha egy még rövidebb munka befejezésétől függ.

LPT 2m-1:

m(m-1)db 1 le fog futni párhuzamosan m-1 idő alatt m gépen, majd lefut a ½ -> m-½ m idő alatt.

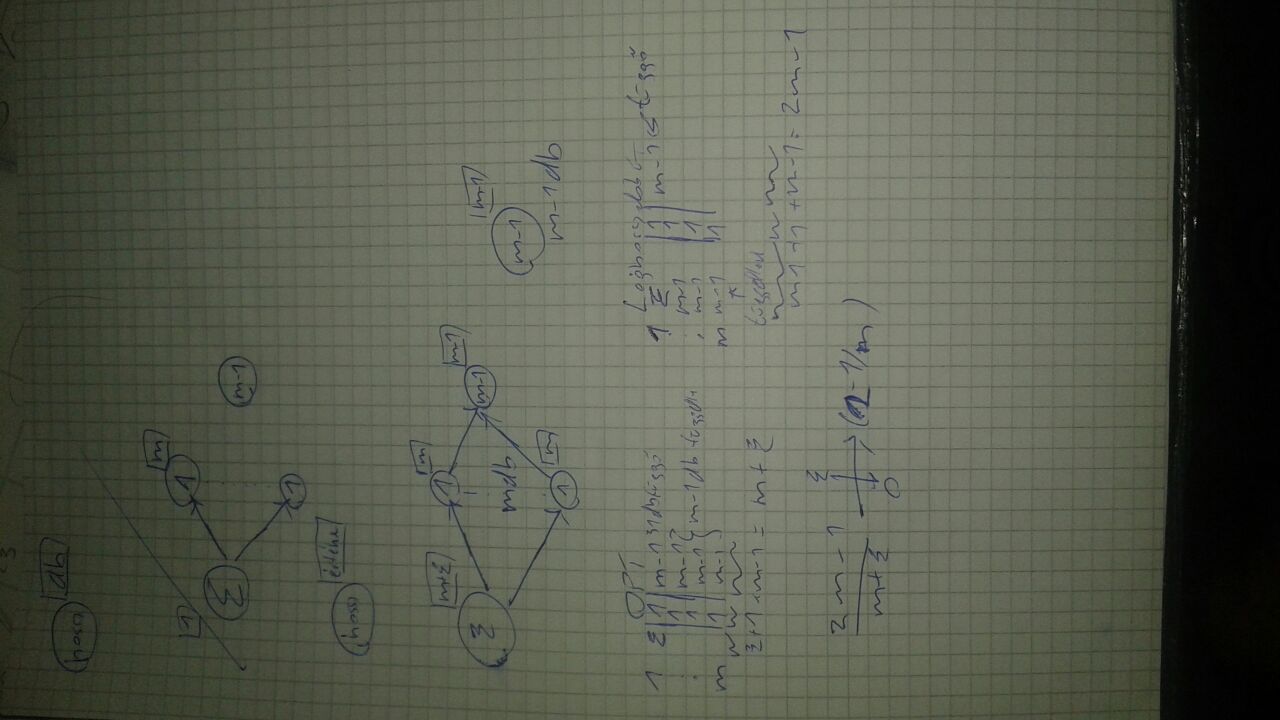
OPT m:

Egyik gép futtatja a ½ -> m-½, a másik m-1 pedig m db 1-est, így egész lefut m alatt.



### Leghosszabb út szerinti sorrend éles példa

Az LPT hibájából tanulva gondolhatnánk arra, hogy akkor a leghosszab helyett azt válasszuk, amiből a leghosszabb munkaút indul (precedencia gráfon, ami DAG, ezért pol időben lehet ezt megtalálni).



A job oldali ábra csalóka, vagy akár rossz.

Az epszillon végeztével több egyest is elkezdhet elvégezni, de az m -1 időpontig max m-2-t tud elvégezni, ezért lesz olyan 1-es, amit akkor kezd el és így a függő m-1-t csak az m-dik pillanatban tudja elkezdeni, de nem egyszerre fognak menni az 1-esek, hisz listás ütemezésnél nem lehet üresen egy gép, ha van végezhető feladat.

Tehát itt abba a csapdába esik az algoritmus hogy elkezdi futtatni a hossú m-1-ket ahelyett hogy üresen hagyná az automatákat és megvárná míg lefutna az E. Így végül az 1->m-1 hosszabb feladat száll egy rövidebb m-1 feladat szállra vár, hisz a listás ütemezés nem engedi meg azt, hogy az egyik automata üresen álljon.

## P|prec, pj=1|Cmax

### Bonyolultság

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.

### 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.

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