Számítástudomány alapjai tételek 2018/19 ősz

2018/19 VISZAA05 vizsgatematika a Számítástudomány alapjai c. tárgyhoz

Feltétlenül tudni kell a **fölkövéren** szedett fogalmakat, tételeket ill. algoritmusokat definiálni, helyesen kimondani, ill. leírni. A bekeretezett állításokat bizonyítottuk, az aláhúzottakat nem. A vizsgán az anyag értő ismeretét kérjük számon, az elégséges osztályzathoz bizonyítást nem kell tudni.

1. Leszámlálási alapfogalmak: **permutációk, variációk és kombinációk (ismétlés nélkül és ismétléssel)** példával, **kiszámításuk**, binomiális együtthatók közti összefüggések, a binomiális tétel.
2. Gráfelméleti alapfogalmak: **pont, él, fokszám**. Egyszerű gráf, részgráf, feszített részgráf, izomorfia, élsorozat, séta, út, kör, **összefüggő gráf**, komponens. **Gráfok fokszámösszege**, erdő, **fa**, fák egyszerűbb tulajdonságai: két elsőfokú pont, fák (erdők) élszáma.
3. **Feszítőfa** létezése, **minimális költségű feszítőfa, Kruskal algoritmusa**, ennek helyessége.
4. Legrövidebb utakat kereső algoritmusok (**BFS, Dijkstra**, Ford, Floyd), ezen algoritmusok helyessége, legrövidebb utak fája Bejárásokkal kapcsolatos fogalmak: bejárási fa, faél, előreél, visszaél, keresztél. **Legszélesebb utak** keresése irányítatlan gráfban: Módosított Kruskal algoritmus, helyessége.
5. **Mélységi keresés** és alkalmazásai (élek osztályozása, mélységi számozás, befejezési számozás, fa-, előre-, vissza- és keresztélek, irányított kör létezésének eldöntése DFS-sel), alapkörrendszer. Aciklikus (irányított kört nem tartalmazó) irányított gráfok (DAG-ok), jellemzésük a topologikus sorrenddel, topologikus sorrend keresése, **PERT-módszer**, kritikus utak és tevékenységek.
6. **Euler-séta és körséta**, létezésének szükséges és elégséges feltétele. **Hamilton-kör és út** létezésére szükséges, ill. elégséges feltételek: komponensszám ponttörlések után ill. Dirac, Ore tételei.
7. **Gráfszínezés, kromatikus szám, klikkszám**, alsó és felső korlát a kromatikus számra. Síkgráfok kromatikus száma: négyszíntétel, ötszíntétel.
8. Hálózati folyamok: **hálózat, folyam, folyamnagyság (avagy folyamérték)**, st**-vágás**, st**-vágás kapacitása**. **Ford-Fulkerson tétel**, javító utas algoritmus (előre- és visszaélek). EgEr lemma, Edmonds-Karp tétel, illusztráció. Általánosított hálózatok visszavezetése szokásos hálózatra.
9. **Páros gráfok**, definíciók ekvivalenciája. **Párosítások** (páros és nem páros gráfban), teljes párosítás, adott ponthalmazt fedő párosítás, Hall, Frobenius és König tételei, alternáló utas algoritmus maximális párosítás keresésére. **Lefogó és független pont- ill. élhalmazok**, az **ezekből származó gráfparaméterek** (τ, α, ρ, ν), triviális egyenlőtlenségek, **Gallai két tétele**.
10. **Síkbarajzolhatóság**, gömbre rajzolhatóság, tartomány, sztereografikus projekció. Külső tartomány nem kitüntetett volta. Az **Euler-féle poliédertétel** és következményei: egyszerű, síkbarajzolható gráfokon felső korlát az élszámra. **Kuratowski gráfok**, síkbarajzolhatósága, **soros bővítés**, **Kuratowski-tétel** könnyű iránya.
11. **Oszthatóság, legnagyobb közös osztó**, **euklideszi algoritmus**, prímek és felbonthatatlan számok, a számelmélet alaptétele, **kanonikus alak**, osztó, lnko kanonikus alakja, osztók száma, nevezetes tételek prímszámokról: prímek száma, prímek közti hézag, prímszámtétel.
12. **Kongruencia fogalma**, műveletek kongruenciákkal. Teljes és redukált maradékrendszer, az **Euler-féle** φ**-függvény**, $\varphi(n)$ kiszámítása. Az **Euler-Fermat tétel** és a kis Fermat-tétel. **Lineáris kongruenciák** megoldhatósága és konkrét módszer a megoldásra.
13. Algoritmusok bonyolultsága (inputméret, lépésszám az inputméret függvényében, polinomidejű algoritmus), döntési problémák. **P, NP, co-NP** bonyolultsági osztályok, feltételezett viszonyuk, példa ilyen problémákra. Polinomiális visszavezethetőség (Karp-redukció), **NP-teljesség**, Cook-Levin tétel, nevezetes NP-teljes problémák: **SAT**, **HAM**, **3-SZÍN**, k-**SZÍN**, **MAXFTN**, **MAXKLIKK**.
14. Számelméleti algoritmusok: alapműveletek, (modulo m) hatványozás és az euklideszi algoritmus lépésszáma. Prímtesztelés, **Fermat-teszt**. **Nyilvános kulcsú titkosírás, digitális aláírás**. Az RSA titkosítási módszer (Az üzenetből számok képzése, p és q prímek generálása, n, m kiszámítása, e és d választása, titkos és nyílt adatok, kódoló és dekódoló függvények, dekódolás működik).

A tételek a wiki-n található Fleiner Tamás jegyzet, a 2013-as Eke Máté, valamint a 2017-es Molnár Martin-féle ZH kidolgozásokon alapulnak.

1. tétel: Leszámlálási alapfogalmak

	Permutáció	Variáció	Kombináció
ismétlés nélkül	$n!$	$\frac{n!}{(n-k)!}$	$\binom{n}{k}$
ismétléssel	$\frac{n!}{k_1! * k_2! * \dots * k_m!}$	n^k	$\binom{n+k-1}{k}$

Permutáció: n elem sorba rendezése ismétléssel vagy ismétlés nélkül.
Ismétlés nélküli példa: 5 ember hány féleképpen állhat sorba.
Ismétléses példa: Ha egy héten egy tanórából több is van, akkor a lehetséges heti órarendek száma.

Variáció: n elemből k kiválasztása és a k elem sorba rendezése ismétléssel vagy ismétlés nélkül.
Ismétlés nélküli példa: n versenyzőből, hány féle lehet az első k.
Ismétléses példa: versenynapon n versenyző, k résztáv, hány féle lehet a résztáv győztesek sorrendje.

Kombináció: n elemből k elem kiválasztása úgy, hogy nem számít a sorrend.
Ismétlés nélküli példa: Lottó.
Ismétléses példa: Cukrászdában n féle sütemény van, mindegyikből végtelen áll rendelkezésre, és ezekből akarunk k darabot kiválasztani.

$$\binom{n}{k} = \frac{n!}{(n-k)! * k!}$$

$\binom{n}{k} = 0$, ha n<k vagy k<0

Binomiális tétel:

$$(a+b)^n = \sum_{k=0}^{n} \binom{n}{k} a^k b^{n-k}$$

Binomiális együtthatók közötti összefüggések:

- $\binom{n}{k} = \binom{n}{n-k}$
- $\binom{n-1}{k-1} + \binom{n-1}{k} = \binom{n}{k}$
- $\binom{n}{0} + \binom{n}{1} + \binom{n}{2} + \dots + \binom{n}{n-1} + \binom{n}{n} = \sum_{i=0}^{n} \binom{n}{i} = 2^n$ (azaz az n elemű halmaznak 2^n részhalmaza van)
- $\binom{n}{0} = \binom{n}{n} = 1$
- $\binom{n}{k} = 0$, ha $k > n$

Alternáló összeg

$$\sum_{k=0}^{n} (-1)^k \binom{n}{k} = \binom{n}{0} - \binom{n}{1} + \binom{n}{2} - \dots = 0 \text{ minden } n > 0.$$

Vandermonde-azonosság

$$\sum_{j=0}^{k} \binom{m}{j} \binom{n}{k-j} = \binom{m+n}{k}.$$

2. tétel: Gráfelméleti alapfogalmak

Gráf definíciója: G=(V,E) G gráf V(G) jelöli a G pontjainak, E(G) pedig G éleinek a halmazát. (V nem üres halmaz, egyszerű gráf esetén E(G) eleme V(G) kételemű részhalmazainak)

Fokszám: G gráf v csúcsának fokszáma – d(v) – a G v végpontú éleinek száma (hurokél kétszer számít). **ΔG** a G maximális fokszáma.

Pont:

- **Izolált pont:** d(v)=0. Az izolált pont önmagában egy komponens.

Él:

- **Hurokél:** a kiindulási és a végcsúcs azonos.
- **Párhuzamos él:** a két él ugyan azt a két csúcsot köti össze.

Tétel: Gráfok fokszámösszege: Ha G véges, akkor $\sum_{v \in V(G)} d(v) = 2 * |E(G)|$ vϵV(G), **azaz a fokszámok összege az élek számának kétszerese.**

G k-reguláris ha minden vϵV(G)-re d(v)=k.

G egyszerű gráf, ha:

(1) nem üres.
(2) E(G) elemei V(G) valamilyen kételemű részhalmazai.
(3) nem tartalmaz hurok- és párhuzamos éleket.

Nevezetes egyszerű gráfok:

- Kn n pontú teljes gráf. Minden pont minden ponttal szomszédos.
- Cn n pontú kör. Minden pont pontosan két másik ponttal szomszédos.
- Pn n pontú út. A végpontok kivételével minden pont pontosan két másik ponttal szomszédos.

Részgráf: H a G részgráfja, ha H él-, és csúcstörléssel kapható G-ből.
Feszített részgráf: H a G feszített részgráfja, ha H csúcstörléssel kapható G-ből.
Feszítő részgráf: H a G feszítő részgráfja, ha H éltörléssel kapható G-ből.

Izomorfia: két gráf izomorf, ha csúcsaik között létezik olyan kölcsönösen egyértelmű megfeleltetés, amelyre minden él helyesen van megfeleltetve.

Élsorozat: csúcsok és élek felváltva követik egymást, csúccsal kezdve és befejezve.
Séta: olyan élsorozat, aminek minden éle különböző.
Út: olyan séta, aminek minden pontja különböző.
Kör: olyan út, aminek a kezdő és a végpontja azonos.

Összefüggő gráf: ha bármely két pontja között vezet élsorozat = vezet séta = vezet út.
Komponens: K ⊆ V(G), ha bármely u, v ∈ K között vezet út, de u ∈ K és v ∈ (V(G)\K) között nem. *Bármely gráf egyértelműen felbontható komponensek uniójára.*
Komplementer gráf: A G egyszerű gráf komplementere a $\bar{G}$:= (V(G), $\binom{V}{2}$\ E(G)) gráf.
Irányított gráf: D(V,A), ha V nem üres halmaz, és A részhalmaza $V^2 - nek$.

G gráf erdő, ha körmentes.
G gráf fa, ha körmentes és összefüggő. *Egy erdő minden komponense fa.*
Levél: v levél, ha F fa része és d(v)=1.
Tétel: Ha egy fa legalább két csúcsú, akkor a fának van legalább két elsőfokú csúcsa.
Tétel: Ha G n pontú körmentes gráf, G pontosan akkor összefüggő, ha |E(G)|=n-1, illetve ha G egy n pontú és k komponensű erdő, akkor |E(G)| = n-k.

3. tétel: Feszítőfa létezése, minimális költségű feszítőfa, Kruskal algoritmusa és annak helyessége

F gráf fa, ha körmentes és összefüggő.
Feszítő részgráf: H a G feszítő részgráfja, ha H éltörléssel kapható G-ből.
F a G feszítőfája, ha F fa, és F a G feszítő részgráfja.
Ha G-nek létezik feszítőfája, akkor G összefüggő.

Tétel: G gráf akkor és csakis akkor összefüggő, ha G-nek létezik feszítőfája.

Költségfüggvény: k: $E \rightarrow R$+ minden élhez egy költséget rendel.
Minimális költségű feszítőfa: olyan $F \in E$ élhalmaz, amire (V,F) fa, és k(F) minimális.

Kruskal algoritmus:

- *Input:* G=(V,E) gráf, és k: $E \rightarrow R$+ költségfüggvény.
- *Output:* minimális költségű feszítőfa.
- *Működés:* minden lépésben megépítjük a legolcsóbb élt, ami nem hoz létre kört. Mohó algoritmus, mert csak azzal törődik, ami éppen a legalacsonyabb költségű. Az így keletkezett fa a G gráf minimális költségű (súlyú) feszítőfája.
- Helyességének bizonyítása: tegyük fel, hogy az algoritmus helytelen, ekkor létezik egy f él, amit e helyére bevéve olcsóbb feszítőfát kapunk. Ekkor azonban f költsége kisebb, mint e költsége, így f-et az algoritmussal korábban már ellenőriztük, tehát ellentmondásra jutottunk, azaz a feszítőfa minimális költségű.

4. tétel: Legrövidebb utakat kereső algoritmusok, bejárásokkal kapcsolatos fogalmak, legszélesebb utak keresése irányítatlan gráfban

Bejárással kapcsolatos fogalmak:
Bejárás: egy G gráf csúcsainak valamely sorrendben történő végiglátogatása.
Bejárási fa: olyan irányított fákból álló erdő, melyek élei a gyökérponttól kifele irányítottak.
Elérési sorrend: gráfbejárás után G csúcsainak olyan sorrendje, ami megadja mikor értük el először az adott csúcsot.
Befejezési sorrend: gráfbejárás után G csúcsainak olyan sorrendje, ami megadja mikor foglalkoztunk utoljára az adott csúccsal.
Faél: ha az uv él a bejárási fának egy éle.
Előreél: ha u-ból v-be irányított út vezet.
Visszaél: ha v-ből u-ba irányított út vezet.
Keresztél: ha u és v között nincs irányított út a bejárási fában.
Irányítatlan gráf esetén előreél és visszaél ugyanazt jelenti.

Legrövidebb út: u-ból v-be vezető utak közül a legkevesebb éllel rendelkező út.
Legrövidebb utakat kereső algoritmusok:

BFS (Breadth First Search) bejárás: szélességi bejárás. Soron következő csúcsot mindig a legkorábban meglátogatottból látogatja meg. Az így kapott szélességi fa, a legrövidebb utak fája.
Működés: G gráfban kiindulunk tetszőleges (A) pontból. Ha A-ból vezet él még eléretlen csúcsba, akkor azt behúzzuk; ezek a csúcsok elérté válnak. Ha A-ból nem tudunk több eléretlen csúcsba menni, akkor A befejezetté válik. Ekkor az A után először elért csúcsból folytatjuk ugyanezt. Több eléretlen csúcs esetén abc-sorrendben választunk eléretlen csúcsot. **A bejárás után elérési és befejezési sorrendeket kapunk (amik egyébként megegyeznek).** A BFS bejárás fáját úgy kapjuk meg, hogy minden elért csúcsra felírjuk melyik él által értük el. Ezek a faélek alkotják a BFS-fát. **BFS bejárás után csak fa-, kereszt-, esetleg visszaél lehet, előreél nincsen**; át kellene ugrania egy faélt. A BFS-fa egy gyökérpontból minden más csúcsba az egyik legrövidebb utakat tartalmazza, a BFS (szélességi keresés = Breadth First Search) által megkapott BFS-fa a legrövidebb utak fája. Lépésszáma legfeljebb konstans*(n+m), ahol n a csúcsok, m az élek száma.
Tétel: a BFS által kapott szélességi fában u-ból v-be irányított út vezet, akkor ez az út egyben az (egyik) legrövidebb u,v út is.

Hosszfüggvény: G=(E,V) gráf, l: $E \rightarrow R$ egy hosszfüggvény, ami minden élhez egy hosszt rendel. (A hosszfüggvény konzervatív, ha nincs negatív köre.)

Dijkstra algoritmus:

- *Input:* G=(V,E) gráf, és l: $E \rightarrow R$+ nem negatív hosszfüggvény, és $u \in V$ pont.
- *Output:* a dist(u,-) hosszfüggvény, azaz **G minden v csúcsára meghatározunk egy legrövidebb u,v utat.**
- *Működés:* Irányított G gráfban r gyökérpontból indulva szeretnénk elérni az összes többi csúcsot a lehető legrövidebb úton. G gráfon legyen értelmezett egy hosszfüggvény (l), amely minden élre egy l hosszt határoz meg. A kiindulási pont (gyökérpont) távolsága legyen 0, a többi (még eléretlen) csúcsé kezdetben végtelen (mert nem értük el). Sorban feljegyezzük, hogy melyik csúcsot milyen távolságban (vagy milyen költség árán) érjük el. Az A-ból vezető éleket kezdetben mindenképpen behúzzuk a csúcsokba. Ezen csúcsokhoz feljegyezzük, hogy az A-ból értük el őket, és azt is, hogy milyen messze vannak tőle. Ekkor A-ból már nem tudunk mit csinálni, befejezzük. Az elért csúcsok közül a legolcsóbbat (legközelebbit) kiválasztjuk, és megnézzük hova vezet él. Abban az esetben, ha olyan csúcsba vezet él, melyet már A-ból elértünk, megpróbálunk élmenti javítást végezni. Ha C pontot A-ból 90 messze tudtunk elérni, de B távolsága A-tól 30, illetve C 40 messze van B-től, akkor a C-be vezető teljes út hossza így csak 70 lesz, ha B-n keresztül érjük azt el. Ezt az élmenti javítást minden lehetséges esetben megtesszük az algoritmus futtatásakor, és az előzőekhez hasonlóan feljegyezzük, hogy milyen csúcson keresztül értük el, és mennyi összköltség fejében. Az algoritmust mindig a lehető legolcsóbban elért csúcsból folytatjuk, és minden iterációban felírjuk egymás alá sorokban az összes csúcs adatát, állapotát. **Ha helyesen futtatjuk a Dijkstra algoritmust, akkor végül egy élsúlyozott legrövidebb utak fáját kapunk** (ezen éleken keresztül állítottuk be a végső költségeket vagy távolságokat). Lépésszáma legfeljebb konstans*n^2 .

Ford algoritmus:

- *Input:* G=(V,E) gráf, l: $E \rightarrow R$ konzervatív hosszfüggvény, és $u \in V$ pont.
- *Output:* a dist(u,-) hosszfüggvény, azaz G minden v csúcsára dist(u,v).
- *Működés:* Szinte megegyezik a Dijkstra algoritmussal, de megengedünk negatív élsúlyokat is. A feltétel mindössze annyi, hogy negatív összsúlyú kör nem lehet G gráfban. Kezdetben $d(u0) := 0$, $ill.\ d(v) := \infty$. Az algoritmus fázisokból áll, és minden fázisban megpróbálunk sorra $e1$, $e2$… éleken javítani. Ha egy fázisban nem sikerült javítani, akkor az algoritmus véget ér. Lépésszám: mivel egy útnak legfeljebb n-1 éle lehet, ezért legkésőbb az (n-1). lépésre az algoritmus végez. Mivel minden fázisban körülbelül élszámnyi lépést végez, így a lépésszám konstans* $n * m$-el becsülhető. **Az algoritmus segítségével ellenőrizhető, hogy létezik-e negatív kör a gráfban.**

Floyd algoritmus:

- *Input:* G=(V,E) irányított gráf, l: $E \rightarrow R$ konzervatív hosszfüggvény.
- *Output:* dist(u, v) minden u, v-re.
- *Működés:* ez az algoritmus inputként egy irányított G gráfot kap, outputként pedig meghatározza bármely két csúcs közti út hosszát. Alkalmazásakor egy gráf csúcsaiból mátrixot alkotunk (n darab csúcsból n*n-es mátrix); a szimmetriatengely természetesen mindenhol 0, az egymásból közvetlen elért csúcsok közti hosszokat beírjuk, míg a többi helyre végtelen kerül. Ez a nulladik lépés. Utána az első lépés, hogy az első oszlopot és sort letakarjuk a mátrixban, úgymond ezek érintetlenek maradnak, ahogy a szimmetriatengely is. A változatlanul leírt értékek metszéspontjai az összegek lesznek, és ha az összeg kisebb, mint az amúgy ott lévő, akkor lecseréljük. A lépésszám maximum konstans*n^3.

Legszélesebb utak: Legyen G = (V, E) egy irányított vagy irányítatlan gráf, és legyen w: $E \rightarrow R$+ az egyes élek „szélességét” leíró függvény. Ha P a G egy útja, akkor w szélessége a P legkeskenyebb élének szélessége: w(P) := min{w(e): e∈E(P)}.
Szélességfüggvény: w: $E \rightarrow R$+ az egyes élek „szélességét” leíró függvény. Ha P G útja, akkor P szélességét a „legkeskenyebb” útja határozza meg.
Mohó algoritmus a legszélesebb utak keresésére: Megépítjük a legszélesebb utat, ami nem hoz létre kört. Ekkor egy legszélesebb utakat tartalmazó feszítőfát kapunk. A feszítőfa G bármely két csúcsa között G egy legszélesebb útját tartalmazza.
Az algoritmus helyességének bizonyítása: Tegyük fel, hogy létezik szélesebb út u-v között. Hagyjuk el a legkeskenyebb e élt, ekkor a keletkező két komponens közt létezik szélesebb f él, de ezt az algoritmussal korábban már ellenőriztük. Ellentmondásra jutottunk.

5. tétel: Mélységi keresés és alkalmazásai, alapkörrendszer, DAG-ok, topologikus sorrend, PERT-módszer

Mélységi bejárás (DFS, Depth First Search): olyan bejárás, ahol egy v_0 gyökércsúcsból indulunk, és mindig a legutóbb elért v_i csúcsból egy még bejáratlan v_{i+1} csúcsba igyekszünk egy gráfél mentén. Ha ez nem sikerül, akkor visszalépünk egy v_j csúcsba, ahonnan v_i -t elértük, és innen próbálunk bejáratlan csúcsba lépni. Ha visszajutunk v_0-ba, és nem tudunk tovább menni, de még nem jártunk be minden csúcsot, akkor új gyökérpontot választunk.
Működés: G=(E,V) gráfban legyen értelmezett l hosszfüggvény, és keressük a gráfban a leghosszabb utat. A DFS (Depth First Search) bejárás magyarul mélységi keresést vagy bejárást jelent. A bejárás során a következő csúcsot mindig a lehető legkésőbb elért csúcsból érjük el (abc sorrend számíthat). Outputként elérési és befejezési sorrendet kapunk. Ha egy csúcsból nem tudok tovább menni (nem tudok még el nem ért élt elérni), akkor a csúcs befejezetté válik, és az előzőből próbálkozunk. **Aciklikus gráfnál DFS futtatása által kapott befejezési sorrend egy fordított topológiai sorrend. DFS után irányítatlan gráfban nem lehet keresztél.** Az algoritmus lépésszáma legfeljebb konstans*(n+m).

A mélységi bejárás bejárási erdőjét **mélységi fának** nevezzük.
A **mélységi számozás** a mélységi bejárás elérési sorrendje.
Faél, vagy előreél: az az uv él, ahol v mélységi száma nagyobb, mint u-é. v az u közvetlen leszármazottja.
Visszaél uv él, ha u mélységi száma nagyobb, mint v-é, és u a v leszármazottja.
Keresztél, ha u mélységi száma nagyobb, mint v-é, és u és v nem leszármazottai egymásnak.

G (irányított) gráf aciklikus gráf **(DAG: directed acyclic graph)**, ha nincsen benne (irányított) kör. Ezzel egyenértékűek: G DAG; G DAG csúcsainak létezik topologikus sorrendje; DFS után DAG-ban nincsen visszaél.

Tétel: Ha G irányított gráf, akkor az alábbi négy állítás ekvivalens:

1. G aciklikus.
2. G egyetlen mélységi bejárásában sincs visszaél.
3. G valamely mélységi bejárásában nincs visszaél.
4. G csúcsai sorba rendezhetőek úgy, hogy G minden éle egy sorrendben későbbi csúcsba mutat – ez a sorrend a **topologikus sorrend.**

Tétel: Ha F a G feszítőfája, akkor minden e∈E(G) élhez, ami nincs benne a feszítőfában, tartozik egy egyértelmű **alapkör**, ez pedig az $F \cup e$ egyetlen köre.

Ha F a G irányítatlan gráf feszítőfája, akkor F-hez tartozik egy **fundamentális körrendszer**: a G\F éleihez tartozó alapkörök.

Ha F a G irányítatlan gráf feszítőfája, akkor F minden éle meghatározza a G csúcsainak egy két részre osztását. Ezen felosztások alkotják a **fundamentális vágásrendszert**.

PERT-módszer (Project Evaluation and Review Technique): a probléma lényege egy F feladat optimális ütemezése. A szabályok mindegyike olyan alakú, hogy egy v tevékenységet nem kezdhetünk meg hamarabb u tevékenység után c(uv) időnél. Definiálható F-hez egy P(F) irányított gráf, aminek a csúcsai a tevékenységek. Minden szabálynak megfelel P(F) egy súlyozott, irányított éle. **P(F) aciklikus.** F feladat k ütemezése abból áll, hogy F minden tevékenységéhez úgy jelölünk ki egy-egy kezdési időpontot, hogy minden vonatkozó szabályt betartunk. Tehát minden v tevékenységhez úgy jelölünk ki egy k(v) kezdési időpontot, hogy minden uv élre teljesüljön $k(v) \geq k(u) + c(uv)$. Az F feladat k ütemezés szerinti elvégzéséhez szükséges idő k(t). Optimális esetben k(t) minimális. Az F feladat h(F) hossza, az F végrehajtásához szükséges idő, azaz h(F)=k(t), ha k(t) minimális. Egy u tevékenységet **kritikus tevékenységnek** nevezünk, ha u kezdési időpontja nem függ az optimális ütemezéstől. PERT célja egy optimális ütemezés kiszámítása, és a kritikus tevékenységek meghatározása.

vagy

A projekt menedzselési probléma adott tevékenységek által alkotott projekt ütemezését segít meghatározni. Egy G gráfban határozzunk meg egy topológiai sorrendet (DFS-sel a befejezési sorrend megfordítva vagy források folyamatos elhagyásával), majd rajzoljuk újra a gráfot a topológiai sorrend segítségével. Ezután határozzuk meg a legkorábbi kezdési időpontokat úgy, hogy az előzőt (élen futó érték) mindig hozzáadjuk az aktuálishoz; több lehetséges összeg esetén mindig a maximálisat vegyük. Ezután a projekt utolsó tevékenységétől visszafele haladva megnézem, hogy minek a hatására kapta a kezdési időpontját. Ezeket az éleket bejelölve megkapom a kritikus utat, amely a nyelőbe vezető leghosszabb él. Az ezen a kritikus úton elhelyezkedő tevékenységeket kritikus tevékenységeknek nevezzük: ezek csúszása az egész projekt csúszásához vezetnek. A nem kritikus úton lévő tevénységek is okozhatnak csúszást, ha a késedelem elég nagy.

Tétel: a fentiek szerint megadott F feladat hossza megegyezik a P(F) gráfban az irányított st-utak maximumával.
Egy u **tevékenység pontosan akkor kritikus**, ha létezik u-n keresztül irányított, h(F) súlyú st-út.
Kritikus útnak a P(F) h(F) súlyú irányított st-útját nevezzük.

6. tétel: Euler-séta és körséta, Hamilton-kör és út, komponensszám ponttörlések után, Dirac- és Ore-tétel

Euler-(kör)séta: egy G=(E,V) irányított gráfnak létezik Euler-körsétája, ha van G-ben olyan séta (körséta), amely G minden élét pontosan egyszer tartalmazza. Euler-körsétáról beszélhetünk, ha a séta kezdő és végpontja azonos; Euler-sétáról ha ezek különbözőek.

Ha G-nek van Euler-körsétája, akkor G az izolált pontoktól eltekintve összefüggő.
Ha G-nek van Euler-sétája, akkor lehet benne Euler-körséta.

Euler-körséta pontosan akkor (szükséges és elégséges feltétel) létezik G-ben, ha:

(1) G izolált pontjaitól eltekintve összefüggő, és
(2) G-ben minden fokszám páros.

Euler-séta pontosan akkor létezik G-ben, ha:

(1) G izolált pontjaitól eltekintve összefüggő, és
(2) legfeljebb 2 páratlan fokú csúcs van (ezek a kezdő és a végpont, és csak 1 páratlan nem lehet).

Ha a véges, irányított G gráfnak létezik Euler-körsétája, akkor G minden csúcsának ugyanannyi a befoka mint a kifoka, azaz tetszőleges v csúcsra igaz, hogy a v-be befutó élek száma megegyezik a v-ből kiinduló élek számával. Ha Euler-sétája van G-nek, akkor lehet két kivételes csúcs: az egyikben a befok egyel több a kifoknál, a másiknál a kifok nagyobb a befoknál eggyel.

Euler-körsétát úgy csinálhatunk egyszerűen, ha egy irányított G gráfot éldiszjunkt körökre bontunk fel úgy, hogy minden csúcsban összekötünk tetszőleges élpárokat. Ha nem egy körsétát kaptunk, akkor egy csúcsban összefűzzük őket, és mindezeket addig ismételjük, míg pontosan egy körsétát nem kapunk.

Hamilton-kör: G gráf olyan köre, ami G minden pontját tartalmazza.
Hamilton-út: G gráf olyan útja, ami G minden pontját pontosan egyszer tartalmazza.

Szükséges (de nem elégséges) feltétel: ha G véges gráfban létezik Hamilton-kör (vagy út), akkor G-ből k tetszőleges pontot törölve, a keletkező gráfnak legfeljebb k (vagy k+1) komponense van.

Elégséges feltételek:

(1) **Dirac-tétel:** $n \geq 3$ csúcsú G egyszerű gráf esetén ha minden csúcs fokszáma legalább n/2, akkor G-nek létezik Hamilton-köre.
(2) **Ore-tétel:** ha $n \geq 3$ pontú, G egyszerű gráf minden két, nem szomszédos u,v pontjára igaz, hogy $d(u) + d(v) \geq n$, akkor G-nek van Hamilton-köre.

A Dirac-feltételből következik > az Ore-feltétel, **az Ore-tételből következik > a Dirac-tétel.**

7. tétel: Gráfszínezés, kromatikus szám, klikkszám, síkgráfok kromatikus száma, 4-szín és 5-szín tételek

Egy G gráf **k-színezése** a csúcsok k db színnel történő olyan kiszínezése, hogy utána minden él különböző színű csúcsok között fut.
k-színezhető egy G gráf, ha minden csúcs kiszínezhető k adott szín valamelyikére, úgy hogy minden szomszédos csúcs különböző színű.
Kromatikus szám χ(G): χ(G)=k, ha G k-színezhető, de nem (k-1)-színezhető.
Egy színezés után az azonos színűre színezett csúcsok halmazt alkotnak; ez a **színosztály**.
Megfigyelések:

1. Ha G tartalmaz hurokélt, akkor nem színezhető.
2. G gráf színosztályának (adott színezéshez tartozó) csúcsai között nem fut él.

A G gráf csúcsainak U részhalmaza **független**, ha G-nek nincs olyan éle, melynek mindkét végpontja U-beli.
Klikk: a G gráf teljes részgráfja.
Klikkszám $\omega(G)$: a G legnagyobb klikkjének pontszáma, azaz $\omega(G) = k$, ha G-ben van k méretű, de nincs k+1 méretű klikk.

Minden irányítatlan, véges G gráfra teljesül, hogy a **kromatikus szám alsó becslése** a maximális klikkméret: ω(G)≤χ(G).
Minden irányítatlan, véges G gráfra áll, hogy a **kromatikus szám felső becslése** a maximális fokszám+1: χ(G)≤Δ(G)+1. **Ezen két állításból: ω(G)≤χ(G)≤Δ(G)+1.**

Nevezetes kromatikus számok:

1. n pontú teljes gráfra: χ(K_n) = n és ω(K_n) = n
2. n pontú körre: χ(C_n) = ha páros: 2; ha páratlan: 3, illetve ω(C_n) = ha n>3 akkor 2; n=3-ra 3

Brooks-tétel: G véges, egyszerű, összefüggő. Ha G nem teljes, és nem páratlan kör, akkor χ(G) ≤ Δ(G).

G gráf páros, ha 2-színezhető. Másképpen: χ(G) maximum 2, tehát G csúcsai két halmazra (színosztályra) bonthatók fel úgy, hogy minden él a két halmaz között fut, de egy halmaz két pontja között sosem fut él.

k-élszínezhető egy G gráf, ha minden él kiszínezhető k adott szín valamelyikére, úgy hogy minden szomszédos él különböző színű.
Élkromatikus szám χ'(G) = k, ha G k-élszínezhető, de nem (k-1)-élszínezhető.

G=(V,E) gráf **síkbarajzolható**, ha van G-nek olyan diagramja, amin élek csak végpontban metszik egymást. G síkbarajzolható, ha létezik síkbarajzolása.
4-szín tétel: Minden egyszerű, síkbarajzolható gráf 4-színezhető.
5-szín tétel: Minden egyszerű, síkbarajzolható gráf 5-színezhető, azaz χ(G) ≤ 5.

8. tétel: Hálózati folyamok

Hálózat: (G,s,t,c(e)), ahol G irányított gráf, s és t G különböző csúcsai, és G minden élét jellemzi c(e) nem negatív szám – **az él kapacitása**. *Példa:* egy csővezeték, ahol s-ből t-be szállítjuk a folyadékot.

Folyam: olyan f függvény (G,s,t,c) hálózatban, ami G minden éléhez egy számot rendel, úgy hogy: - a folyam minden élen legfeljebb kapacitásnyi lehet (**kapacitás feltétel**) - minden s-től és t-től különböző pontra igaz, hogy a befolyó folyam összmennyisége azonos a kifolyóéval (**Kirchhoff-szabály**).

Folyamnagyság: az f folyam m_f nagysága az a nettó folyammennyiség, ami s-ből kifolyik.

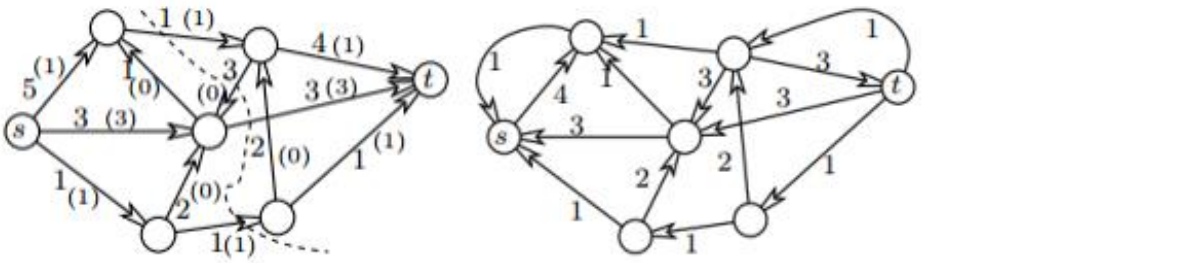
st-vágás: X a G csúcsainak egy s-t tartalmazó, de t-től diszjunkt részhalmaza. Az X és V(G)\X között futó éleinek egy halmazát a hálózat egy st-vágásának nevezzük. Az X által meghatározott **st-vágás kapacitása** az X-ből V(G)\X-be futó élek kapacitásösszege.

Ford-Fulkerson tétel: Ha (G,s,t,c) egy véges hálózat, akkor létezik egy f folyam, és egy $s \in X \subseteq$ V(G)\$\{t\}$ részhalmaz úgy, hogy az m_f folyamnagyság azonos az X által definiált st-vágás kapacitásával.

Az st-vágás tehát egy kézenfekvő eszköz annak bizonyítására, hogy a folyamnagyság nem lehet nagyobb egy adott mennyiségnél. Valójában ennél jobb bizonyíték nem is kell: a maximális folyamnagyság pontosan megegyezik a minimális vágáskapacitással. Ezt mondja ki a ” max-flow min-cut” (MFMC) tétel.

Bizonyítás: javító utas algoritmussal (A Ford-Fulkerson / javító-utas algoritmus alkalmas a maximális folyam – minimális vágás megkeresésére) :

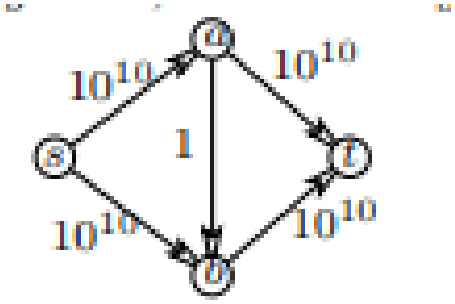
Gf segédgráfot hozunk létre, irányítsuk úgy G éleit, hogy ahol még növelhető a folyamnagyság, ott előreél, ahol pedig a folyam csökkenthető, ott visszaélek szerepelnek. A Gf segédgráfon definiáljuk a c(u,v)=c(u,v)-f(u,v) ha uv előreél, c(u,v)=f(v,u) ha uv visszaél, kapacitásokat. Az algoritmus addig fut, amíg van s-t út. Ha talál ilyen utat, akkor az út előre élein növeljük a folyamot, az út visszaéleinek megfordítottjain pedig csökkentjük. Ez alapján látszik, hogy ha nincs már s-t út, akkor maximális folyamot találtunk. Megkeressük azokat a pontokat, amik s-ből elérhetőek, ők alkotják az X halmazt. Ehhez a halmazhoz tartozó vágás minimális vágás lesz.



Hálózati folyam. A zárójelekben az f folyam által felvett értékek állnak. A folyamérték 1+3+1=5. A szaggatott vonal 5 értékű st-vágás. A másik ábra a segédgráf - már nem tartalmaz javító utat

Egészértékűségi lemma: ha a hálózatban minden kapacitás egész szám, akkor létezik olyan maximális folyam, ami a gráf minden élén egész értéket vesz fel. Az ilyen folyamot **egész folyamnak** nevezzük.

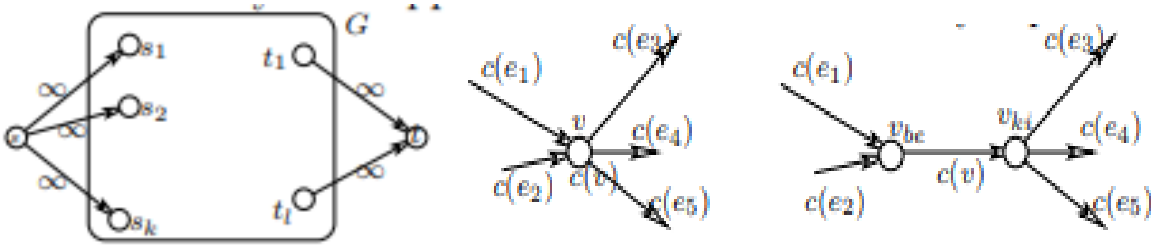
Edmonds-Karp tétel: ha a hálózatban a maximális folyamot javító utas algoritmussal keressük, és mindig egy legkevesebb élből álló út mentén növelünk, akkor a folyamat lépésszáma felülről becsülhető n^3 -al.



1. ábra: Illusztráció az Edmonds-Karp tételhez

Az Edmonds-Karp tétel tehát azt biztosítja, hogy a legrövidebb javító utakon maximális mértékű javításokat végrehajtva gyorsan találjunk maximális folyamot. Ha esztelenül próbálunk javítani, akkor indokolatlanul sok munkába kerülhet egy maximális folyam megtalálása: az ábrán látható hálózatban felváltva az sabt ill. sbat javító utakat választva mindig csak egységnyit tudunk emelni a folyamnagyságon, tehát az Edmonds-Karp algoritmus által két javítás után megtalált, $2 \cdot 10^{10}$ nagyságú maximális folyamot csillagászati számú lépés után találjuk csak meg.

A **többtermelős-többfogyasztós hálózatokat** vissza kell vezetni az ismert egytermelős-egyfogyasztós hálózatra végtelenhez hasonló terminálokkal. A folyamprobléma kiterjeszthető arra az esetre is, ha több forrásból több nyelőbe akarunk folyamot vezetni, de nincs megkötés arra, hogy melyik forrásból melyik nyelőbe érkezzék a folyam. Ha tehát $s_1, s_2, \ldots, s_k$ a források, $t_1, t_2, \ldots, t_l$ a nyelők, akkor bevezetünk egy-egy új s, illetve t csúcsot, majd s-ből minden s_i-be, illetve minden t_j-ből t-be vezetünk egy ∞ kapacitású élt.



1. ábra: Többtermelős-többfogyasztós és pontkapacitásos hálózatok

Értelmezhető az a folyamprobléma is, ahol nemcsak az éleknek, hanem a pontoknak is van kapacitásuk, ami felső korlát a ponton átfolyó folyammennyiségre. Ez a probléma is visszavezethető a szokásos folyamproblémára az alábbiak szerint.

Lehetséges általánosítás még, hogy a hálózatban irányítatlan élek is vannak, amelyeken mindkét irányban folyhat folyam. Ekkor bevezetve két, ellentétesen irányított élt az irányítatlan él két végpontja között az elhagyott irányítatlan él kapacitásával, akkor a probléma ismételten visszavezethető hálózati folyamokra.

9. tétel: Páros gráfok, párosítások, lefogó és független pont- és élhalmazok

Páros gráfok: G csúcsai két diszjunkt halmazra bonthatóak, úgy hogy minden él a halmazok között fut. Ekvivalens definíció: G akkor páros, ha $\chi(G) \leq 2$, azaz G két színnel színezhető.

Megfigyelések:

1. Páros hosszú kör páros, mert felváltva ki lehet színezni a pontjait.
2. Páratlan kör nem páros, mert mire körbeérünk, két azonos színű pont lesz egymás mellett.
3. Ha egy gráf páros, akkor a részgráfja is páros.
4. Páros gráf nem tartalmazhat páratlan kört.

Tétel: G véges gráf pontosan akkor páros, ha nem tartalmaz páratlan kört.
Következmény: Mivel a fában nincs kör, így minden fa páros.

Definíció: a G=(V,E) gráf éleinek M részhalmaza független, más szóval M (részleges) **párosítás**, ha az M-beli élek végpontjai különbözőek. M **teljes párosítás**, ha G minden csúcsát fedi.

A G=(V,E) gráf $X \subseteq V$ ponthalmaz **szomszédjainak számát** N(X) jelöli.

Hall tétele: a G=(A,B,E) véges, páros gráfnak pontosan akkor létezik A-t fedő párosítása, ha $|X| \leq |N(X)|$ minden $X \subseteq A$-ra.

Frobenius tétele: a G=(A,B,E) véges, páros gráfnak pontosan akkor létezik teljes párosítása, ha |A|=|B|, és $|X| \leq |N(X)|$ minden $X \subseteq A$-ra. **Bizonyítás:** triviálisan következik Hall tételéből.

Adott G gráf esetén **$\nu(G)$ jelöli a maximális független élhalmaz** méretét, azaz G maximális párosításának elemszámát.
Adott G pontjainak U halmaza **lefogó ponthalmaz**, ha G minden élének van U-beli végpontja.
Minimális lefogó ponthalmaz: $\tau(G)$.

Kőnig tétele: ha G=(A,B,E) véges, páros gráf akkor $\nu(G) = \tau(G)$.
Állítás: ha G véges gráf, akkor $\nu(G) \leq \tau(G)$.

Alternáló utas algoritmus maximális párosítás keresésére: kiindulunk az üres párosításból, és azt javítgatjuk. Ha már találtunk M párosítást, akkor tekintjük az ahhoz tartozó segédgráfot, azaz M éleit B-ből A-ba irányítjuk, G egyéb éleit fordítva. Ha ebben a segédgráfban létezik út egy fedetlen A-ból, eddig fedetlen B-be, akkor az ún. alternáló úton

az eddigi párosítás éleket elhagyva, és az út párosításon kívüli éleit bevéve egy eggyel nagyobb méretű párosítást kapunk. Ha nincs ilyen út, akkor M maximális párosítás.

Egy G gráf pontjainak U részhalmaza **független ponthalmaz,** ha U nem feszít élt. **A maximális független ponthalmaz elemszáma $\alpha(G)$.**
G gráf éleinek F részhalmaza **lefogó élhalmaz,** ha G minden pontjából indul F-beli él.
Minimális lefogó élhalmaz elemszáma $\rho(G)$.

Gallai tételei:

1. ha G-ben nincs hurokél, akkor $\tau(G)$ + $\alpha(G)$ = n
2. ha G-nek nincs izolált pontja: $\nu(G)$ + $\rho(G)$ = n

Kőnig tétele: ha G véges, páros gráfnak nincs izolált pontja, akkor $\alpha(G) = \rho(G)$.
Ha G véges gráf és nincsen izolált pontja, akkor $\alpha(G) \leq \rho(G)$.

$\alpha(G) \leq \rho(G)$	max. függtn.	min. lef.	(K) páros gráfra $\nu(G)=\tau(G)$
pontok	α	τ	(G1) ha G-ben nincs hurokél, akkor $\tau(G)+\alpha(G)=n$
élek	ν	ρ	(G2) ha G-nek nincs izolált pontja: $\nu(G)+\rho(G)=n$
$\nu(G) \leq \tau(G) \leq 2\nu(G)$			(K) páros összefüggő gráfra $\alpha(G)=\rho(G)$

10. tétel: Síkbarajzolhatóság, gömbre rajzolhatóság, külső tartomány nem kitüntetett volta, Euler-féle poliédertétel, Kuratowski-gráfok

Definíció: G=(V,E) gráf **síkbarajzolható**, ha van G-nek olyan diagramja, amin élek csak végpontban metszik egymást. G síkbarajzolható, ha létezik síkbarajzolása.

Egy síkbarajzolható (röviden sr) gráf által meghatározott síktartomány (röviden **tartomány**) a lap. **Külső tartománynak** nevezzük a nem korlátos – végtelen tartományt.

Tétel: G gráf pontosan akkor síkbarajzolható, ha **gömbrerajzolható**. (Ilyenkor nincs külső tartomány.)
Bizonyítás: sztereografikus projekcióval.

Tetszőleges síkbarajzolás bármely T tartományához létezik egy másik síkbarajzolás, amiben T a külső tartomány.

Tétel: tetszőleges konvex poliéder élhálója síkbarajzolható.
Bizonyítás: vetítsük az élhálót egy belső P pontból P középpontú gömbre, így az élháló gömbrerajzolható -> síkbarajzolható.

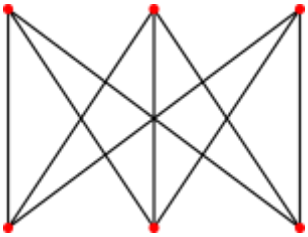
Tétel: Ha G síkbarajzolt gráf, akkor $n + t = e + k + 1$, ahol n a pontok, t a tartományok, e az élek, k pedig a komponensek száma.
Euler-féle poliédertétel: G síkbarajzolt, összefüggő gráf, akkor $n + t = e + 2$.
Következmények:

1. Ha G síkbarajzolható, akkor bármely síkbarajzolásának ugyanannyi tartománya van.
2. Ha G egyszerű, legalább 3 pontú, síkbarajzolható, akkor **e ≤ 3n – 6**.
3. Sem K_5 sem $K_{3,3}$ nem síkbarajzolható.

Topologikusan izmorf: H és G gráf, ha G megkapható H-ból ezt a két lépést ismételve:

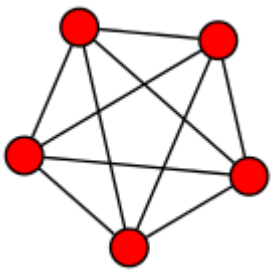
1. egy élt kettéosztunk egy ponttal.
2. törlünk egy pontot, és egy éllel összekötjük a két szomszédját.

G és H gráfok topologikusan izomorfak, ha H meghapható **soros bővítéssel** (élfelosztással) vagy annak fordítottjával ezeket tetszőlegesen sokszor alkalmazva.
Topologikus izomorfia nem befolyásolja a síkbarajzolhatóságot.



vagy $K_{3,3}$-al topologikusan

2. ábra: K3,3



3. ábra: K5

Kuratowski tétele: G gráf pontosan akkor síkbarajzolható, ha nem tartalmaz K_5-el, izomorf részgráfot.

11. tétel: Oszthatóság, legnagyobb közös osztó, euklideszi algoritmus, számelmélet alaptétele, prímek

Definíció: a és b egész számokról azt mondjuk, hogy a **osztja** b-t, vagy b az a többszöröse (a|b), ha b=aq valamely q egész számra.
Triviális osztó: $n \neq 0$, esetén ±1, $\pm n$ az n triviális osztói.
Valódi osztó: az n nemtriviális osztói.

Maradékos osztás: a és b egész számokra (ahol a≠0), b szám felírható úgy, hogy $b = a * \left\lfloor \frac{b}{a} \right\rfloor + m$, $0 \leq \frac{b}{a} - \left\lfloor \frac{b}{a} \right\rfloor < 1$, és $0 \leq m < b$ maradék adódik. *Például:* 111 9-es osztási maradéka 3.

Felbonthatatlan szám: $p \in Z$ szám felbonthatatlan, ha csak triviális osztói vannak, és $|p| \neq 1$.
Például: 2, -5, 11.

Állítás: bármely z egész szám előáll felbonthatatlan számok szorzataként, ha |z|>1.

A számelmélet alaptétele: ha egy z egész számra |z|>1, akkor z előáll felbonthatatlan egész számok szorzataként, és az ilyen előállítások csak a tényezők sorrendjében és előjeleiben különböznek.
Például: $-24 = 2 * 3 * (-2) * 2 = (-3) * (-2) * (-2) * 2$.

Kanonikus alak: $1 < n \in N$ szám kanonikus alakja $n = {p_1}^{\alpha_1} * {p_2}^{\alpha_2} * \ldots * {p_n}^{\alpha_n}$, ahol a p-k különböző (pozitív) felbonthatatlanok, α-k pedig pozitív egészek.

Prím: egy $p \in Z$ szám akkor prím, ha |p|>1, és csak úgy tud osztani egy szorzatot, ha a szorzat valamelyik tényezőjét osztja – p|ab akkor p|a, vagy p|b.
Számelmélet alaptételének következménye: a prím és a felbonthatatlan ugyanazokat a számokat jelöli.

Állítás: egy d szám pontosan akkor osztója az $n \in N$ számnak, ha d kanonikus alakjában pontosan ugyan azok a számok szerepelnek, mint n kanonikus alakjában, és minden ilyen prím kitevője d-ben legfeljebb akkora, mint n-ben.

Következmény: legyen $\prod_{i=1}^{k} {p_i}^{\alpha_i}$ az n kanonikus alakja. **Az n pozitív osztóinak száma** $d(n) = \prod_{i=1}^{k} (\alpha_i + 1)$.

Legyen $a, b \in Z$ olyan, hogy $a \neq 0$ vagy $b \neq 0$, és $a = = {p_1}^{\alpha_1} * {p_2}^{\alpha_2} * \ldots * {p_n}^{\alpha_n}$ és $b = = {p_1}^{\beta_1} * {p_2}^{\beta_2} * \ldots * {p_n}^{\beta_n}$.

Legnagyobb közös osztó: (a,b) az a legnagyobb szám, ami a-nak és b-nek is osztója. $(a, b) = {p_1}^{\min(\alpha_1, \beta_1)} * {p_2}^{\min(\alpha_2, \beta_2)} * \ldots * {p_n}^{\min(\alpha_n, \beta_n)}$ – azaz a közös prímek, a legkisebb hatványon.

Állítás: Ha a és b egészek, akkor (a, b) = (a−k*b, b).
Relatív prím: a és b relatív prímek, ha (a,b)=1. *Például:* 25 és 36.
Definíció: a és b **ikerprímek**, ha prímek, és különbségük 2. *Például:* 5 és 7.

Legkisebb közös többszörös: [a,b] az a legkisebb $n \in N$ szám, amire a|n és b|n. $[a, b] = {p_1}^{\max(\alpha_1,\beta_1)} * {p_2}^{\max(\alpha_2,\beta_2)} * \ldots * {p_n}^{\max(\alpha_n,\beta_n)}$ – azaz az összes prím. a legnagyobb hatványon.

Euklideszi algoritmus:
Input: a és b egészek (mondjuk $b \leq a$).
Output: (a,b).
Működés: Legyen $a_0 := a$, $a_1 := b$. Ha már meghatároztuk $a_0 \geq a_1 \geq \cdots \geq a_i$ számokat, akkor legyen $a_{i-1} = q_i * a_i + a_{i+1}$, azaz osszuk el maradékosan a_{i-1}-t a_i-vel, és legyen a_{i+1} a maradék. Az eljárás addig megy, amíg $a_{i+1} \neq 0$, ekkor (a, b) = a_k.
Például: (360, 225)=45

$$360 = 225 * 1 + 135$$
$$225 = 135 * 1 + 90$$
$$135 = 90 * 1 + \mathbf{45}$$
$$90 = 45 * 2 + 0$$

Tétel: a prímszámok száma végtelen.
Bizonyítás: Mivel n! az 1,2…n számok mindegyikével osztható, így N:=n!+1 relatív prím ezen számok mindegyikéhez, így N nem osztható az n-nél kisebb prímek egyikével sem.

Tétel: Tetszőlegesen hosszú sorozat képezhető szomszédos összetett számokból, azaz bármely $n \in \mathbb{N}$-re létezik olyan N, amire az N+1, N+2,…,N+n számok mindegyike összetett.
Másképpen: a **prímszámok között tetszőlegesen nagy hézag lehet.**

Csebisev tétel: Tetszőleges n pozitív egészre létezik p prím, $n < p \leq 2n$.

Nagy prímszámtétel:

$$\lim_{x \to \infty} \frac{\pi(x)}{\frac{x}{lnx}} = 1$$

ahol $\pi(x)$ jelöli az x-nél nem nagyobb prímek számát.

12. tétel: Kongruencia, maradékrendszerek, Euler-féle φ-függvény, lineáris kongruenciák

Definíció: a,b,m ∈ $\mathbb{Z}$, 0<m esetén azt mondjuk, hogy **a kongruens b modulo m**, ha m|a-b.
Jelölése: $a \equiv b(mod\ m) \rightarrow a \equiv b(m)$ Pl.: 12 ≡ −3(5).
Megfigyelés: Az egész számok halmaza modulo m **maradékosztályokra** partícionálható.
Például: 3-mal kongruens számok modulo 5: 3, 8, 13, 18… és -2, -7, -12…
A maradékosztályok jelentősége az, hogy ha két szám azonos maradékosztályba esik (modulo m), akkor kongruensek egymással modulo m, egyébként pedig nem.

Két kongruencia összeadható, és összeszorozható: ha $a \equiv b(m)$ és $c \equiv d(m)$,
akkor $a + c \equiv b + d(m)$, és $ac \equiv bd(m)$.
Ha d|a és d|b, akkor $\frac{a}{d} \equiv \frac{b}{d}\left(\frac{m}{(m,d)}\right)$, azaz kongruencia osztásakor a modulust is osztjuk, az osztó és a modulus legnagyobb közös osztójával.

Következmények:
1. $a \equiv b(m)$ pontosan akkor teljesül, ha $a + k \equiv b + k(m)$.
2. ha d relatív prím az m-hez, akkor $a \equiv b(m)$ kongruencia ekvivalens az $ad \equiv bd(m)$ kongruenciával, tehát kongruencia szorzása csak akkor ekvivalens átalakítás, ha a modulushoz relatív prímszámmal szorzunk.
3. Ha d>0 rögzített egész, akkor $a \equiv b(m)$ ekvivalens $ad \equiv bd(md)$.

Megfigyelés: ha $a \equiv b(m)$, akkor (a,m)=(b,m).

TMR: rögzített m>1 esetén az m elemű, $T = \{a_1, a_2, \ldots, a_m,\}$ halmazt modulo m **teljes maradékrendszerének** nevezzük, ha minden m szerinti maradékosztályból pontosan egy elemet tartalmaz. *Például:* Modulo 10 TMR a {100, 21, −21, 42, −42, 13, −13, 44, 55, 66} halmaz.
RMR: $R \subset \mathbb{Z}$ halmaz **redukált maradékrendszer** modulo m, ha minden m-hez relatív prím m szerinti maradékosztályból pontosan egy elemet tartalmaz, mérete: $\varphi(m)$.
Például: Modulo 12 RMR a {17, -5, 11, -11} halmaz.

Euler féle *φ*-függvény: egy adott egész számhoz a nála kisebb relatív prím pozitív egész számok számát adja meg – ez az RMR mérete.

Megfigyelés: $\varphi(m)$ megegyezik az 1 és m közé eső m-hez relatív prímszámok számával.

- p prímre $\varphi(p) = p - 1$
- p prímre $\varphi(p^\alpha) = p^\alpha - p^{\alpha-1}$
- Ha (a,b)=1, akkor $\varphi(a * b) = \varphi(a) * \varphi(b)$

Euler-Fermat tétel: Ha (a,m)=1, akkor $\alpha^{\varphi(m)} \equiv 1(m)$.
Kis Fermat tétel: ha p prím, akkor bármely a egészre $a^p \equiv a(p)$.

Lineáris kongruencia: $ax \equiv b(m)$, ahol a és b adott egészek, m pedig adott pozitív egész. A lineáris kongruencia megoldása azt jelenti, hogy meghatározzuk mindazon egészeket, amelyeket x helyébe írva a kongruencia igaz lesz.

Tétel: $ax \equiv b(m)$ **lineáris kongruencia pontosan akkor oldható meg**, ha (a,m)|b. A kongruencia megoldáshalmaza (a,m) darab modulo m maradékosztály.

Lineáris kongruencia lehetséges megoldási módjai:

1. Euklideszi algoritmust felhasználva.
2. ha a modulus elég kicsit, akkor TMR minden elemét behelyettesítve, pontosan azok a maradékosztályok alkotják a megoldáshalmazt, amelyeknek reprezentánsait behelyettesítve teljesül a kongruencia.
3. ha ismert m kanonikus alakja, akkor $\varphi(m)$-et kiszámítva, az Euler-Fermat tételt kihasználva tudjuk megoldani a kongruenciát, a leosztás után, amikor is már (a,m)=1 teljesül. Így beszorozhatjuk mindkét oldalt $a^{\varphi(m)-1}$, m-hez relatív prím számmal. $a^{\varphi(m)-1} * ax \equiv a^{\varphi(m)-1} * b(m)$, azaz megkapjuk a lineáris kongruencia egyértelmű megoldását.
4. **leggyakoribb módszer:** az a együttható abszolút értékét csökkentjük egészen 1-ig ekvivalens átalakításokkal.
    a. a-t vagy b-t vele kongruens másik számmal helyettesítünk.
    b. ha (a,b)>1, akkor osztunk, szükség esetén a modulust is.
    c. a modulushoz relatív prímmel szorzunk – és a modulushoz nem nyúlunk.

Például:

$62x \equiv 24(36)$ *kongruencia. Mivel* $62 \equiv 26(36)$, *ezért a*
$26x \equiv 24(36)$ *kongruenciát kapjuk.* $(26, 36) = 2$, *tehát osztunk:*
$13x \equiv 12(18)$ *adódik. Sajnos nem szorozhatunk* 2, 3 *ill.* 4*-gyel, így inkább* $13 \equiv -5(18)$*-t helyettesítünk:*
$-5x \equiv 12(18)$, *majd szorzunk* (-1)*-gyel, mert nem szeretjük a negatív együtthatót.*
$5x \equiv -12(18)$, *ismét helyettesítünk:*
$5x \equiv 6(18)$ *Most jó lenne* 4*-gyel szorozni, hogy* 2 *legyen az együttható, de ezt nem tehetjük, hisz a* 2 *nem relatív prím* 18*-hoz. Viszont ügyesen észrevesszük, hogy* 7*-tel szorozhatunk:*
$35x \equiv 42(18)$, *és megint helyettesítünk:*
$-x \equiv 6(18)$, *szorzunk* (-1)*-gyel:*
$x \equiv -6 \equiv 12(18)$. *Most már csak a* 36 *modulusra kell áttérni:*
$x \equiv 12(36)$ *vagy* $x \equiv 12 + 18 = 30(36)$, *győztünk.*

(Diofantoszi egyenlet: ax+by=c ahol a ,b ,c adott egészek, x, y ismeretlen egészek. Diofantoszi egyenlet megoldható kongruenciák használatával.

Például:

$7x + 5y = 29$
$7x \equiv 29(5)$ 7 ≡ 2(5) miatt:
$2x \equiv 29(5)$ 29 ≡ 4(5) miatt:
$2x \equiv 4(5)$ (2,4)=2 miatt:
$x \equiv 2(5)$
Ekkor x=5k+2 visszahelyettesítve:
$y = \frac{29-7*(5k+2)}{5} = \frac{15-35k}{5} \rightarrow \mathbf{y = 3 - 7k}$)

13. tétel: Algoritmusok bonyolultsága, döntési problémák, bonyolultsági osztályok, polinomiális visszavezethetőség, nevezetes NP-teljes problémák

$\boldsymbol{f_A(n)}$ az A algoritmus **legnagyobb lépésszáma** n hosszúságú input esetén.

Egy algoritmus **polinomiális**, ha lépésszáma felülről becsülhető a bemenet méretének polinomjával.
Egy algoritmus **exponenciális,** ha a futásideje exponenciális függvénye lehet a bemenet hosszának.
Például: összeadás és szorzás polinomiális, hatványozás exponenciális.

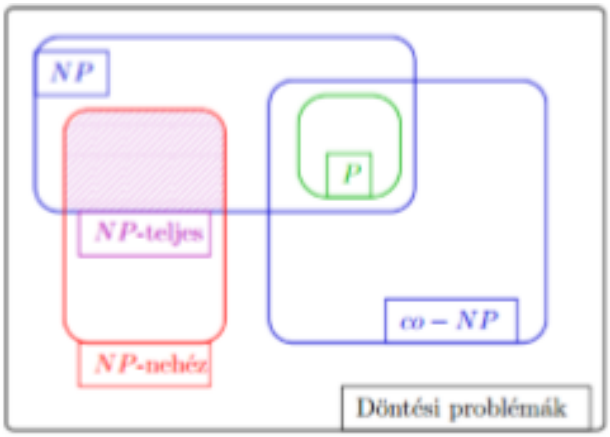
Döntési probléma: adott bemenetre igen vagy nem választ ad.
Például: prímtesztelés, bemenete egy szám, kimenete pedig egy bit, ami 1 ha prím, 0 ha nem.

Bonyolultsági osztályok:
P: a polinom időben eldönthető döntési problémák osztálya.
Például: Van-e Euler kör a gráfban? Bemenet egy n pontú, m élű gráf (bemenet mérete n+4m). Azt ellenőrizzük, hogy összefüggő-e a gráf, valamint, hogy minden pont foka páros-e. Ezeket meg lehet tenni polinomiális algoritmusokkal. Létezik tehát $c * (n + m)$ hosszú algoritmus a probléma eldöntésére.
További példák: gráf összefüggősége (BFS-el), k-méretű párosítás létezése, k nagyságú folyam létezése (max. folyam kereséssel), PERT elvégezhető-e k idő alatt (optimális ütemezés keresése), síkbarajzolható-e a gráf.

NP: az olyan π döntési problémák osztálya, melyekre létezik p_πk - polinom minden egyes olyan b bemenetre, amire igen a válasz, hogy ez legfeljebb $p_\pi(|b|)$ lépésben bebizonyítható.
Például: Hamilton-teszt esetén a bizonyíték a Hamilton-kör leírása volt, ugyanis ennek ismeretében $c * n$ lépésben bizonyítható, hogy igen a válasz.

co-NP: az olyan π döntési problémák osztálya, melyekre létezik p_πk - polinom minden egyes olyan b bemenetre, amire nem a válasz, hogy ez legfeljebb $p_\pi(|b|)$ lépésben bebizonyítható.

Megfigyelés: ha $\pi \in P$ akkor $\pi \in NP$ és $\pi \in co - NP$ egyaránt igaz. Azaz $\pi \in NP \cap co - NP$.
Sejtés: ha $\pi \in NP \cap co - NP$ akkor $\pi \in P$ is igaz. (*Például:* létezik-e teljes párosítás G-ben?)



Visszavezethetőség: Van két döntési probléma, π és π' és π-re létezik egy A algoritmusunk. Elképzelhető, hogy π'-re tudunk olyan A' algoritmust konstruálni, ami felhasználja az A algoritmust, azaz A' felírja A egy bemenetét, és meghívja A-t. Ha a meghívást egy lépésnek számítva A' polinomiális, akkor π' (polinomiálisan) visszavezethető π-re.
Megfigyelés: ha π'' polinomiálisan visszavezethető π'-re, és π' pedig π-re, akkor π'' is polinomiálisan visszavezethető π-re.
Állítás: ha $\pi \in P$ és π' polinomiálisan visszavezethető π-re akkor $\pi' \in P$.

NP-nehéz: π probléma NP-nehéz, ha bármely NP beli π' probléma polinomiálisan visszavezethető π-re.
NP-teljes: π probléma NP-teljes, ha π NP-nehéz, és $\pi \in NP$.
Sejtés: $P \neq NP$.

Cook-Levin tétel: a SAT probléma NP-teljes.
Következtetés: ha sikerül egy NP-beli problémára visszavezetni SAT-ot (vagy egy már korábban SAT segítségével NP-teljesnek nyilvánított problémát), akkor az adott probléma is NP-teljes. Tehát ha bármely NP-teljes problémára sikerülni polinomiális megoldást találnunk, akkor az összes NP-beli probléma megoldhatóvá válna polinom időben – nem túl valószínű.

Nevezetes NP-teljes problémák:

- **HAM probléma:** inputja egy gráf, és a válasz igen, ha létezik Hamilton-út a gráfban. Korábban már példaként megmutattuk, hogy **NP-beli. NP-teljes**, mert 3-SAT probléma polinomiálisan visszavezethető rá.
- **3-SZÍN probléma:** bemenete egy G gráf, és a válasz igen, ha G 3-színezhető. **NP-beli**, mert az igen válasz polinom időben bebizonyítható: egyszerűen megadjuk G egy 3-színezését. **NP-teljes:** 3-SAT probléma visszavezethető rá.
- **k-SZÍN probléma:** bemenete egy G gráf, és a válasz igen, ha G k-színezhető. Ha k>3, akkor **NP-beli**, mert az igen válasz polinom időben bebizonyítható: egyszerűen megadjuk G egy k-színezését. **NP-teljes:** 3-SZÍN probléma visszavezethető rá.
- **MAXFTN probléma:** bemenete egy G gráf, és k szám, és a válasz annak függvénye, hogy G-ben van-e k független csúcs. A probléma **NP-beli** – ha mutatunk k független csúcsot, akkor polinom időben bebizonyítható, hogy igen a válasz. **NP-teljes**, mert 3-SZÍN probléma visszavezethető rá.
- **MAXKLIKK probléma:** bemenete egy G gráf, és k szám, és a válasz annak függvénye, hogy G-ben van-e k méretű klikk. A probléma **NP-beli** – egy k méretű klikk megadása után polinom időben bizonyítható, hogy az adott pontok G-ben klikket alkotnak. **NP-teljes,** mert MAXFTN probléma visszavezethető rá.

14. tétel: Számelméleti algoritmusok, prímtesztelés, Fermat-teszt, titkosírás, digitális aláírás, RSA-módszer

$\boldsymbol{f_A(n)}$ az A algoritmus **legnagyobb lépésszáma** n hosszúságú input esetén.

Egy algoritmus **polinomiális**, ha lépésszáma felülről becsülhető a bemenet méretének polinomjával.
Egy algoritmus **exponenciális,** ha a futásideje exponenciális függvénye lehet a bemenet hosszának.
Például: összeadás, szorzás és Euklideszi-algoritmus polinomiális, hatványozás exponenciális.

Számelméleti algoritmusok:

- **Összeadás:** a műveletigény minden helyiértéknél legfeljebb 2, hisz két számot adunk össze az adott helyiértéken, plusz még egy esetleges maradékot az előző helyiértékről. A lépésszámra felső korlát tehát a $2 * \max(\log n, \log m) < 2 * (\log n + \log m)$, ami lineáris, vagyis **polinomiális.** *A kivonásra hasonló igaz.*
- **Szorzás:** a szokásos írásbeli szorzás működik, és megvalósítható log n darab összeadással, ahol minden összeadandó az m egyjegyű számmal összeszorzott többszöröse. Egy egyjegyű számmal m-et 2 log m db lépésben össze lehet szorozni, ugyanis minden jegyet szorozni kell, és az esetleges maradékot a szorzathoz kell adni. Tehát az össz lépésszám $2(\log n)(\log m) < (\log n + \log m)2$, vagyis a szorzás **polinomiális.** *Az írásbeli osztás is polinom időben elvégezhető.*
- **Hatványozás:** az n^m szám számjegyeinek a száma $k * 2^l$, ahol k és l az n, illetve m jegyeinek a száma kettes számrendszerben. Mivel itt a bemenet mérete k+l, ezért a végeredményt még leírni sem tudjuk a bemenet hosszával, tehát a hatványozásra nincs polinomiális algoritmus, tehát **exponenciális**.
- **Hatványozás modulo m:** az input n, k és m és a cél $n^k(mod\ m)$ meghatározása. Legyen $k = \sum_i k_i 2^i$ azaz $k = \cdots k_2, k_1, k_0$ kettes számrendszerbeli alak. Sorra kiszámoljuk a 0 és n – 1 közé eső $n_0, n_1, n_2 \ldots$ számokat, ahol $n_0 \equiv n(m)$, $n_1 \equiv n^2(m), \ldots, n_i \equiv n^{2^i}(m)$. Az n_{i+1}-et az $n_{i+1} \equiv n_i^2(m)$ alapján egy szorzással, és egy maradékos osztással kaphatjuk, ráadásul n_i mérete mindig legfeljebb log m lesz. Tehát egy n_i kiszámítása mindig legfeljebb egy log m méretű szám négyzetre emelését, és egy legfeljebb 2 log m méretű eredmény maradékos osztását igényli. A szükséges n_i-k kiszámításához mindezt log k-szor kell megtenni. Az n^k meghatározását pedig $n^k = \prod_{i=0}^{\infty} n^{k_i 2^i} \equiv \prod_{i=0}^{\infty} {n_i}^{k_i}(m)$ alapján további, legfeljebb log k darab, legfeljebb log m méretű szám szorzásával, és legfeljebb log k darab, legfeljebb 2 log m méretű szám maradékos osztásával kapjuk. Ebből látszik, hogy a modulo m hatványozás **polinomiális.**

- **Euklideszi algoritmus:** Az euklideszi algoritmus egy lépésében adott $a_i \leq a_{i+1}$ esetén kell egy maradékos osztást elvégezni, és meghatározni azt a $0 \leq a_{i+2} \leq a_{i+1}$ -et, melyre $a_i = q_{i+1} * a_{i+1} + a_{i+2}$ áll. Az a_i mérete legfeljebb akkora, mint a_0 és a_1 mérete közül a nagyobbik, tehát az euklideszi algoritmus mindegyik lépése polinomiális időt igényel. A nagy észrevétel, hogy $a_{i+2} \leq \frac{a_i}{2}$, ezért a fentieket legfeljebb $\log a_0$-szor kell elvégezni, amitől az eljárás **polinomiális** marad.

Euler-Fermat tétel alapján: Ha n prím, akkor $k^{n-1} \equiv 1(n)$ bármely k esetén.

- n szám **árulója** az a k, amire (k,n)=1 és $k^{n-1} \not\equiv 1(n)$, ha van áruló, akkor a számok legalább fele áruló.
- **Leleplező** az a k, amire $(k,n) \neq 1$. Nyilván ezekre is igaz, hogy $k^{n-1} \not\equiv 1(n)$.
- k **cinkos,** ha n összetett, és $k^{n-1} \equiv 1(n)$.

Prímtesztelés: Fermat-teszt – nagy pontossággal (de nem tökéletes bizonyossággal) határozza meg egy számról, hogy prím-e.
Működése: Véletlenül választunk egy $0 < k < n$ számot, ha $k^{n-1} \not\equiv 1(n)$, akkor kész vagyunk, n összetett, ha $k^{n-1} \equiv 1(n)$, akkor valószínűsítjük, hogy k prím. **A teszt hibázhat, de csak az lehet a hibája, hogy egy összetett számot prímnek mond, fordítva soha.** Valamint ha létezik áruló, akkor a hiba lehetősége legfeljebb $\frac{1}{2}$. Ha tehát m-szer választunk, akkor a hiba lehetősége legfeljebb $\frac{1}{2^m}$. Többször megismételt Fermat teszt **polinomiális** számú, polinom időben elvégezhető lépést használt.

Léteznek olyan számok, úgynevezett álprímek, vagy **Charmichael számok**, amelyeknek csak cinkosai és leleplezői (elenyésző számban) vannak. Ezeket a Fermat-teszt majdnem biztosan prímnek találja. *Például:* 561.

Egyirányú függvény: egyirányú függvénynek nevezünk egy f: {1,2,…,n} → {1,2,…,n} függvényt, mely hatékonyan számolható, de f^{-1}kiszámítása pusztán f ismeretében reménytelen. Elképzelhető, hogy f^{-1} kiszámítására is létezik hatékony eljárás (persze nem pusztán f ismeretében), ezeket hívjuk **kiskapus egyirányú függvényeknek**. Ezekre épül a nyilvános kulcsú titkosírás.

Nyilvános kulcsú titkosírás: rögzítünk egy Σ-val jelölt ABC-t: ennek a jeleivel írjuk le az üzenetet. A kódolandó M üzenetről feltehető, hogy t betűből áll, tehát $M \in \Sigma^t$, hiszen a hosszabb üzeneteket t hosszúságú blokkokra vághatjuk. Feltehetjük, hogy Σ^t szavai 1 és $|\Sigma|^t$ közötti természetes számoknak felelnek meg. A nyilvános kulcsú titkosírási rendszert egy olyan kiskapus egyirányú f: {1,2,…,n} → {1,2,…,n} függvény írja le, melyre $n \geq |\Sigma|^t$. Ezt a leképezést egy **nyilvános kulcs** segítségével megadjuk, és bárki számára hozzáférhetővé tesszük. Feltételezzük, hogy az A-nak nevezett címzett képesf^{-1} hatékony számítására, mert rendelkezik az azt leíró titkos kulccsal. Ha tehát akarunk A-nak küldeni egy titkosított M üzenetet, akkor elküldjük neki $M' = f(M)$-et, ő pedig hatékonyan ki tudja számítani f^{-1} (M') = M-et. Aki lehallgatná ezt az üzenetet, f ismeretében csak arra képes, hogyha sejti, hogy mi az üzenet, akkor ellenőrizni tudja, hogy valóban az-e.

Digitális aláírás: segítségével bizonyítható, hogy az üzenet kitől érkezett. Tegyük fel, hogy minden szereplőnek van kiskapus egyirányú függvénye, A-é f_A, B-é pedig f_B. Ha B alá akarja írni az A-nak küldött M üzenetet, akkor A-nak az $M' = f_B^{-1}(M)$-et küldi el, ezt A vissza tudja fejteni a nyilvános f_B segítségével: $f_B(M') = f_B(f_B^{-1}(M)) = M$. Ez alapján bizonyítható, hogy az üzenet B-től érkezett, anélkül, hogy B-nek fel kéne fednie a titkos kulcsát.

RSA titkosítási módszer: először választunk két kellően nagy prímszámot, p-t és q-t, elég nagyot ahhoz, hogy $n := pq$ ne legyen faktorizálható, valamint $n \geq |\Sigma|^t$ teljesül. Legyen $m := \varphi(n) = (p-1) * (q-1)$, és válasszunk egy $1 \leq e \leq m$ számot úgy, hogy (e,m)=1. Legyen $f(M) := M^e (mod\ n)$. n és e lesz a nyilvános kulcs, ezt közhírré lehet tenni. p, q és m számok titokban maradnak! A kapott üzenet megfejtéséhez szükség van f^{-1} hatékony számítására. d-re meg kell oldani $ed \equiv 1(m)$ kongruenciát, amit (e,m)=1 miatt hatékonyan és egyértelműen meg lehet tenni. A d meghatározásával megkaptuk az (n,d) titkos kulcsot. Ha kapunk egy $M' = f(M)$ kódolt üzenetet, akkor az inverz leképezés: $f^{-1}(M') = M'^d (mod\ n)$.